SESSION

GRID RESOURCE MANAGEMENT AND INFRASTRUCTURES + ACCESS CONTROL + ACCOUNTING SERVICES

Chair(s)

TBA

A Modular Access Control Architecture for the Earth System Grid Federation

Philip Kershaw[1], Rachana Ananthakrishnan[2], Luca Cinquini[3 4], Dennis Heimbigner[5], and Bryan Lawrence[1]

[1]STFC Rutherford Appleton Laboratory, NCAS/British Atmospheric Data Centre, Didcot, Oxfordshire, United Kingdom

[2]Argonne National Laboratory, Argonne, IL, USA,

[3]Jet Propulsion Laboratory, National Aeronautics and Space Administration, Pasadena, CA, USA

[4]Earth System Research Laboratory, National Oceanic and Atmospheric Administration, Boulder, CO, USA

[5]University Corporation for Atmospheric Research, Boulder, CO, USA

We present key aspects of the federated access control solution required for the Earth System Grid Federation (ESGF), including a standard mechanism for securing OPeNDAP-based services and corresponding extensions to the NetCDF software libraries to support this paradigm.

ESGF is an international collaboration to enable access to Earth science data – beginning with a deployment in support of the Coupled Model Intercomparison Project, phase 5, a framework of climate model experiments whose results will be available in a distributed, globally accessible archive. By maintaining a separation of concerns between the various aspects of the system, it has been possible to devise a highly flexible access control architecture adaptable to the spectrum of needs presented. Such a modular approach is only possible through the definition of interfaces: at the inter-organisational level with web services and at the application level with the use of server side middleware and REST-based principles.

Keywords: Security, OPeNDAP, NetCDF, ESGF, CMIP5

1 Introduction

The production, evaluation and interpretation of climate model simulations are integral activities within Earth system science. Since the very first General Circulation Models run more than forty years ago, to the very latest Earth System Model simulations running now as part of the Coupled Model Intercomparison Project, phase 5 (CMIP5), these activities have always been on the leading edge of computing. As the models have improved, adding more internal processes, and running at higher resolution, so has the volume of data produced increased.

CMIP5, organised under the auspices of the World Climate Research Programme (WCRP), will deliver science that will feed into the next Intergovernmental Panel on Climate Change (IPCC) assessment report. As such, the analysis and interpretation will be a global activity, requiring global access to petascale data archives held on multiple continents. Traditional centralised archive solutions will clearly not suffice.

In order to address this challenge, a global federation – the Earth System Grid Federation (ESGF) has been built on the nucleus of the U.S. Earth System Grid Center for Enabling Technologies (ESG-CET)[1]. The ESGF was established to cope with data production at O(25) sites globally conforming to O(50) distinct numerical experiments and resulting in O(100,000) years of simulated climate corresponding to O(6500) years of the real-world climate.

A key tenet of the design philosophy of CMIP5 was to identify the "core" output from the simulations – that is the data which was likely to see the most analysis by scientists. The consequential key requirement for ESGF has been to maximise the exposure of that core data (expected to be approximately 2.5 petabytes), even as it exposes all the data produced for CMIP5. Thus, ESGF is essentially a federation of the originating modelling archives (or their proxies), and a number of replicant archives, three of which have committed to persist that core data indefinitely. These persistent archives will be located at the Program for Climate Model Diagnosis and Intercomparison (PCMDI) at the U.S. Lawrence Livermore National Laboratory, the British Atmospheric Data Centre (BADC) in the UK National Centre for Atmospheric Science, and the German Climate Computing Centre (DKRZ). ESGF itself is described in detail elsewhere[2].

The purpose of this paper is to describe particular techniques used to design the access control to the data in this globally distributed data system. This presents significant challenges highlighted by the heterogeneous nature of the environment in which a solution must be applied. This diversity is expressed on a number of levels: the range of tools and services used within the climate model community, the associated protocols and technology stacks employed, and the varied organisational structures representative across the federation members. We focus then on various aspects of the security architecture used to address these challenges: (1) a service oriented architecture; (2) HTTP based services and key applications; (3) the NetCDF[3] client implementation of the OPeNDAP[4] protocol and finally, (4) a walkthrough of a use case for data download, illustrating how the various components function together in the working system.

2 Requirements

ESGF has six key requirements that motivated the security solutions provided:

1) Seamless access to data hosted by all organisations in the federation, that is, single sign-on such that the same credentials can be used across the federation

2) A mechanism to set policy on restricting access to chosen datasets, per dataset on a case by case basis

3) The ability to notify users of changes to data and services. This requires the collection of user attributes including e-mail addresses whilst at the same time respecting user privacy.

4) The ability to collect metrics about data download, specifically the number of unique downloads.

5) Seamless integration with multiple interfaces to a service or resource, specifically, browser-based access and thick client access.

6) Clean integration with services and tools that scientists commonly use.

These requirements are considered in turn in the following sections, but we begin by outlining the overarching deployment environment and architectural requirements. The ESGF architecture defines Data Nodes and Gateway Nodes. Data Nodes are sites that host the model data and associated access services. Replication services enable the CMIP5 core data to be mirrored across the key archiving sites and publishing services make that data discoverable through Gateways - portals to the system.

For requirement (1), we look at the application of a service-oriented architecture, and requirements (2), (3) and (4) are addressed in attribute management and authorisation solutions. For requirement (5), ESGF includes both GridFTP[5] and HTTP based data access services. However, for the purposes of this paper we concentrate on the application of access control functionality to the HTTP server side architecture. Finally, requirement (6) focuses on work carried out for ESGF to add a standardised access control layer to OPeNDAP, which is a core data access service for the federation.

PCMDI has a lead role for CMIP5, holding the delegated authority of the various modelling groups to allow access according to their varying access criteria (co-ordinated by the WCRP). As such it needs to control the assignation of CMIP5 access authorisation, on a dataset-by-dataset basis, to individuals in the user community. Each ESGF institution may host CMIP5 datasets other than their own, but in doing so they need to honour the PCMDI role, even as they retain control over their own datasets, and those under other authorisation domains. Much of the data will be available with liberal licensing conditions. ESGF currently enforces a simple registration policy for CMIP5 access, coupled with the requirement for an e-mail address that can be validated. Thus, the level of assurance required is low in comparison to that of many systems. Even so, for resource providers, the security architecture should provide some level of protection for their finite computing assets, for example from malicious or unintended requests which might overload network or server resources.

We emphasise that in order to function as a federation, ESGF must have the ability to collect, curate, and publish trusted federation service metadata.

3 Service-Oriented Architecture

ESGF is deployed in various locations, alongside existing activities. Fundamental then to the development of a federated access control infrastructure is the interfaces between organisations. A standards-based approach was employed wherever practicable to facilitate interoperability and ensure the use of peer-reviewed protocols. In this section we describe the services and their interfaces, looking in turn at authentication and single sign-on, attribute management and authorisation.

3.1 Authentication and Single Sign-on

The distributed nature of the ESG architecture meant that single sign-on was favoured from the outset as a means to simplify access for users and join the user management infrastructures of the different participating institutions together. The OpenID[6] standard was chosen early by the ESG team to provide single sign-on capability[7]. An evaluation exercise showed that particular vulnerabilities in the specification could be addressed by stipulating SSL for OpenID Provider endpoints. As a consequence, ESGF OpenID Relying Parties are able to utilise SSL-based peer authentication to whitelist OpenID Provider identities to a given set of registered Identity Providers (IdPs) within the federation. The restricted set of IdPs allowed ESGF to leverage an agreed set of site attributes, and to enforce trust and service level agreements on the IdP. Each ESGF Gateway Node hosts an OpenID Identity Provider, where a user can register to get a login account.

OpenID is augmented with the use of SAML[8] (the Security Assertion Mark-up Language) v2.0 with the SOAP (Simple Object Access Protocol) binding to provide standard interfaces for the various other security services required to broker access. As a baseline, all interactions with services are secured with Transport Layer Security (TLS), with mutual authentication. Whitelisting of client certificate subject names enables services to restrict queries to a trusted set of retrievers.

3.1.1 Dual Authentication Mechanisms

While OpenID is suited for interaction with browser clients, it does not lend itself well to use with thick clients. To support the latter, each Gateway Node site runs a MyProxy[9] Online CA service. This can issue short-lived X.509 credentials which can be used with PKI-aware applications. The Online CA is backed by the same user authentication system as the OpenID service, thus issuing a

certificate to any user who has a valid OpenID login. Certificates issued from the MyProxy server are configured to include the respective OpenID URI in the certificate subject name. These credentials are used for authentication with GridFTP servers and also HTTP based applications including OPeNDAP-based services as will be described later in this paper.

3.2 Attribute Management

User attributes are exchanged between trusted parties within the federation. They fall into two categories which derive directly from requirements (2), (3) and (4) listed in Section 2:

- Site attributes include a limited amount of personal user information used for registration and notification purposes and are specific to the user's IdP.

- Virtual Organisation (VO) attributes include access control attributes used to restrict access to data. They are scoped for the community and may be assigned at some other ESGF authority than their IdP via a registration process.

VO-level attribute agreements were necessitated for two key use cases: access to the distributed CMIP5 data archive and bulk replication of data between archiving sites. For CMIP5 data access, PCMDI has authority to issue users with access rights. For the replication use case, the originating site of the data to be replicated has authority. In all cases, attributes names are namespace constrained to ensure enforcement of the issuing authority.

3.2.1 Push and Pull Models for Attribute Retrieval

As we explored these use cases, it became apparent that the system would benefit from both push and pull models for the transmission of attributes to consumers. Attributes may be pushed at the authentication stage with OpenID via the AX (Attribute Exchange) mechanism or with PKI based authentication by including a SAML assertion as an extension in a user certificate[11]. The latter was applied for the replication use case where the authorisation layer of the GridFTP service, can extract the attribute assertion to determine access for a given resource.

In some scenarios, a pull model is more suited such as where attribute information is required out of band of the authentication process or where the source of authority for attribute information is not itself an IdP. Any authority such as PCMDI, which has responsibilities for a specific data set or a group of data sets, may enroll users with the corresponding access attributes. They provide a registration interface for this purpose and also a SAML-based attribute service. This interface enables consumers to query user attribute entitlement. These services are associated with the resources they protect, and authorities may have users registered with them from a number of different IdPs from within the federation[12]. Attribute Services use whitelist techniques, based on the federation trusted service metadata (see Section 3.4), to restrict access to user attributes and preserve user privacy.

3.3 Authorisation Service

Each organisation within ESGF that hosts secure services (e.g. OPeNDAP or GridFTP), also hosts an authorisation service which exposes a SAML interface allowing authorised remote entities in the ESGF to query for decisions on access to given resources. This service supports a pull model to obtain user attributes. A registry maps user attribute names onto their respective issuing attribute service, so that for example, a resource secured with a CMIP5 attribute will trigger a query to the PCMDI Attribute Service to verify the user's entitlement to this attribute.

3.4 Federation Metadata

An essential aspect of any federation is the establishment and curation of federation credentials, which provides the core trust roots of the federation. In the case of ESGF, for authentication purposes the following metadata is required:

1) Trusted CA certificates, Signing Policy, and CRLs (Certificate Revocation Lists).

2) The whitelist of OpenID Identity Providers.

1) is used to validate any certificate chain presented in on a TLS channel (both for client to service, and service-to-service communication). 2) is used by OpenID Relying Parties to restrict which IdPs can assert user identities in the federation. In addition, the metadata also contains information about the various trusted services, including attribute and authorisation services, and data download services, which may query them. We have defined a schema to describe the data, and are in the process of building an infrastructure that will allow each organisation to own and register their metadata, and obtain the complete federation data for their use.

3.4.1 Service Discovery

OpenID 2.0, supports the Yadis[13] protocol whereby a HTTP GET request for a user's OpenID yields an eXtensible Resource Descriptor Service (XRDS) document containing the service endpoint for the respective OpenID Provider. XRDS can be further exploited to advertise multiple identity services. This has been leveraged for ESGF so that a given user's OpenID may be introspected to discover MyProxy server and attribute service endpoints associated with their IdP.

4 Modular Architecture for HTTP Based Services

In this section, we describe the architecture adopted for integrating security with the HTTP-based access services. Prior to the work with ESGF, lessons drawn from previous software development projects at the BADC[14]

had highlighted the need for non-intrusive approaches to access control for HTTP services - the layering of access control functionality over services in such a way as to minimise the impact on their existing interfaces. Two strong themes emerged: the use of REST[15]-based principles to govern access control policy and the use of Aspect Oriented Programming (AOP)[16] techniques.

Security is often cited as an exemplar for AOP. HTTP server-side interface specifications like the Python WSGI[17] (Web Server Gateway Interface) and Java Servlets provide a means to layer access control middleware components without the need to modify the underlying application. This separation of concerns between access control functionality and application has further implications. The use of a given middleware interface specification constrains the range of properties upon which access may be determined to within the scope of the parameters of that interface, for HTTP: the URI, request method and so on. Adopting REST-based principles, URIs may be associated with resources to be protected and so a URI based access control policy can be realised. This has the advantage of performance – request content need not be parsed, only the request URI – and clarity: resources to be protected have a clear mapping to the URIs by which they are exposed. Not all services are easily amenable to this practice, however. For example, some operations for OGC [18] (Open Geospatial Consortium) web services require the use of the POST method. In such cases the access control middleware may need to consume the request message body so as to apply a given access policy.

A consequence of a URI based access policy is that the granularity of the URI scheme must match the granularity of access control policy required. In practice this has meant some careful consideration of the ESGF URI schemes for protected applications and data. This whole philosophy differs in approach to security application frameworks that embed access control functionality in the application code itself. Whilst they provide flexibility and fine-grained control over access, they break the separation between application code and access control functionality. In general, they cannot be deployed in environments where service stacks are maintained and developed independently of the security framework.

A filter-based architecture also enables the assembly of independent middleware components into a pipeline or chain since they all adhere to a common interface. This characteristic can be exploited to divide up access control functionality. For example, HTTP response codes can be used to separate the more generic function of flagging an unauthenticated request – by setting a HTTP 401 Unauthorized code – to the more application specific function of enforcing some associated response (e.g. displaying a sign in user interface).

4.1 Filter Chain for ESGF Services

Filters are defined to perform specific authentication and authorization related functions and follow a specific order. This is illustrated in figure 1:

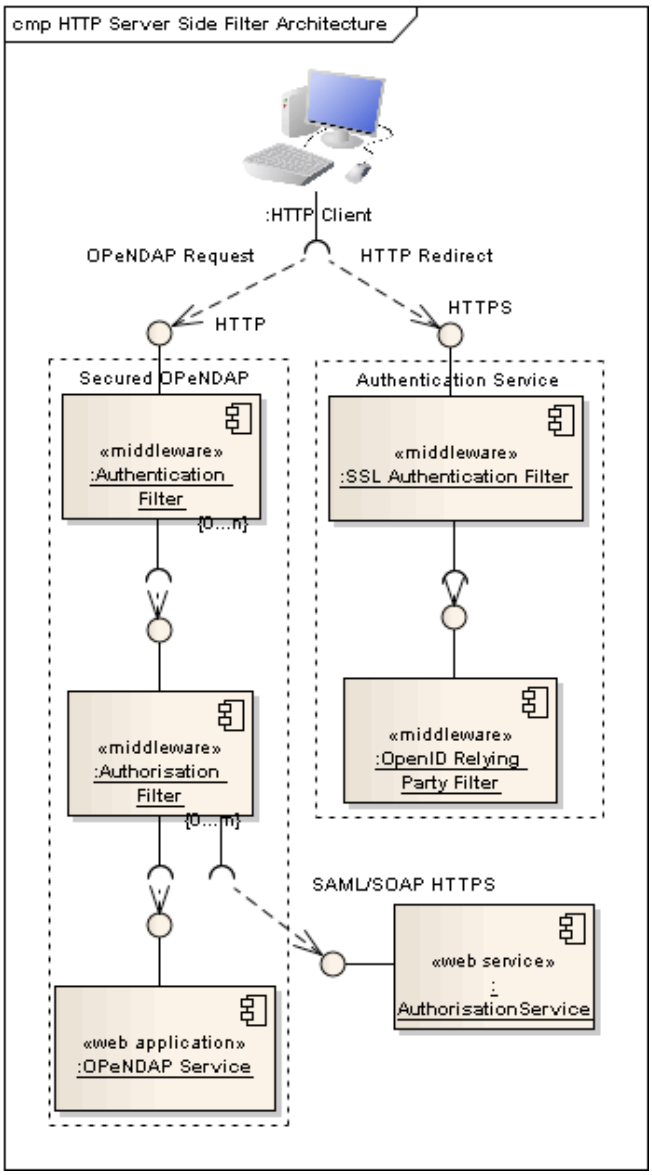


Figure 1: HTTP Server Side Filter Chain

Two filter chains are shown. The first fronts the application to be protected; the second one shown alongside it, deals specifically with the authentication process. The request from a client goes to the data serving application to be accessed, in this case an OPeNDAP service. An authentication enforcement filter is first to intercept the request. This checks the access restrictions for requested resource by consulting the policy. If no restriction is in place, control is passed on to the underlying application to serve the request. If a secured resource has been requested, the filter checks for the presence of a valid session cookie. In the absence of this, the client is returned a HTTP 30x response requesting redirection to an authentication service endpoint (shown on the right in figure 1), which listens over HTTPS. This dual HTTP/HTTPS arrangement allows authentication to be executed over a secure channel whilst at the same time avoiding the performance penalty associated with large data transfers over an encrypted connection.

The authentication service itself uses a twofold chain to enable a client to authenticate with either PKI-based

credentials or via OpenID. Notably, with this arrangement, the server side is agnostic to the client request method employed. The first filter checks for a user X.509 certificate obtained from the SSL handshake. If present, authentication proceeds based on verification of this identity: otherwise control passes to the next filter, which initiates an OpenID Relying Party interface. The default behaviour then is to assume OpenID-based sign in from a browser but note that the response code will be HTTP 401 Unauthorized to signal to non-browser-based clients that authentication credentials are required.

Whatever authentication method is used, a positive result will trigger a HTTP 30x redirect response to return the client back to the HTTP-based authentication filter. A signed authentication cookie is returned with this in the HTTP header. The recipient must be within the same cookie domain so that the returned cookie is visible to the authentication filter fronting the data serving application. On receipt of the cookie, this filter verifies it, sets the users authenticated status and passes control on to the next filter. The sequence in figure 2 illustrates the steps.

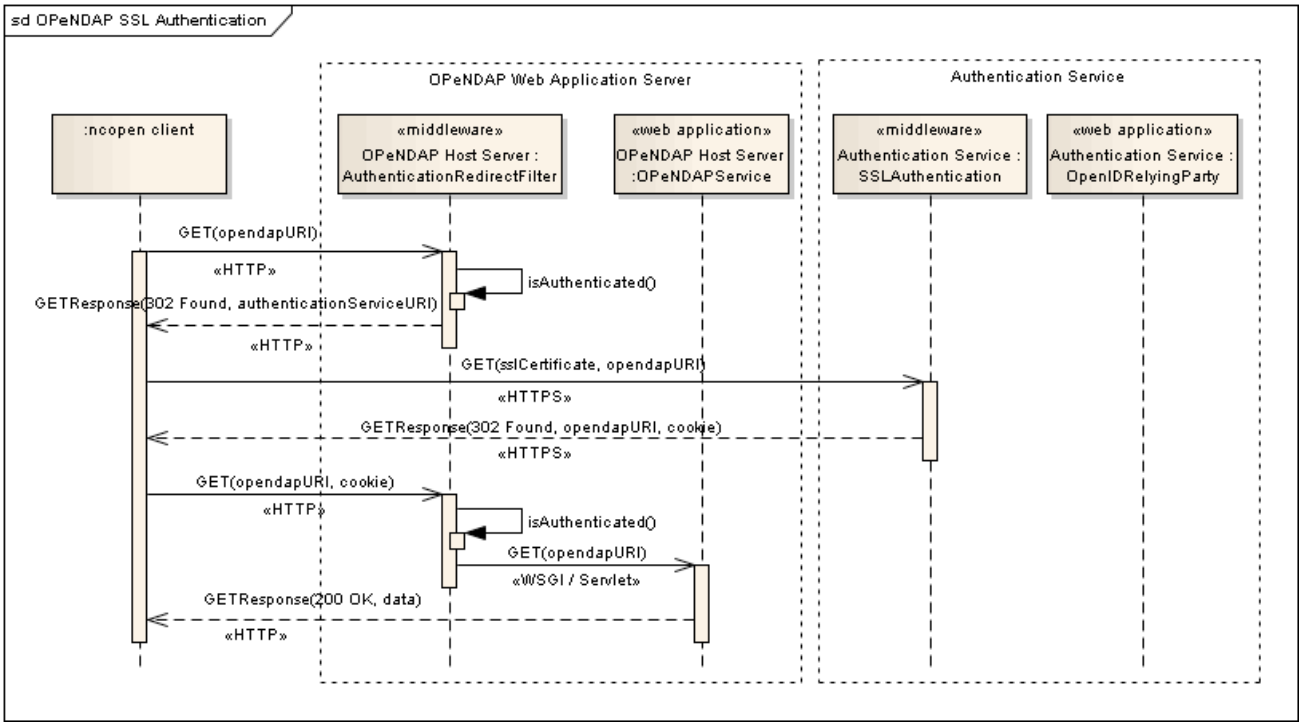


Figure 2: SSL Client Based Authentication with security filters (authorisation filters omitted for this illustration)

After the authentication filter, the request is intercepted by one or more authorisation filters. Typically, a chain will contain at least one SAML-based authorisation filter that is responsible for issuing requests to the external authorisation service. This may also enforce the authorisation decisions it receives or else delegate this to a separate, dedicated enforcement filter.

4.1.1 Python and Java Implementations

A Python implementation, developed at the BADC for the NERC (Natural Environment Research Council) DataGrid[14], was used to pilot many of the features of the filter-based architecture used for ESGF. A parallel Java implementation has also been written, which can be used to secure a generic web application that runs within a Java servlet container. This implementation is deployed widely at the ESGF Data Node sites.

5 Securing OPeNDAP Based Services

OPeNDAP is a data access framework widely used in the fields of oceanography and atmospheric science research, and was a key service to be supported by the ESGF security architecture. Data is served over a network interface, which abstracts the underlying data format from the client, and provides sub-setting functionality. The default Data Node configuration currently uses the THREDDS Data Server[19] (TDS) implementation of the OPeNDAP protocol. With the server side filter-based architecture as described in the previous section, it has been possible to configure both TDS and PyDAP[20] based OPeNDAP server implementations to support dual OpenID and SSL client based authentication mechanisms.

5.1 Extensions to NetCDF for ESGF Security-Aware Clients

Whilst the ability to apply this flexible approach to the server-side security layer is important, the development of compatible client software is vital to the adoption of these services across a wide user base. The redirect-based pattern with PKI-based credentials makes this solution suitable for simple HTTP clients. Wget, a utility available on most UNIX-based systems, can also easily be configured in this way. Clearly though, for this solution to have significant adoption, the relevant changes would need to be integrated into OPeNDAP client libraries. The software libraries for NetCDF were an obvious starting point. NetCDF is the standard format chosen for CMIP5 data and these are

widely used as the basis for client tools in the climate science community. By inserting changes at the NetCDF level in the software stack, all these dependencies would collectively benefit.

Working with Unidata, the maker of the NetCDF software, the C NetCDF library was adapted to enable custom SSL client settings. These were applied at the level of the user's `.dodsrc` file so that no changes to the C API were necessary. Thus existing software that builds on the NetCDF libraries requires no change to source code to support ESGF-based security, besides relinking with the latest version of the libraries. The security extensions are included in the NetCDF 4.1.2 release. This has been built with a number of different applications including Ferret[21], and NetCDF Python bindings[22]. Work is also underway to add support to the Java NetCDF client libraries and extensions to the PyDAP client libraries have enabled PyDAP-based packages like CDX[23] to access ESGF hosted data. By instrumenting both NetCDF C/Java and PyDAP libraries, we are instantly enabling a large portion of the current earth science analysis toolkits with an access control layer.

6 Secured Data Access Walkthrough

In this section we present a walkthrough of a typical use case to illustrate how the individual components in the security architecture interact.

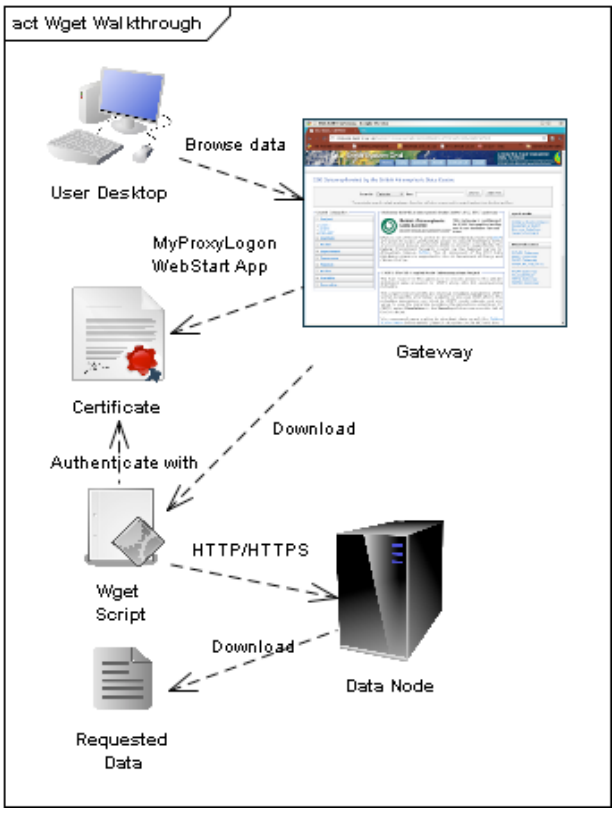


Figure 3: Secured Wget based Data Download

A user browses for CMIP5 data via the search facility of a Gateway Node and discovers data hosted at the BADC's Data Node. Individual datasets may be downloaded directly via the browser using OpenID. Alternatively for multiple downloads, an option is provided to generate a data download script for the user to download and execute. This uses the Wget program to perform the HTTP based retrievals. To download secured datasets, the Gateway provides a Java MyProxyLogon[9] WebStart program to enable users to obtain PKI credentials from MyProxy. The credentials are saved to a standard location on the user's file system visible to the Wget script.

When the script is called, the various datasets are retrieved from the Data Nodes specified in the script download URLs. For any given request, the security filters fronting the data serving application authenticate the request based on the PKI credentials provided and check for authorisation by calling the respective authorisation services. For CMIP5 data, a given authorisation service will check user entitlement with the corresponding authority for CMIP5 attribute registration: the PCMDI attribute service. If the user is registered, access is granted.

7 Future Work and Related Developments

Although the initial deployment of the Earth System Grid Federation has been in the context of supporting CMIP5, many other applications are expected to be deployed with the same infrastructure. Within both Europe and the U.S. there are major collaborative projects being built around ESGF. Significantly, enabling PKI-based authentication, opens up OPeNDAP based services to the Grid based security paradigm and in particular user delegation using proxy certificates[24]. A short NERC-funded proof-of-concept project MashMyData is exploring how OPeNDAP services and an OGC Web Processing Service can be coupled together in a workflow leveraging the ESGF security infrastructure with support for proxy certificates.

8 Conclusions

Modular design principles applied on a number of levels through the security architecture have resulted in a highly flexible solution applicable to the target domain whilst at the same time minimising the impact on the underlying APIs of existing services and tools.

The extensive use of existing standards in the service-oriented architecture has facilitated interoperability, with Python and Java implementations of services freely interchangeable. The filter-based HTTP server side architecture has enabled the same access control solution to be applied over a range of applications. It has also made possible a flexible approach to access control configuration where any given application may be fronted with multiple authentication and authorisation schemes. This is demonstrated by dual OpenID- and PKI-based authentication support. The latter, by exploiting characteristics inherent in HTTP/HTTPS, has minimised the entry point for client side tools to support it. This has meant that the user community can turn to simple freely available tools such as Wget to access secured data within

ESGF. Moreover, by applying security extensions to NetCDF, a software library used widely across the Earth science community, all the dependent software packages and tools built on it are enabled with the security support.

9 Acknowledgements

We acknowledge the contributions of the all the software development teams involved – for their hard work and support in the realisation of this architecture into a full implementation and deployment in an operational federation, amongst these: Stephen Pascoe (BADC), Neill Miller (ANL), Estanislao Gonzalez (MPIM, Hamburg), the PCMDI development team including Gavin Bell (PCMDI), Bob Drach (PCMDI) and Charles Doutriaux (PCMDI); Nathan Wilhelmi (NCAR), Eric Nienhouse (NCAR) and the development team at NCAR, Boulder, Colorado; Roland Schweitzer (NOAA/OAR). Thanks also to Dean Williams and the PIs from the various contributing projects.

The author also acknowledges the Software Sustainability Institute (UK) and NERC (UK) for their support with the NDG Security precursor work. Work on the ESGF security system was supported in part by the U.S. Dept. of Energy under Contract DE-AC02-06CH11357. Also supported in part by the Jet Propulsion Laboratory, managed by the California Institute of Technology, under a contract with NASA.

10 References

[1] Williams D. N., Ananthakrishnan R., Bernholdt D. E., Bharathi S., Brown D., Chen M., Chervenak A. L., Cinquini L., Drach R., Foster I. T., Fox P., Fraser D., Garcia J., Hankin S., Jones P., Middleton D. E., Schwidder, J., Schweitzer R., Schuler R., Shoshani A., Siebenlist F., Sim A., Strand W. G., Su M., Wilhelmi N., "The Earth System Grid: Enabling Access to Multi-Model Climate Simulation Data", Bulletin of the American Meteorological Society, February 2009.

[2] Williams D. N., Middleton D. E., Lautenschlager M., Lawrence B. N., "Delivering Globally Accessible Petascale Data for CMIP5", IEEE Software, April 2011.

[3] NetCDF (Network Common Data Form), http://www.unidata.ucar.edu/software/netcdf/

[4] OPeNDAP (Open-source Project for a Network Data Access Protocol), http://www.opendap.org

[5] GridFTP, http://dev.globus.org/wiki/GridFTP

[6] OpenID, http://openid.net

[7] Siebenlist F., Ananthakrishnan R., Bernholdt D. E., Cinquini L., Foster I. T., Middleton D. E., Miller N., Williams D. N., "Earth System Grid Authentication Infrastructure: Integrating Local Authentication, OpenID and PKI", TeraGrid 2009, June 2009

[8] OASIS Security Services (SAML) TC, http://www.oasisopen.org/committees/tc_home.php?wg_abbrev=security

[9] MyProxy, http://grid.ncsa.uiuc.edu/myproxy/

[10] Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, http://tools.ietf.org/html/rfc5280

[11] Barton T., Basney J., Freeman T., Scavo T., Siebenlist F., Welch V., Ananthakrishnan R., Baker B., Goode M., Keahey K., "Identity Federation and Attribute-based Authorization through the Globus Toolkit, Shibboleth, GridShib, and MyProxy", 5th Annual PKI R&D Workshop, 2006

[12] Sinnott R. O., Chadwick, D.W., Koetsier J., Otenko O., Watt J. and Nguyen, T.A. (2006) "Supporting Decentralized, Security Focused Dynamic Virtual Organizations Across the Grid", Proceedings of the Second IEEE International Conference on e-Science and Grid Computing 2006 (e-Science '06), December 2006, Amsterdam, The Netherlands.

[13] Yadis, http://yadis.org

[14] Kershaw, P., Blower, J. and Lawrence, B., "Practical Access Control Using NDG Security", e-Science All Hands Meeting, September 2007

[15] Fielding R. T., "Representational State Transfer (REST), Architectural Styles and the Design of Network-based Software Architectures", http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

[16] Kiczales G., Lamping J., Mendhekar A., Maeda C., Videira Lopes C., Loingtier J.-M., and Irwin, J, "Aspect-Oriented Programming", Proceedings of ECOOP 1997.

[17] Web Server Gateway Interface, http://wsgi.org/wsgi

[18] Open Geospatial Consortium, http://www.opengeospatial.org/

[19] THREDDS Data Server, http://www.unidata.ucar.edu/projects/THREDDS/tech/TDS.html

[20] PyDAP, http://pydap.org

[21] Ferret, http://ferret.wrc.noaa.gov/

[22] Python NetCDF, http://code.google.com/p/netcdf4-python/

[23] CDX (Climate Data eXchange), http://cdx.jpl.nasa.gov/

[24] Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile, http://www.ietf.org/rfc/rfc3820.txt

Recent improvements in HLRmon, an accounting portal suitable for national Grids

A. Cristofori[1], E. Fattibene[1], and L. Gaido[2]
[1]Istituto Nazionale Fisica Nucleare (INFN) CNAF, Bologna, Italy
[2]Istituto Nazionale Fisica Nucleare (INFN) Sezione di Torino, Torino, Italy

Abstract - *In a production computing Grid, the accounting infrastructure is responsible for the collection of usage records, both for CPU and storage resources. Accounting data have to be properly managed and presented through a graphic interface in order to be useful to different user categories. HLRmon is the web interface for the Distributed Grid Accounting Service (DGAS) and it is used as the accounting portal by National Grid Initiatives (NGI). It shows accounting information organized according to various levels of aggregation. In this paper we describe the new HLRmon functionalities. In particular we will focus on the available aggregation fields in the CPU accounting views. In addition the new section on storage utilization will be described.*

Keywords: Grid - accounting - data aggregation - web portal - storage

1 Introduction

More and more scientific communities have recently shown a growing interest in Grid computing infrastructures that were built to address the huge computational needs of high energy physics, such as those of the Large Hadron Collider (LHC) experiments at Conseil Européen pour la Recherche Nucléaire (CERN) in Geneva. The use of computational Grids is expanding to other scientific disciplines such as biology, astronomy, earth sciences, life sciences, etc. Those systems allow the sharing of computational resources, data, instruments and storage space among researchers belonging to different Institutes and scientific fields spread over a wide geographic area.

The effective utilization of Grid technologies relies also on the availability of systems for the accounting of the consumed resources. The huge amount of data produced by the accounting infrastructures requires tools to ease the visualization and reporting of that information. The HLRmon accounting portal has been developed to achieve this goal. It offers graphic and tabular views on the accounting data by aggregating them on different configurable ways. HLRmon relies on the Distributed Grid Accounting Service (DGAS) [1][2][3], a Grid middleware integrated and interoperating with the gLite Grid middleware distribution [4].

This paper presents the recent evolutions of HLRmon. The CPU accounting views have been improved by offering the possibility to aggregate reports on the basis of different fields such as Grid site, Virtual Organization (VO) and subgroups within a VO, Certification Authority (CA) and Institute of origin of the Grid users submitting the jobs. Moreover, a new web section reporting the storage space utilization is being developed.

This paper is structured as follows. Section 2 gives an overview of the HLRmon tool, while Section 3 describes the new CPU accounting aggregation fields. Section 4 is dedicated to the new storage accounting view. Section 5 concludes the paper.

2 HLRmon overview

HLRmon [5] has been conceived to provide high level interface to accounting data for different kind of users such as Grid managers, site administrators, VO managers and Grid end users. Due to the complexity of the accounting information and to the different aggregation levels that users may need, HLRmon is designed to aggregate Usage Records (UR) in various different ways.

The main source of data for HLRmon is the Home Location Register (HLR), the DGAS data repository. Accounting information is retrieved from the HLR on a daily basis, aggregated and stored into a local *MySQL* database. By querying the local database, data are handled to produce a number of charts highly customizable by the users and organized in different web pages. Moreover, HLRmon can provide a table showing information about computing resource usage aggregated by Grid user and retrieved from the HLR database on-demand.

Access to the portal is possible to all users with a valid personal *x.509* certificate released by a Grid accredited CA. For privacy reasons the access to the section with detailed information aggregated by Grid user is restricted to people registered to the portal with the appropriate Grid roles.

HLRmon has been adopted as accounting portal for some national production Grids and projects. Three instances are

hosted at INFN for the Italian Grid Infrastructure[1] (IGI) and for the We-NMR[2] and the Cybersar[3] projects. Other two instances are hosted at Foundation for Research and Technology – Hellas (FORTH) in Greece for Hellasgrid[4] and in Germany for the D-Grid[5] infrastructure.

3 CPU accounting aggregation fields

In order to give users the freedom to choose the relevant subset of information, HLRmon aggregates the URs extracted from the HLR and stores the result of this process in the local database. The aggregation is done according to the following relevant fields:

- Site: the administrative site hosting the resources used by the job.

- VO: the group of people typically belonging to the same scientific experiment or to a Grid administrative group.

- VOMS (VO Management Service) group. VOMS [6] is the service responsible for the user authorization on the Grid and provides information about the user's relationship with his/her VO. Each VO member may be part of several VOMS groups defined within a VO and may also have a specific set of roles for each group. This information is taken from the user VOMS Fully Qualified Attribute Name (FQAN), a string containing the VOMS attributes.

- VOMS role. Each VOMS role specifies the capabilities associated to a VO member. Roles can be granted to perform specific tasks within a VO. This information is collected from the VOMS FQAN.

- CA: the authority that released the personal certificate used by the VO member that submitted the job to authenticate himself/herself to the Grid. This information is contained in the Distinguished Name (DN) of the personal certificate that univocally identifies the Grid user. There is not a standard agreement for the CA name format within the DN, so different formats are used. However, whenever the CA name is recognizable in one of the DN fields HLRmon is easily configurable to get such information and aggregate data accordingly.

- CA subgroup (if applicable): information about the CA structure, e.g. the unit that performed the user authentication. Again HLRmon can be configured to extract a specific set of attributes from the user certificate's DN and aggregate the data accordingly. The attributes can be determined by means of a regular expression that parses the DN string and extracts the relevant information. As an example, this feature is useful when the DN contains the information about the Registration Authority (RA) that identified the user since this may provide information about the Institute or the Institute's unit the user belongs to.

The web interface presents accounting reports in graphical or tabular mode. A selection menu allows to choose the desired subset of values for each of the above mentioned aggregation fields. The VOMS group and role lists in the selection menu are dynamically generated according to the chosen VOs. In the same way, the CA subgroup list is dynamically generated according to the selected CAs. Moreover users can specify the time interval and the metrics that are of interest to them. The metrics that can be selected are the following:

- Number of jobs

- CPU time expressed in days and CPU time normalized by the mean power capacity of the site

- Wall clock time and normalized Wall clock time

- Job efficiency - ratio between CPU time and Wall clock time

According to the selections, HLRmon produces a report about the resource usage in terms of the selected metric expressed through the following aggregations:

- per day

- per site and per site per day

- per VO and per VO per day

- per VOMS group/role and per VOMS group/role per day

- per CA and per CA per day

- per CA subgroup and per CA subgroup per day

- per job type (Grid or local)

[1] HLRmon for IGI: https://dgas.cnaf.infn.it/hlrmon/report/
[2] HLRmon for We-NMR: https://hlrmon-01.pd.infn.it:50080/hlrmon/report/
[3] HLRmon for Cybersar: https://hlrmon-cyb.ca.infn.it/hlrmon/report/
[4] HLRmon for Hellasgrid: https://hlrmon.hellasgrid.gr/hlrmon/report/
[5] HLRmon for D-Grid: https://hlrmon.d-grid.uni-hannover.de/hlrmon/

3.1 Relevance for the Italian NGI (IGI)

The European Grid Infrastructure (EGI) [7] is composed by national computing Grids, known as National Grid Initiatives (NGI), providing a unique representation at the European and international level for all the communities related to national Grid infrastructures: from resources providers to scientific users. At the time of writing the EGI infrastructure includes 30 NGIs registered into the Grid Operations Centre Data Base (GOCDB) [8][9], the official repository for storing and presenting EGI topology and resources information. Each NGI is composed by different research Institutes and Universities.

IGI [10] is the Italian NGI and is composed by 12 partners. Among them, INFN has its own CA that issues personal certificates to people working for all the IGI partner Institutes and Universities. Each partner has at least one RA. When more then one RA is present for a partner, each RA usually corresponds to a unit within the same Institute. The DNs of the personal certificates issued by the INFN CA [11] contain the "L" field, representing the RA. The IGI HLRmon server is configured to aggregate data per RA and thus provides information about the amount of resource used by the members of different Institutes as well as of the different units within the same Institute. Furthermore, the HLRmon web interface offers the possibility to group two or more RAs in a new entity in order to visualize data related to the selected RAs as a unique group. E.g.: if an Institute has more than one RA, the user can select all the RAs belonging to that Institute and merge them in a new group; in this way it is possible to create different groups, each representing an IGI partner Institute.

Other useful aggregation levels provided by HLRmon are those based on the VOMS groups and VOMS roles. Users can compare the activity of groups within a single VO or over a set of chosen VOs. This can be useful for VOs that aggregate their members in different groups and assign specific roles to them. As an example, catch-all VOs, can get resource usage reports according to the community or experiment the users belong to.

4 Storage accounting view

Accounting systems deployed within EGI are considered sufficiently mature and stable in a production environment for the CPU accounting task. On the other hand, storage accounting activity is not as advanced if compared to CPU accounting. Recently, however, there has been a growing interest on this field and the need for the development of a reliable storage accounting service got across. For this reason the DGAS team is developing a storage accounting service and HLRmon is evolving towards an interface that can produce storage reports too.

4.1 Storage accounting view for WLCG

The HLRmon portal includes some specific views customized according to the needs of the Worldwide LHC Computing Grid (WLCG) [12]; the most important are the comparison between the used and pledged computing resources (per site and per VO) and some very basic information about storage usage. This section reports only sites and VOs of interest for WLCG (ALICE, ATLAS, CMS and LHCb).

Storage accounting data are collected through a simple script running on the SRM instances managed at each given site once a day. The script includes:

1. a site-specific part that gathers the needed storage information from the SRM node; it is customized for various SRM implementations such as StoRM, DPM and dCache. Typically this script is tailored to the local storage configuration by the site administrators.

2. a general part that gathers disk utilization information, fills a XML document and sends it to the HLRmon server. Currently this is done via email.

Storage accounting data for WLCG resources are shown through a graphic view. For each site, the first plot shows the total and free disk space installed at the site compared with the fraction used by each LHC VO as a function of time, while other charts show the total disk space assigned to each LHC VO, the disk space used and also the disk space used by each Storage Area (SA, a logical portion of storage assigned to a VO).

4.2 The new storage data acquisition process

The need for a unique and reliable model for both CPU and storage accounting data workflow required a change in the system used to handle and publish storage accounting information. The DGAS accounting system is evolving to manage storage accounting information in the same way as for the CPU. A standard UR for the storage accounting has not been defined yet. As a consequence, non standard UR schema has been used in the current DGAS implementation; it might be a contribution to the standard definition within the appropriate Open Grid Forum (OGF) working group. The storage URs follow the same path as the CPU ones and this allows the use of the same HLR repository for both CPU and storage records. HLRmon has been further improved in order to retrieve the storage data from the HLR. This is a remarkable improvement since storage accounting data retrieval may benefit of the DGAS infrastructure already deployed in the production Grid. Moreover, HLRmon is now able to retrieve storage accounting information for all the sites that send their data to the queried HLR repository.

4.3 Storage data aggregation

As for CPU accounting data, the HLRmon server daily retrieves storage information from a HLR repository. This is accomplished by means of a new software component that has been developed; it also appropriately elaborates the data received from the HLR, aggregates them and stores the result into the HLRmon database.

The current version of the storage UR schema contains a field that defines the time duration associated with the resource consumption and calculated as the difference between the current and the previous measurement. By means of this field, HLRmon calculates the daily disk consumption in terms of space-per-hour as the integral of the used space over time. The retrieved information has the granularity of the SA. The UR stored into the HLR database does not provide any information on the pledged storage space, since this is simply a static administrative value. In order to compare the storage space utilization with the pledged one, HLRmon queries the Grid Information System (IS) for the specific attribute defined within the GLUE schema [13].

HLRmon is therefore ready to display disk usage reports through a web view as performed for the CPU accounting. The definition of the charts types and the aggregation fields available in the selection menu is still under study at the time of writing.

5 Conclusions

This paper presents the new features of HLRmon, a Grid accounting portal based on DGAS. The variety of web views, its maturity and stability make HLRmon a good solution, together with DGAS, for a comprehensive Grid accounting system. This is proved by its adoption in some NGIs and Grid projects. The high level of the CPU accounting views customization has been further enhanced by adding the possibility to choose among new aggregation fields. Those functionalities allow users to evaluate the resource consumption by different VOMS groups and roles for those VOs that divide members in categories. Moreover, it is possible to aggregate accounting data per CA or other information contained into the DN of the Grid user's personal certificate. E.g.: the IGI HLRmon server is configured to aggregate data per RA, which means being able to distinguish over different partner Institutes and units of the same Institute.

The recent developments include also improvements on the storage accounting view. The storage UR distribution has been revised allowing HLRmon to retrieve accounting data directly from DGAS repositories. In the future we will focus our work on the definition of the aggregation and visualization schema for the storage accounting views.

Acknowledgements

HLRmon and DGAS are supported by the INFN Grid project. DGAS was also supported by the EU projects EGEE, EGEE-II and EMI, sponsored by the European Union under contract number INFSO 508833, INFSO-RI-031688 and RI-261611 respectively.

6 References

[1] Guarise A., Piro R.M., (2009), "Distributed Grid Accounting System, Market Oriented Grid and Utility Computing", R. Buyya and K. Bubendorfer (eds), ISBN: 978-0470287682, Wiley Press, Hoboken, New Jersey, USA.

[2] Piro R.M., Guarise A., Patania G., Werbrouck A., (2008), "Using historical accounting information to predict resource usage of grid jobs", Future Generation Computing System, Volume 25, Issue 5, May 2009, pp. 499-510, doi:10.1016/j.future.2008.11.003.

[3] Distributed Grid Accounting System home page (http://www.to.infn.it/dgas/).

[4] Laure E., Fisher S.M., Frohner A., Grandi C., Kunszt P., Krenek A., Mulmo O., Pacini F., Prelz F., White J. et al., (2006), "Programming the Grid with gLite", Computational methods in science and technology.

[5] Fattibene E., Ferrari T., Gaido L., Misurelli G., Solagna P., "HLRmon, a Tool for Advanced Visualization of Resource Utilization", Data Driven E-science: Use Cases and Successful Applications of Distributed Computing Infrastructures (ISGC 2010).

[6] Alfieri R., Cecchini R., Ciaschini V., Dell'Agnello L. et al., (2004), "VOMS, an authorization system for virtual organizations", Grid Computing.

[7] EGI home page (http://www.egi.eu/).

[8] Mathieu G., Casson J., "GOCDB4, a New Architecture for the European Grid Infrastructure", Data Driven E-science: Use Cases and Successful Applications of Distributed Computing Infrastructures (ISGC 2010).

[9] GOCDB home page (https://goc.gridops.org/).

[10] IGI home page (http://www.italiangrid.org/).

[11] INFN CA home page (http://security.fi.infn.it/CA/).

[12] WLCG home page (http://lcg.web.cern.ch/lcg/).

[13] Andreozzi S.; Burke S.; Donno F. et al.; "GLUE Schema Specification, version 1.3", (OGF GLUE-WG 2007).

Design and Implementation of GUISET-Driven Authentication Framework

Ekabua, Obeten O.
Dept. of Comp. Sci. and Infor. Syst
University of Venda, Private Bag
X5050, Thohoyandou 0950
South Africa
obeten.ekabua@univen.ac.za

Adigun, Matthew O.
Dept. of Computer Science,
University of Zululand, Private Bag
X1001, KwaDlangezwa 3886
South Africa.
matthewo@pan.unizul.ac.za

Isong, Bassey E.
Dept. of Comp. Sci. and Infor. Syst.
University of Venda, Private Bag
X5050, Thohoyandou 0950
South Africa
bassey.isong@univen.ac.za

Abstract

Authentication is the process of identifying a user on the basis of the credentials provided. In reality, the user does not necessarily have to be a person; it can be an application that is making a remote call from the intranet or Internet. Different Grid domains provide different security policies and GUISET is not an exception. Authentication ensures that each application or user must be able to identify itself by providing the required credentials. With an exponential increase in the number of users and applications in GUISET infrastructure, the need for authentication becomes inevitable. More so, authentication ensures that resources are accessed only by users with appropriate credentials. Many authentication frameworks exist, but these are not applicable to our GUISET domain. Therefore as reported in this paper, we have developed and implemented a GUISET-driven authentication framework, as a security agent, that identifies a user based on the provision of appropriate credentials (username and password) for service requesters and providers in GUISET environment.

1. Introduction

The scalability of network security infrastructure is becoming a growing concern as the exponential growth of the internet continues unabated [1].The Internet today has become an insecure place and many of the protocols used do not provide any security. Also, many tools exist, for instance, tools to "sniff" passwords off the network are in common use by malicious hackers today. Thus, applications which send an unencrypted password over the network are extremely vulnerable [2]. As the number of new users and applications continues to grow exponentially, so has the number of users and applications who require authentication.

Authentication is the process of identifying a user base on the credentials provided. In reality, the user does not necessarily have to be a person; it can be an application that is making a remote call from the intranet or Internet. This application must be able to identify itself by providing the required credentials. These credentials can be in the form of a username and password, also known as Username Token, a digital certificate, such as an X.509 certificate or a Kerberos token [3]. If the credentials are valid, the entity (user or application) that submits the credentials is considered an authenticated entity. Once an entity has been authenticated, the authorization process would then determine whether that particular entity has access to a given GUISET resource or not. However, due to continual existence of open standards, protocols and interfaces, any organization or individual can become a Web Service provider [4]. This lack of standardization thus promotes heterogeneous and independent web services which employ arbitrary authentication and access control methods thus prompting many challenges for service composition. So what are the implications of having incompatible data and access control models in heterogeneous web services? The implication is users are required to provide different authentication credentials each and every single time that they want to access a resource that is provided by a different service provider. This is not only frustrating for a user but it can also lead to users choosing passwords that are seen to have low security strength and thus leaving a system vulnerable to security breach [5].

Grid computing has emerged as one of the most fascinating research fields with the ability to share resources and services distributed across multiple institutions called virtual organizations (VO). Members in VO by using state-of-the-art cyber-infrastructure services can work coherently as a whole to conduct scientific research and industrial design, or solve business problems. Furthermore, each VO shares a pool of service-enabled resources and communication-enabled tools that may include software applications, hardware, data collections, computational power, storage and networks [4]. Individual pieces of hardware may be used in more than one grid, people may be members of more than one VO and the different resources in a grid may have different access policies, including how they authenticate and authorize users. Nevertheless, these types of environments will invariably pose many complex requirements for access control and authentication between separate entities. This is also applicable to Grid-based Utility Infrastructure for SMME Enabling

Technology (GUISET) research infrastructure. GUISET proposed by Adigun et al [6] is an infrastructure that is used to solve problems experienced by Small Micro and Medium Enterprises (SMMEs) in Africa. It allows SMMEs to share information that helps them market and sell their products online with less technology expending.

However, due too the complex nature of Grid computing environment, existing security mechanisms in the GUISET infrastructure lacks the necessary capacity to fully protect its users and applications from malicious intruders. This has thus prompted the need to conduct research into finding an appropriate security solution for GUISET. Existing Authentication frameworks are not entirely fitted for our GUISET domain, as frameworks in most cases are domain dependent. We have identified core emerging security issues need to be considered with respect to this security dilemma facing the GUISET infrastructure. These issues include, but are not limited to, Authentication, Authorization, Access control and Trust.

In this paper, an authentication framework for the GUISET infrastructure is developed, which allows users or applications to use a single set of credentials to authenticate into disparate web services. This is achieved by developing a framework that facilitates authentication between disparate web services using different authentication methods or protocols and developing a single-sign-on (SSO) architecture for the framework. The work would be limited to and only the authentication aspect of security in the GUISET infrastructure.

2. Grid Services and Security

First and foremost, it would be wise to define what service and grid are before delving deeper into this section. A service can be defined as an act of help or assistance [7] while grid is a collection of distributed (networked) computers that pool their resources together in such a way that users may utilize processing, storage, software and data resources from any of the interconnected computers, leading to greater resource sharing and higher utilization ratio [8].

The plight to integrate security protocols of heterogeneous web services has not received its due emphasis from the academic society [4]. Therefore issues such as how to integrate security protocols with modern highly efficient group communications systems and the issues that arises have been left neglected [9]. A secure group communication system needs to provide confidentiality and integrity of client data, integrity and possibly confidentiality of server, control data, client authentication, message source authentication and access control of system resources and services [4]. The architecture developed as part of this research is flexible, allowing many different protocols to be supported and even be executing at the same time; it is modular so that security protocols can be implemented and maintained independently of the network and distributed protocols that make up the group messaging system.

With the proliferation of open standards, protocols and interfaces, any organization or individual can become a web service provider. The heterogeneous and independent nature of web services prompts too many challenges for service composition. Beyond problems in match making are challenges in bridging incompatible data models and access control methods [4] typically in Grid or VOs [6]. Irrespective of the advances in access control approaches applicable to the grid computing environment, there still remains issues that impede the development of effective access control for grid applications; security and trust management. Security is a critical issue in the tremendous benefit of the wide use of internet applications, as security failure can expose confidential data, etc. Client authentication has been used as an alternative to protect services and information, but this has its own weakness as an attack can break this protection and the services could be accessed without authorization. These pose a great challenging task for ensuring secure and authorized access to remote services and information resources in a dynamic collaborative environment [7].

3. The GUISET Infrastructure

The concept of GUISET is based on the idea that technology that is affordable is developed for SMME. This can be achieved by finding appropriate strategy to make affordable technologies available using the utility approach to service delivery. The GUISET infrastructure as earlier discussed plays vital roles of allowing effective information sharing and marketing/selling aids for SMME with less spending on technology [6]. GUISET infrastructure research focus was decomposed into three objective phrases, namely: pay-as-you-go, quality of service constrained, and utility computing services. This signifies that we envisage a future in which service providers will competitively provide computing services at a variable compared to currently fixed cost to clients based on their quality of service requirement [10]. The opportunities provided by this project include are tremendous:

- Making technology diffusion have more direct impact on the community that is being assisted.
- Uplifting the community to the status of becoming part of the Mobile business value network, and
- Leveraging the economic benefits of mobile technology such as access to market information, new business opportunities and staying within the reach of

customers and business contacts.

There is an increasing recognition of the important role SMMEs play in creating job and promoting economic growth. Given the accelerating pace of globalization and liberalization, traditional boundaries no longer separate the vendor from the purchaser and competition amongst business entities has intensified. This makes it difficult for SMMEs to compete successfully in the global market places. The adoption of enterprise-wide applications for devising an enterprise-wide IT strategy has not yet occurred for SMMEs around the world. SMMEs in South Africa are not left out of this trend. To remain in business, SMMEs must be able to identify the Information and Communication Technology (ICT) infrastructures required to improve performance and global competitiveness. But SMMEs lack the capacity to own ICT infrastructure and they are more conservative to adopt ICT infrastructures before they can be sure of the return on such investments [11]. SMMEs in the developing world face more competitive challenges in the global economy. While it is true that globalization creates opportunities for SMMEs to be effectively involved in global markets, it is doubtful if African SMMEs can take full advantage of the potential opportunities and successfully compete in such a global environment. This is because they are beset with numerous problems such as, lack of sufficient and good quality information to inform decisions making on use of technology and market characteristics restricting deployment of some technologies [8].

4. Framework Design

In this section, the formal steps leading to the design of the GUISET authentication framework is described. These include diagrammatic illustrations, careful analysis of requirements, protocols and other various scenarios to name but a few. The authentication framework will be developed to become the cornerstone of GUISET security system; because it is an irrefutable fact that authentication forms the foundation of any security mechanism. The benefits attributed to this include management of multiple security credentials of all of their Subjects through Credential Manager Service (CMS) and the provision of a single-sign- architecture. In subsequent sections, all these will be discussed in details.

4.1 Requirements Specification

For ease of comprehension of the GUISET authentication framework design and functionalities, it is of the essence to review the requirements the design will meet. Such requirements through elicitation are outlined as follows:

Single Set of Credentials: A feature that allows a user to input only a single set of credentials for a single session, in-order to mitigate the problem of having the user provide a different set of credentials to every resource participating in the GUISET framework.

Credential Retrieval System: A feature that allows the infrastructure to retrieve user credentials from the various heterogeneous web services. Since it is unreasonable to ask the user to enter multiple credentials, or to have the end service change their authentication schemes to suit the user, this becomes very useful.

Credential Storage: it deals with storing long term and short term security tokens in a repository to be used by the security token service (STS) in the issuing of the security context token.

Revoke Credentials: The client may ask the STS to revoke the security context token in a case where a security token is compromised, which can be due to a security threat or hardware/software failure.

Issue: This operation involves the creation of a new security context token by the STS as requested by the client so as to establish a secure communication between the client and the target service.

Renew: An operation involving the renewal of a security context token by STS, in this case a token can only be renewed if the credentials inside the token are marked renewable for a requested period of time.

Exchange: This operation involves the exchange of a security token into a different type or format i.e. custom security token, username/password token for a different trust domain. For this operation to be performed there is a need for an additional service for translation purposes and used at the backend.

4.2. GUISET Secure Communication Protocol

We proposed protocol for authentication alongside its security mechanisms for performing authentication prior to granting access to a service in a GUISET environment in this section. The protocol provides a security token for a client wanting to access a resource in a Grid-based environment such as GUISET. Its underlying objective is that users can logon to the system once, receiving a token that both identifies and authenticates them [4, 12].

The protocol is based on the WS-Trust request and response pair from the service requestor to the STS. The security token requestor issues a Request Security Token (RST) request and sends it to the STS which verifies the validity of the request and then performs the operation as denoted by the security token requestor and then sends a Request Security Token Response (RSTR) back to the service requestor.

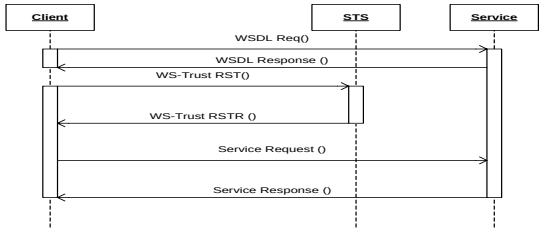


Fig 1.0. The GUISET Request/Response Messaging Protocol

Fig 1.0 shows UML interaction diagram that clearly depicts how the interaction occurs between the security token requestor and the STS using the issue forward-use mechanism.

4.3. Authentication Model

Going by the goal guiding this work which is the management of multiple security credentials held by Subjects, the issue is how and where they are to be stored, how they are to be retrieved according to the requirements of the Service, how they are to be delivered to the Service etc. Here a solution is propose where the Domain will have a Service to store, retrieve and maintain multiple security credentials of all of their Subjects via CMS.

CMS manages multiple Subjects' credentials provided by the Domain to which the Subject belongs. The Subjects store all their security credentials in CMS. To access a Service, the Subject first discovers the security requirements of that Service and checks the required security credentials in CMS. If the security credentials are present, it gets them from CMS by authenticating itself to CMS. If not present, the Subject may get them from relevant certificate authority or other third party. These credentials are then used by Subject to access the Service. After this, the credentials are stored in CMS also for future use.

Another hitch that affects authentication is single sign-on and delegation. Single sign-on and delegation features depend on the credentials used for authentication. E.g. these features cannot be provided if the Subject is using username/password for authentication, but, can be easily provided if the Subjects use X.509 certificates [5, 13]. In order to combat this problem, the GUISET environment will make use of the Security Assertion Mark-up Language (SAML) to provide single sign on and delegation features. Subjects will have to provide X.509 certificate to use these features. Subjects are issued certificates by certificate authorities. Certificates bind Subject's DN (Distinguished Name) to its public and private key. Public key is written in the certificate whereas private key remains with the Subject. For Subjects to prove their identity they must have a Certificate and a Private Key. Authentication process involves presenting the certificate and proving the possession of the private key. Private Key is held by Subjects and is not disclosed to anyone under any circumstances. Proxy certificates enable GUISET to implement non repudiation also. As the Subjects are digitally signing the certificates, they cannot deny them from having sent the access request, as signature is possible only through their private key. Delegation is also possible through proxy certificates.

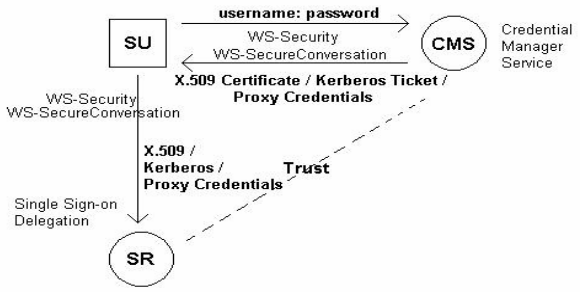


Fig 2.0 Credential Manager Service

The implementation of CMS will require GUISET to make use of WS-Security and related specifications implemented by WSE 3.0. Using these specifications, GUISET will be able to guarantee the confidentiality and integrity of the messages exchanged. Credential Manager is a feature rich and flexible service for storing, retrieving, updating and managing multiple user credentials. Users can store their multiple security credentials (username/password, X.509 certificates, Kerberos Tickets etc) in CMS. Access to CMS is either through X.509 Certificates or Username/password. Fig. 2.0 shows how CMS is access by the Subject.

4.4. GUISET Architecture and Authentication Framework

The general design of the GUISET architecture upon which this work is based is presented in this section. The architecture of GUISET alongside its relevant components is divided into 3 tiers which are:

- Multi-modal Interfaces Interface,
- Middleware Layer, and
- Grid Infrastructure Layer.

These tiers are only shown diagrammatically in Fig. 3.0 with their components without further details. However, to mitigate the problem of providing different set of credentials to every resource participating in the GUISET infrastructure, an authentication framework has been developed and embedded in the GUISET infrastructure to facilitate authentication in grid environment. This is shown in Fig. 4.0 - showing how the credential services interact with other components of the GUISET architecture.

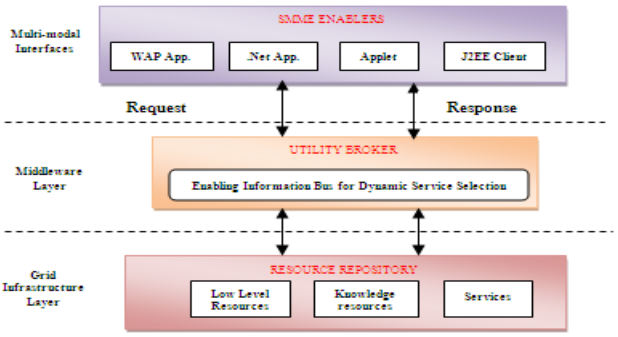


Fig. 3.0. GUISET architecture

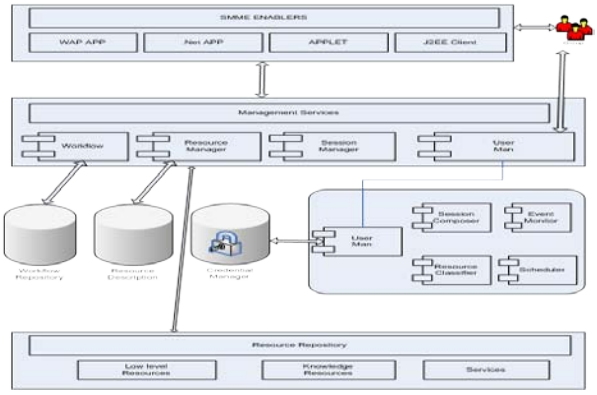


Fig. 4.0 Authentication frameworks embedded in GUISET

4.5. GUISET Authentication Scenarios

This section presents an authentication scenario using GUISET.

Scenario name: Request Authentication

Participating actors: Mr. Nkosi, Nongoma Arts & Craft and Makhaye HW c.c.

Flow of Events:

- Mr. Nkosi a new customer in the GUISET infrastructure request authentication
- Mr. Nkosi is prompted to enter his authentication credentials.
- Mr. Nkosi having been granted access to the GUISET infrastructure and its services searches for a product of his choice.
- Mr. Nkosi purchases his chosen item from the Nongoma Arts and Crafts group.

Scenario Name: Revoked X.509 certificate

Participating actors: Mr. Nkosi and Versign

Flow of Events:

- Mr. Nkosi request authentication into the GUISET infrastructure.
- Mr. Nkosi is then prompted to provide his authentication credentials.
- Mr Nkosi then provides the private key for his X.509 certificate which was issued by VeriSign to use to gain authentication.

Mr Nkosi certificate is however rejected as the certificate has expired and thus been revoked by the CA.

4.6. GUISET Environment Risk Management

During the development of any security infrastructure, dealing with the performance of security threats of various kinds that may hinder or harm the normal functioning of the proposed system is of the essence. The following are some of the threat profiles that may transpire in the GUISET environment:

Security Token/password no longer in possession of a genuine holder – An issued security token/password can sometimes be no longer in the possession of the original requester due to security threats such as 'packet sniffing' or 'man in the middle attack'. With this, it is difficult to determine whether the token/password is with the attacker or inactive - Denial of service attack. To prevent this from occurring, the services of the lost token should be revoked as soon as the unavailability of the token is discovered.

Security Token is damaged or unusable – Security tokens can become damaged due to hardware faults or through attackers' intervention, thereby causing tokens to take longer time to operate or even render some sensitive key material in the token unusable. In this case, it is vital to revoke the services of the damaged token so as to prevent potential abuse.

Tampering with data in the security token – An attacker can extract sensitive data stored in the security token resulting to cloned tokens which can then be used to impersonate the legitimate user of that token. One strategies that can be adopted in preventing this security threats from occurring is by enforcing mechanisms that will ensure that sensitive data is handled securely (WS-Secure Conversation) during a conversation between clients and services in such environments.

Unknown Certificate Authority - The basis of all X.509 based authentication is an anchor of trust that should be installed on each authenticating host. The authenticated system presents an X.509 certificate, or a proxy certificate chain, but the original X.509 certificate should be signed by a CA that is trusted by the authenticating system. In practice this means that all systems participating in the Grid have a designated directory containing all certificates of the CAs that should be trusted. On a UI this directory is usually stored in the environment variable X509_CERT_DIR. If this directory is missing, or not

containing all certificates of trusted CAs authentication will fail.

Proxy certificate not yet valid- Due to the nature of proxy certificates - used just after they are created, clock skew between authenticating systems might result in a proxy being rejected because the certificate is not yet deemed valid by the authenticating host. To allow for some clock skew a proxy certificate should by default created with a start time of five minutes in the past, but if the UI would be more than five minutes ahead, a failure condition can trigger.

5. Framework Implementation

The GUISET Authentication Framework was implemented by transforming the artifacts discussed in previous sections ranging from requirements analysis and design into an executable/workable system. For ease of comprehension, the GUISET Authentication Framework presented here will be discussed diagrammatically, showing interface relationships and displaying how the various components of GUISET interact.

5.1 GUISET Authentication

Authenticating a GUISET client according to the proposed framework involves series of steps. They include the following:

- The GUISET client accesses the GUISET portal
- The GUISET client is prompted to login by entering their authentication ID and passphrase.
- After providing their Login ID and passphrase a token is generated and returned to the client, which the client uses for authentication.

The aforementioned steps are illustrated pictorially showing client user interfaces in subsequent subsections.

5.1.1. Home. The GUISET home page is the first page that user encounter when they visit the GUISET portal, it is here where the register and login links are found.

5.1.2. Login. Login is the first step towards receiving authentication from the GUISET infrastructure to its underlying resource. Here the GUISET client is presented with the GUISET login page and is prompted to (1) Enter their GUISET user-ID and (2) Their GUISET password, in-order to gain access to GUISET portal. Two possibilities are presented here (1) Unsuccessful Login and (2) Successful Login as shown in Fig. 5 and Fig. 6 respectively. Successful login represent a login situation where valid credentials are submitted, otherwise is unsuccessful login.

Fig. 5.0. GUISET Log in Access denied

Fig. 6.0 GUISET Log in Access Granted

5.1.3. Security Token. After presenting their unique GUISET password, the client's user-ID and password are checked against the data repository to see if they exist in the database and whether there is a match. If this condition is true, the GUISET client is then presented with a security token which is what actually identifies and authenticates them.

5.1.4. Credential Manager Service. CMS manages the GUISET client credentials. It has a data repository that is invoked when user password and token are presented to check if the user is actually valid and exists in the database as well as the services they are authorized access on. That is, operation the CMS may perform on them such as performing an update on the user's credentials, and so on.

5.1.5. Implementation Result. This section illustrates diagrammatically how authentication process takes place in GUISET, taking the various functionalities into effect. This is shown in Fig. 7.0, where the various actors involved in GUISET authentication process interact as already discussed.

Fig 7.0. GUISET Authentication Process

6. Conclusion and Future Direction

User authentication is a crucial security component for most computing systems and since the security needs of different systems vary widely, authentication mechanisms are similarly diverse. In this work, we have explored the issues pertaining to authentication in grid services, issues such as protection of user credentials, and the need for single sign on. To deal with the issue at hand, an authentication framework was designed and implemented specifically for the GUISET architecture. The framework has been designed with the needs of the GUISET infrastructure in mind and thus it has been tailored to solve problems which are specific to GUISET authentication. These problems include but are not limited to issues such as the need to mitigate problems of having users provide multiple security credentials and taking UCSD into consideration when dealing with security systems that interface with users.

The framework presented as part of this work will lay the foundation for a secure authentication mechanism and thus pave the way for future work relating to authentication in the GUISET infrastructure. Also security systems will always be threaten by malicious attacks lunch by unscrupulous elements and thus there is need for continued research in the area since security will always be necessary to keep abreast of those that threaten to compromise security systems.

References

[1] Marvin A. Sirbu, John Chung-I Chuang: Distributed Authentication in Keberos Using Public Key Cryptography. Carnegie Mellon University. Proceedings of Network and Distributed Systems Security Symposium, 1997.

[2] Kerberos: The Network Authentication Protocol. Massachusetts Institute of Technology. 02 Jun 2009

[3] Jeffery Hassan, Maricuio Duran: Secure Web Services with WS-Security. Expert Service Oriented Architecture in C#, Chapter 6, 2005.

[4] Aaron Moss, Sandy Liu, Rene Richard: A unified Authentication Framework for Accessing Heterogeneous Web Service. Institute for Information Technology, National Research Council Canada. 4th International conference on Next Generation Web Services Practices. IEEE2008

[5] D. Carstens, P. McCaulley-Bell, L.C. Malone, and R. DeMara. Evaluation of Human Impact of password authentication practices on information security. Informing Science Journal. 7:67-86, 2004

[6] M.O. Adigun, O.J Emuoyibofarhe, S.O Migiro (2006) Challenges to Access and Opportunity to use SMME enabling Technologies in Africa. University of Zululand. All Africa Technology Diffusion Conference, June 12-14, 2006, Johannesburg South Africa.

[7] S.Z. Msane, O.O Ekabua, M.O Adigun: A Reputation-based Trust Management Model to Enforce Trust amongst Web Services in a GUISET Grid Environment. WorldComp'08: Conference on Semantic Web and Web Services (2008)

[8] Foster, I., Bo Lang, Frank, S., Rachana, A. and Tim, F.: A Multipolicy Authorization Framework for Grid Security. Proceedings of the 5th International Symposium on Network Computing and Applications. IEEE Computer and Society, Washington, DC, USA, pages 269-272, July 2006.

[9] Yair Amir, Cristina Nita-Rotaru, and Jonathan R. Stanton: Framework for Authentication and Access Control of Client-Server Group Communications Systems(2001). John Hopkins University.

[10] Center for Mobile e-Services for Development: Building the Infrastructure University of Zululand. WITFOR 2007

[11] M.G. Sibiya, E. Jembere, S.S Xulu: A Web Service based e-Commerce Business Model for Resource Constrained SMMEs. University of Zululand. SATNAC (2008)

[12] Ekabua O.O. and Adigun M.O: GUISET LogON: Design and Implementation Of GUISET-driven Authorization Framework. Proceeding of 1st International Conference on Cloud Computing, GRIDS and Virtualization, Lisbon, Portugal, November 21-26, 2010.

Coalition: a configurable co-reservation and co-allocation tool for supercomputing infrastructures

S. Campagna[1], **A. Vanni**[1]
[1]Supercomputing, Applications and Innovation Department, CINECA – Consorzio Universitario, 40033 Casalecchio di Reno, Bologna, Italy

Abstract *- Distributed supercomputing infrastructures provide users with large amount of heterogeneous computing resources, like computational resources, graphical resources, data processing and archiving resources etc… Usually, they are managed by batch sub-system in order to organize and schedule users' jobs. Batch sub-system varies site by site, machine by machine. On the other hand users may ask multiple resources at the same time for complex applications or workflows. This paper reports the work carried out in context of DEISA project for the co-reservation and co-allocation of heterogeneous distributed resources for enabling the execution of coupled applications.*

Keywords: Batch scheduler, resource co-reservation, resource co-allocation, distributed infrastructure, coupled application

1 Introduction

This paper describes the work performed in the context of DEISA2[1] project concerning a tool to reserve multiple computing resources (co-reservation) and to assign multiple resources to multiple user jobs (co-allocation) in a distributed heterogeneous High Performance Computing (HPC) infrastructure[1][2][3][4]. The main aim was to provide a configurable package able to search for a specific amount of free resources on the same time windows, spanning over different sites belonging to the DEISA infrastructure[5]. Afterwards, the requirements were extended in order to develop a tool broadly configurable, infrastructure independent and, possibly, without any software installation and configuration at the remote sites.

2 Distributed European Infrastructure for Supercomputing Application

2.1. DEISA project

The Distributed European Infrastructure for Supercomputing Applications (DEISA)[5] is a consortium of leading national Supercomputing centers that aims at fostering the pan-European world-leading computational science research. The partnership involves eleven European Union (EU) supercomputing centers providing a complex and heterogeneous HPC interconnected network. DEISA core members are: BSC (Barcelona, Spain), CINECA (Casalecchio di Reno, Italy), CSC (Espoo, Finland), ECMWF (Reading, UK), EPCC (Edinburgh, UK), FZJ (Juelich, Germany), HLRS (Stuttgart, Germany,) IDRIS (Orsay, France), LRZ (Garching, Germany), RZG (Garching, Germany) and SARA (Amsterdam, The Netherlands). Additionally, further centers joined DEISA as associate partners, they are: CEA-CCRT (France), CSCS (Manno, Switzerland), KTH (Stockholm, Sweden) and JSCC (Moscow, Russia).

The DEISA consortium has deployed and operated the Distributed European Infrastructure for Supercomputing Application, funded through the EU DEISA[2] project, from 2004 to 2008 and DEISA2 from May 2008 to April 2011.

The DEISA Extreme Computing Initiative (DECI[3]) started in 2005 to support computational intensive research activities . The aim was to select leading grand challenge HPC projects, based on a peer review system and approved by the DEISA Executive Committee, to enhance DEISA's impact on the advancement of computational sciences. By selecting the most appropriate supercomputer architectures for each project, DEISA is opening up the currently most powerful HPC architectures available in Europe for the most challenging projects. In addition, this service provisioning model has been extended by short-term project to support longer-term Virtual Science Communities. Collaborative activities are also being carried out with new European and other international initiatives. Since July 2010 the cooperation with PRACE[6] (Partnership for Advanced Computing in Europe) has started in order to put in place synergic actions for the creation of a European supercomputing eco-system made up of Tier-1 supercomputers (DEISA) and leadership-class Tier-0 supercomputers (PRACE).

[1] The project is funded by the European Union grant agreement FP7 - RI – 222919; from 01.05.2008 until 30.04.2011.

[2] The project is funded by the European Union grant agreement FP6 – RI - 508803; from 01.05.2005 until 30.04.2008.

[3] The project is funded by the European Union grant agreement FP6 – RI - 031513; from 01.06.2006 until 31.05.2008

2.2. DEISA infrastructure

The DEISA research infrastructure is constituted by leading national supercomputers in Europe interconnected with a high bandwidth point to point network provided by GEANT2[7] and by the National Research Networks (NRENs). High bandwidth network connectivity is required to guarantee the high performance of the distributed services and to avoid performance bottlenecks.

DEISA deploys different platforms, as IBM Power 6, IBM BP/p, SGI, Cray XT and NEC, and different operating systems AIX, Linux. The consortium has deployed middleware that enables the transparent access to distributed resources, high performance data sharing at European scale, and transparent job migration across similar platforms.

2.3. The DEISA network

The DEISA network provided by GEANT2 and the National Research and Education Networks offers dedicated bandwidth connecting the supercomputers which are spreading across Europe. This internal DEISA network has been built in addition to the standard Internet connectivity offered by each national supercomputer centre. Nearly all DEISA sites are now connected via 10 Gb/s lines. The current network uses dedicated wavelengths on NRENs and GEANT2 fibre-channel links footprint. It is managed by DEISA itself and operates at 10 Gb/s between nearly all DEISA sites. The Figure 1 shows the operative DEISA network.

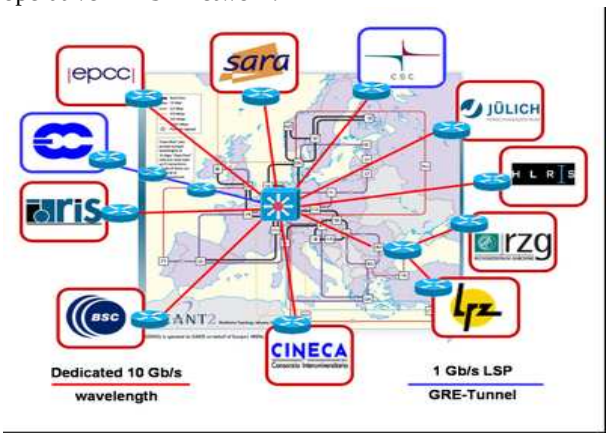


Figure 1 Schematic view of the DEISA high performance Network

2.4. The DEISA Global File System

Each supercomputing system has a common data space, a shared file system which is transparently accessible from every compute node in the cluster. Such a cluster-wide shared file system offers a single system view to compute jobs which are running locally on the cluster[8][9].

The DEISA Global File System, **Errore. L'origine riferimento non è stata trovata.**, extends this site-local data sharing model to a wide-area multi-cluster shared file system that is available within the DEISA HPC infrastructure. It overcomes the need for users to stage data explicitly from site to site before their compute jobs start execution at a remote site.

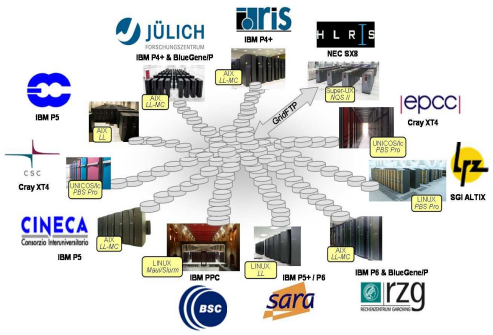


Figure 2: schematic view of the DEISA Global file system

2.5. DEISA software stack

DEISA operates a set of services, called the DEISA Common Production Services[10][11][12] (DCPS), to allow all users to achieve their scientific objectives conveniently and efficiently.

DEISA has defined a three-level classification of all its services, according to the service level targeted. They are named "*Core services*", "*Additional services*" and "*Optional services*":

- the *Core Services* are implemented and supported at each site, where technically possible. They are highly robust and will persist in the long term. DEISA provides users with a high support level for these Core Services. One of these services is Globus *gsissh*, which is exploited by the Coalition tool;
- the *Additional Services* are also supported at every DEISA site when technically possible, but the level of reliability and long term availability is lower than what guaranteed for the Core Services. They are supported on a best effort basis with a medium priority to solve the reported problems;
- the *Optional Services* might be supported on each DEISA platform, but they are not guaranteed to be supported in the near future, and have a low priority level of support.

An exhaustive description of this software stack and a detailed user guide can be found in the DEISA primer[11].

2.6. Interactive access to DEISA

DEISA consortium limited worldwide access for security and administration reasons via Globus *gsissh[14][15][16][17][18]* or *ssh*. Anyway, the *gsissh* access is guaranteed if a user operates into the DEISA network. But DEISA allows public access as well. A few DEISA sites provide access via *gsissh* from the public Internet, these sites are called *Door Nodes*. Figure 3 shows the Door Node concept and that there can be different Door Nodes for different services as well.

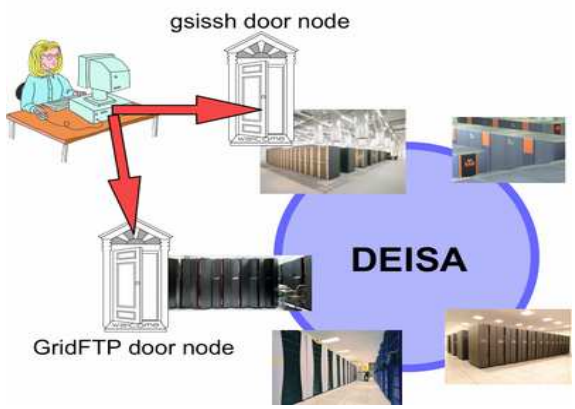


Figure 3: Door Node concept

Only a X.509[19] certificated is needed in order to authenticate, authorize and grant the user access to the entire DEISA infrastructure.

Coalition can use the *Door Node* in order to access all the DEISA clusters. Coalition can indeed be downloaded and installed by the user on its local workstation; the tool can then access the DEISA clusters using the available *Door Nodes*. The configuration for DEISA infrastructure can be done locally and is very simple.

3 Use case and user needs

During years the DEISA consortium has been facing with user request for high demanding complex workflows, sometimes requiring the execution of coupled applications that need co-allocation of resources spanning over different clusters at different sites. Required resources can be homogeneous or heterogeneous; for instance an application can require both computational and visualization resources, that are provided by separate clusters.

An example of a such application could be an atmosphere-ocean coupled model. In this kind of simulations, the output of the weather forecast is the input for the ocean model.

Some applications allow real-time visualization and steering of a running simulation. Application steering consists in changing some parameters of the running simulation, according to the information that real-time visualization of the output data can provide.

These use-cases demonstrated the need for a tool able to find out, reserve and allocate computational and/or visualization resources that are distributed on many clusters.

4 COALITION

Coalition has been developed to allow users to easily create co-reservations. The tool, which currently has a simple command line interface, is based on the Coalition library, providing all the relevant functionalities and modules. This library has an object-oriented design and is entirely implemented using the Python language. It is composed by four main modules: the multi-cluster description, the co-scheduler, the backend and the configuration module.

The multi-cluster description module provides static information about the infrastructure, like the number and identity of the available clusters, their hostname and connection protocol, etc... All this information is provided by the multi-cluster object, which is the core element of the Coalition architecture. All the relevant functionalities are provided through this object. The multi-cluster description module is responsible for correctly instantiating the multi-cluster object.

The co-scheduler module is the kernel of the tool, and is implemented by the multi-cluster object. It is responsible for analyzing the dynamic status of the resources and searching for common time windows for co-reservations. This module implements the algorithm for matching the resources slots with the user requests and discovering all the possible co-reservations.

The information about resources is made accessible by the backend module, which is responsible for gathering the status of resources from available clusters, and for organizing it in an appropriate structure for the co-scheduler. Moreover, if required, the backend module can create remotely the requested co-reservation based on the results provided by the co-scheduler algorithm.

This algorithm is very flexible and its behavior can be tuned by setting many parameters. The configuration module is responsible for providing these parameters to the co-scheduler. It receives a co-reservation request by the user and processes it to produce a valid configuration for the co-scheduler. The co-scheduler configuration sets the parameters for the co-reservation searching algorithm. Valid parameters might be the number of clusters in the co-reservation, the number of nodes, the duration, the minimum or maximum start time, etc. This module is strictly related to the user interface of the coalition tool.

The development in the context of the DEISA project required a mandatory design constraint for Coalition: the tool should be not invasive, and possibly should not require the installation of any component at the remote sites. For this last reasons we can consider it a client tool.

Due to this constraint, the backend module must connect to the clusters using existing protocols like *ssh* and/or *gsissh* and must collect information about jobs, nodes and reservations using the remote batch system command line interface, for instance, through llq, llstatus and llqres commands if the batch system is Tivoli LoadLeveler[20]. This restriction implies some limitations to the information that can be collected, whereas the command line interface does not provide all the available data about jobs, nodes and reservations. Using the scheduler Application Programming Interface (API) would provide much more information; in the future, Coalition could support the installation of some remote information provider to overcome this limitation.

The coalition workflow can be described as follows:

1. loading information about the infrastructure (multi-cluster description module);
2. processing user request (configuration module);
3. gathering information from the clusters (backend module);
4. searching co-reservations (co-scheduler module);
5. [optionally] creating the selected co-reservation (backend module).

Since the tool can be used also without creating any reservation, the last step is optional. This is useful for a "dry run", for instance to test the behavior of Coalition without submitting anything on the batch schedulers. Moreover, a user can exploit Coalition to simply query the dynamic status of the resources, in order to appropriately estimate and set up the resources to use for his jobs, even without creating any real reservation.

Figure 4 shows the coalition architecture and workflow. Green arrows represent all the collected information, that allow the co-scheduler to search for co-reservations. During this phase, resources information is collected from all the three clusters. Information about infrastructure is provided by the multi-cluster description module.

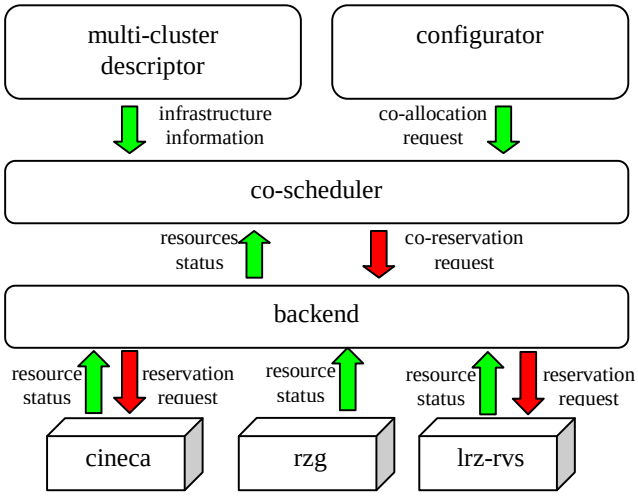


Figure 4: Coalition architecture and workflow

The co-reservation request, coming from the user and processed by configuration module, is made available to the co-scheduler. If there is a feasible co-reservation that satisfies the user request, then the co-scheduler can delegate the backend module to create all the necessary advance reservations on the target clusters; red arrows show this second phase. In the example, the selected co-reservation refers only to *cineca* and *lrz-rvs*.

Coalition implements much more functionalities than required by the DEISA project; it is in fact a flexible tool. It does not make any particular assumption on the cluster infrastructure, so it does not depend on the DEISA infrastructure; moreover, it can find co-reservations using a rich syntax, that allow to make generic co-reservation requests.

A functionality derived from the DEISA requirements is to divide the clusters in groups. In the DEISA use case, the groups are "computing" and "visualization". So a co-reservation request asks for a specific number of computing machines for each group. This grouping of clusters, even if not strictly necessary for the co-scheduler work, simplifies the co-reservation request and can improve the performance of the algorithm. Grouping is generalized, and it is possible to create as many groups as needed For example user can group cluster for operating systems, etc….

Cluster groups are defined by setting cluster properties; for instance if a cluster has the "visualization" property, then it automatically belongs to the "visualization" group. Each cluster can have zero or more properties, so it can belong to more than one group. By default Coalition provides a special group named *__all__* consisting of all the clusters. When there is no need to group clusters, but simply to search co-reservations among all the available cluster, the user can specify the *__all__* group .

4.1. Multi-cluster description module

In order to give the maximum flexibility, the definition of the infrastructure itself is being done through a plug-in, whose duty is to appropriately instantiate the multi-cluster object describing the infrastructure. The plug-in coding is quite simple, since it is only required to add the description of the available clusters to the multi-cluster object.

Each cluster belonging to the multi-cluster has an *Host* attribute, defining information like the hostname, the port and the connection protocol. The connection protocol is defined through the *Protocol* attribute, which determines how the host can be connected for the execution of commands on the remote host. At the time of writing supported protocols include *ssh*, *gsissh*; the execution on the local host is supported as well. Additional protocols can easily be plugged-in. For instance, the DEISA infrastructure required a special protocol called, "bridge", which exploits the DEISA *Door Node* for executing commands on the remote cluster. For instance, in order to run a command on the visualization cluster located at LRZ via the CINECA *Door Node*, it is necessary to run first a *gsissh* command on CINECA, and then, from here, a second *gsissh* command on the LRZ cluster . The code below is an example of single command that leverages the *Door Node* to execute the remote command qstat on the LRZ cluster:

```
$ gsissh grid.sp6.cineca.it -p 2222 \
    bash -c 'module load deisa globus ; \
    gsissh `deisa_service -i -s lrz-rvs` qstat'
```

Thanks to the delegation feature of *gsissh*, no additional authentication is required then the initial one.
Notice that creating the new "bridge" protocol does not require much effort, since it is based on the existing *gsissh* protocol.

4.2. Configuration module

The main goal of the configuration module is to parse a co-reservation request, which is described by using a specific syntax. The co-reservation request specifies many parameters used by the co-scheduler algorithm when searching for co-reservations:

- the name for the co-reservation;
- the max time window for the co-reservation (the co-reservation must be contained in this time_window);
- a minimum and maximum duration;
- margins and delays for the reservations;
- the minimum and maximum number of clusters, nodes, cpus and graphic cards;
- for each available cluster:
  - a boolean value stating if the cluster can or cannot be used;
  - the list of cluster properties, defining the cluster groups;
  - the minimum and maximum number of nodes, cpus and graphic cards;
  - the minimum and maximum number of cpus and graphic cards for each node of this cluster;
  - a filter function determining if a specific node slot can be used, for instance, this function can be used to select only full nodes;
- for each cluster group:
  - a boolean value stating if the group can or cannot be used;
  - the minimum and maximum number of cluster that can be chosen from this group.

As an example of the syntax, a typical DEISA co-reservation request looks like this:

```
01.# define visualization group:
02.lrz-rvs.properties='visualization'
03.# define computing group:
04.{__all__}-{visualization}.properties='computing'
05.# select exactly 1 visualization cluster and
06.# 1 visualization group:
07.[{visualization},{computing}].min=1
08.[{visualization},{computing}].max=1
09.# computing options:
10.${computing}.node_slot_options.filter=FULL_NODE
11.${computing}.min_cluster_resources.nodes=10
12.${computing}.max_cluster_resources.nodes=20
13.# visualization options:
14.${visualization}.node_slot_options.function=TRUE
15.${visualization}.min_cluster_resources.graphics=1
16.${visualization}.max_cluster_resources.graphics=1
17.${visualization}.min_cluster_resources.cpus=1
18.${visualization}.max_cluster_resources.cpus=1
19.# min and max duration:
20.min_duration=Duration(minutes=60)
21.max_duration=Duration(minutes=60)
22.# define coallocation r0:
23.name='r0'
24.min_duration=Duration(hours=1)
25.max_duration=Duration(hours=3)
26.time_window=TimeWindow( \
27. Time.fromstring(“20110225 16:10:00”), \
28. Time.fromstring(“20110226 12:00:00”) )
29.stop
30.name='r1'
31.min_duration=Duration(hours=2.5)
32.time_window=TimeWindow( \
33. COALLOCATION['r0'].slot.time_window.end+ \
34. Duration(minutes=10), \
35. COALLOCATION['r0'].slot.time_window.end+ \
36. Duration(days=1) )
```

As shown in this example, the co-reservation request syntax can be used for creating a co-reservation workflow as well; indeed, many co-reservation requests can be combined, defining some dependencies among them.
Line 2 sets the *visualization* property for the cluster *lrz-rvs*, then line 4 then sets the *computing* property for all the other clusters; after that, three groups exist: the predefined *__all__*, *visualization* (containing only *lrz-rvs)*, and *computing* (containing all but *lrz-rvs*). Lines 7 and 8 states that exactly one computing and one visualization cluster must be selected. Lines 10 to 18 set some properties for the clusters belonging to the *visualization* and *computing* groups. Lines 20 to 29 set other parameters for the first *co-reservation*, whose name has been set to *r0* at line 23. Then, lines 30 to 36 states that a second co-reservation, named *r1*, must be created; it inherits all the parameters of *r0*, and redefines only *min_duration* and *time_window*. Notice that the time window of *r1* is expressed in terms of found co-reservation *r0*: *r1* must start at least 10 minutes after the end of *r0*, and must not end more than 1 day after the end of *r0*.

4.3. Backend module

The backend module is composed by two main components. The first one allows Coalition to execute generic commands on the remote clusters, by using the information provided by the multi-cluster object. Indeed, as stated above, the multi-cluster information describes the cluster hostnames and connection protocols, that is sufficient to be able to execute remote commands.

Through the first component, the second one executes specific commands to gather information about the remote resources. Then the information is used to create an internal representation of the clusters. In order to give more accurate results, it is important that all scheduler data are collected exactly at the same time on all clusters; so, the related remote commands are executed at the same time in separate threads.

The collected data are simply the output of remote commands; this output is then parsed to recognize the definitions for jobs, nodes and reservations. This parsing is performed by a class that depends on the remote resource manager. Currently two resource managers are supported, LoadLeveler and Sun Grid Engine[21]. The two classes inherits by the same base class RM; support for other schedulers can easily be added.

After the parsing phase, resources information is injected in the multi-cluster objects, allowing co-scheduler to search for co-reservations.

Figure 5 shows the two steps performed by the backend module. First, information about the remote clusters is collected by the backend through the remote batch scheduler

interface. Then, this information is processed and structured according to the internal format expected by the co-scheduler.

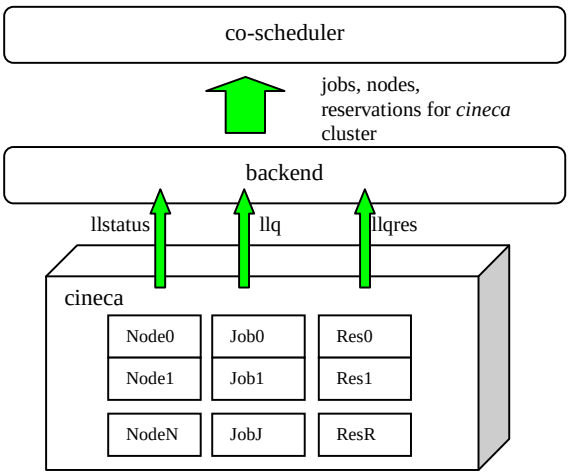


Figure 5: gathering and parsing of information about jobs, nodes and reservations

4.4. Co-scheduler module

The co-scheduler module is the Coalition kernel. It starts when the backend has finished gathering information about resources, and then looks for the requested co-reservations. The algorithm steps can be summarized as follows:

1. looking for matching slots on each cluster (cluster slots);
2. combining cluster slots in order to find a list of feasible co-reservations;
3. reducing co-reservations by applying the upper limits specified in the co-reservation request;
4. sorting the found co-reservations and selecting one.

During the first step, for each available cluster, the algorithm finds all the available slots of free resources matching only with the per-cluster requirements of the co-reservation request. For instance, with the configuration shown above, when looking for the r0 co-reservation on a computing cluster, Coalition will try to find all slots of at least 10 nodes that are free for at least 60' of duration; the slot start time must also match with the time_window keyword, so it must not start before 20110225 16:10:00, and with the min_duration keyword, so it must not start after 20110226 11:00:00 .

At this stage, each cluster has a list of free slots that can match with the co-reservation request, and it is so possible to start the second step, during which cluster slots are combined to form co-reservations. Suppose we have three computing clusters, cineca, rzg and fzj-bg, and a single visualization cluster lrz-rvs. The above co-reservation request asks for one computing and one visualization cluster; lrz-rvs is a visualization cluster, rzg and cineca are computing clusters. So, the co-reservation can be found combining free slots for all the possible combinations of one computing and one visualization cluster: (cineca, lrz-rvs), (rzg, lrz-rvs), and (fzj-bg, lrz-rvs). The second step consists of matching the free slots of the clusters belonging to these combinations; in this phase, also parameters related to the full co-reservation are considered, for instance the minimum number of nodes of the entire co-reservation.

The result is a list of possible co-reservations, where each co-reservation is a selection of slots coming from one of the three sets of clusters. Each co-reservation has a related time window and specific resources.

Notice that during all the co-reservation selection process, in both steps, only lower limits of the parameters are taken into account; for instance, min_duration is considered, but not max_duration. Indeed generally after the first two phases Coalition provides co-reservations that are "bigger" than requested; for instance, the maximum allowed duration can be infinite, or the number of free nodes can be higher than strictly requested by the user.

The third step consists in applying the upper limits to the found co-reservations. For instance, considering r0, one of the found co-reservations can have a time window starting at 20110225 16:10:00 and ending at +inf; after the third step, the time window will be reduced to 20110225 16:10:00 - 20110225 19:10:00, since max_duration is 3 hours.

Finally, since only one co-reservation must be created, the list of resulting co-reservations must be sorted according to a comparison criterion, and the best one is selected. The default criterion is to select the co-reservation with the earliest start time, and potentially, if more co-reservations have the same start time, the bigger one.

4.5. Example

Figure 6 shows the last output lines of Coalition. The co-reservation request is exactly the same shown above. In this case, both r0 and r1 have been found on the same two clusters *cineca* and *lrz-rvs*, even if this was not strictly required.

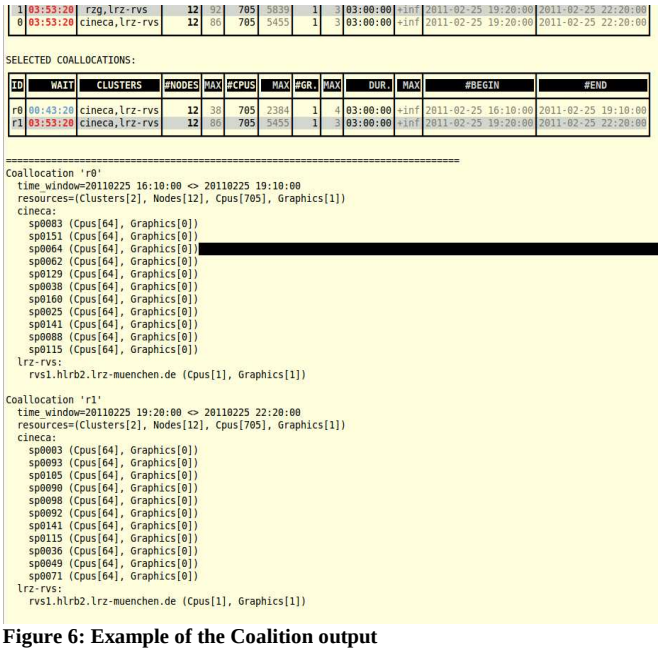


Figure 6: Example of the Coalition output

4.6. Implementation

Coalition is implemented in Python, following an object oriented design. Extensions can easily be written by inheriting from the main classes. Installation is quite easy, since it depends only on a standard python installation (ver. >= 2.6).

The deployment of the tool on an existing infrastructure requires some configuration. Indeed, the current implementation of the multi-cluster description module requires the definition of a python plug-in class, whose constructor fully describes the available clusters. At this stage, it can also be necessary to implement infrastructure specific protocols to connect to the clusters; for instance, the DEISA multi-cluster class uses the DEISA specific "*bridge*", that can easily be coded by inheriting and using the existing protocols.

Implementation of the Python plug-in must be done once per infrastructure. This is the only action that the infrastructure administrators or user support team should execute; in any case, nothing must be installed at the remote sites.

The plug-in can also be provided on-line; Coalition is able to download and dynamically load it by using protocols like http or ftp. In this case, client-side configuration of the tool consists simply in putting the URL of the plug-in on a configuration file.

5 Conclusion and Future work

This work demonstrated that the tool is fruitful to simplify the discovery of resources and the management of reservations and allocations for complex simulations, by providing a flexible set of configuration options capable to describe requests for application workflows. For instance, it helps finding and allocating resources for coupled applications that require heterogeneous resources distributed on two different clusters.. Even if it is infrastructure independent and not invasive, this tool is able to describe heterogeneous and distributed environments with a high degree of details.

One of the most important limitations of Coalition is due to the constraint to avoid installation of software on the clusters. Coalition is extremely quite light and this limits the information that can be collected, and increases the time to gather and process the information.

It is possible to implement remote information providers, that are installed on the clusters and here collect information using standard APIs. These information providers can be implemented using the scheduler's API, or the DRMAA API.

There is no need for changes in the Coalition client: it is simply required to implement a new RM class that processes the information collected by the remote providers; every remote provider has the same front-end interface.

Another possible enhancement could be providing a new graphical interface to simplify the description of the infrastructure having complete visual dashboard that the user can utilise to specify the co-reservations.

6 References

[1] Ammar H. Alhusaini, Viktor K. Prasanna, C. S. Raghavendra. A Framework for Mapping with Resource Co-Allocation in Heterogeneous Computing Systems. HCW '00 Proceedings of the 9th Heterogeneous Computing Workshop.

[2] Neena Kaushik, Silvia Figueira, Stephen A. Chiappari. ACM SIGMETRICS Performance Evaluation Review archive; volume 35 Issue 3, December 2007.

[3] Claris Castillo, George N. Rouskas, Khaled Harfoush. Resource co-allocation for large-scale distributed environments. HPDC '09 Proceedings of the 18th ACM international symposium on high performance distributed computing.

[4] Karl Czajkowski , Ian Foster , Carl Kesselman. Resource Co-Allocation in Computational Grids. Proceedings of the 8th IEEE international symposium on high performance distributed computing.

[5] DEISA infrastructure, www.deisa.eu

[6] PRACE iniziative, http://www.prace-project.eu

[7] GEANT2, http://www.geant2.net/

[8] P. Andrews, C. Jordan and H. Lederer: Design, Implementation, and Production Experiences of a Global Storage Grid, NASA/IEEE Conference on Mass Storage Systems and Technologies, Maryland/US, May 2006.

[9] H. Lederer: DEISA2: Supporting and developing a European high-performance computing ecosystem, Journal of Physics: Conference Series 125 (2008) 011003 doi:10.1088/1742-6596/125/1/011003

[10] DEISA common production environment DCPE, http://www.deisa.eu/usersupport/user-documentation/deisa-common-production-environment.

[11] DEISA primer, http://www.deisa.eu/usersupport/primer/primer.pdf

[12] W. Gentzsch, D. Girou, A. Kennedy, H. Lederer, J. Reetz, Morris Riedel, A. Schott, A. Vanni, M. Vazquez, J. Wolfrat: DEISA – Distributed European Infrastructure for Supercomputing Application, JoGC: Grid453R1 Journal of Grid Computing.

[13] Globus Alliance, http://www.globus.org/

[14] I. Foster, C. Kesselman, G. Tsudik, S. Tuecke. A Security Architecture for Computational Grids. Proc. 5th ACM Conference on Computer and Communications Security Conference, pp. 83-92, 1998.

[15] V. Welch, F. Siebenlist, I. Foster, J. Bresnahan, K. Czajkowski, J. Gawor, C. Kesselman, S. Meder, L. Pearlman, S. Tuecke. Security for Grid Services. Twelfth International Symposium on High Performance Distributed Computing (HPDC-12), IEEE Press, to appear June 2003.

[16] V. Welch, I. Foster, C. Kesselman, O. Mulmo, L. Pearlman, S. Tuecke, J. Gawor, S. Meder, F. Siebenlist. X.509 Proxy Certificates for Dynamic Delegation. 3rd Annual PKI R&D Workshop, 2004.

[17] OpenSSH, http://www.openssh.org/

[18] GSI-OpenSSH, http://www.globus.org/toolkit/docs/5.0/5.0.3/security/openssh/

[19] X.509 Public Key Infrastructure, http://www.ietf.org/rfc/rfc3280.txt

[20] LoadLeveler, http://www-03.ibm.com/systems/software/loadleveler/

[21] Oracle Grid Engine, http://www.oracle.com/us/products/tools/oracle-grid-engine-075549.html

Simone Campagna graduated in Physics at the University of Bologna, he is a staff member of CINECA since 1999. His work at CINECA is mainly focused on meteorological and climatological activities: he developed and currently manages the operational chain providing weather forecast to the Italian Civil Protection. He also developed some other operational chains about similar services. He has good knowledge of many computer languages (FORTRAN,C,C++,Python) and of the object oriented analysis, design/programming model. Currently he is developing software for the ENI oil company.

Andrea Vanni gained a Degree in Computer Science at Università di Bologna. From 2002 he is confirmed employee at CINECA. He is working on Grid computing. He has been involved in several EU projects such: eDeisa, HPC-Europa and Unigrids. He has a wide experience on grid solutions developed for Unicore, Globus and Gridsphere and the management of a distributed computing infrastructure. He leaded the technologies and middleware work package for eDEISA project and he is a member of the PRACE project. Since 2008 he is the leader of the DEISA2 technology team and member of the DEISA technical board.

SESSION

RESOURCE DISCOVERY AND MONITORING + SECURITY ISSUES

Chair(s)

TBA

A Monitoring System Based on Nagios for Data Grid Environments*

Hsiu-Lien Yeh[1], Yan-Fu Chen[1], Tsung-Tai Yeh[2], Pei-Chi Huang[2], Shin-Hao Liu[2], Hsin-Wen Wei[2a] and Tsan-sheng Hsu[2]

[1] Institute of Information System and Applications, National Tsing Hua University, Taiwan
[2] Institute of Information Science, Academia Sinica Taipei, Taiwan

Abstract - *The amount of digital data in today's society is already enormous and it will continue to grow exponentially. Therefore, it is necessary to devise new ways to preserve and manage the data effectively and efficiently. SRB (Storage Resource Broker), and its extension iRODS (the Integrated Rule-Oriented Data System), are data grid technologies for managing colossal amounts of data. In a distributed environment, monitoring systems oversee the operation of computing systems. The monitoring service is crucial because it must ensure a high-quality computing environment and provide reliable services. In this paper, we introduce a monitoring system called SIAM, which is based on Nagios. SIAM supports full monitoring services for SRB/iRODS-based systems, including fault-tolerance and notification functions. This study focuses on extending existing components and notification functions to satisfy clients' needs and improve our system's failover scheme. The results of experiments show that the proposed system is feasible for cloud storage services, and it is adaptable robust, and responsive in the face of system failures. Overall, SIAM enhances the reliability of SRB/iRODS based systems significantly.*

Keywords: Monitoring; Distributed system; Fault- tolerance

1 Introduction

As the digital era continues to evolve, the volume of digital information will grow exponentially. According to the International Data Corporation (IDC), on a global level, the amount of digital information doubles each year [1]. As a result, a great deal of research effort has focused on data preservation and management issues. How to manage, store, maintain and understand the collected data are critical issues. In recent years, data grid systems have been used extensively to handle huge amounts of data, and the grid concept may provide solutions to the above-mentioned issues. According to a number of studies [2][3][4], SRB and iRODS are two of the most widely used data grid systems. The architecture of iRODS is modeled on that of SRB, and combined SRB/iRODS systems are used in many academic projects, as well as in national and international research institutions. Such systems facilitate improved information collaboration and management of mass data storage [5]. Additionally, a monitoring system is usually designed to address certain stability and availability issues in a data system; hence, many research institutes and companies devote considerable resources to designing highly efficient monitoring systems [6][7][8]. The purpose of a monitoring system is to detect system faults immediately. In cases where the system has failed or is about to fail due to excess information or the computational load, e.g., disk corruption, overloading, or communication hot spots, a monitoring system sends out a warning signal, and thereby prevents a system failure or at least reduces the system's downtime. The monitoring system also ensures that grid computing services are not interrupted; therefore, the risk of losing data after a breakdown is minimized, service quality is improved, and clients are satisfied. Consequently, such systems play a major role in helping service providers to ensure the reliability and usability of their services. In addition, a failover handler is an essential mechanism that ensures the operation of monitoring systems is not interrupted.

Several monitoring systems have been designed for various computing environments. Generally, the systems can be classified as built-in systems or distributed systems. The first type is an application that runs in an operating system, such as a Unix-like system [9], which normally contains a monitoring subsystem that analyses a computer's current operational status. The second type is a monitoring host that is capable of observing various machines, such as Nagios, Ganglia, Cacti, and Hawkeye [10][11][12][13], simultaneously. In both cases, a monitoring system requires an active response mechanism for different devices, so that it can quickly identify potential failures and transmit warning messages. Monitoring systems must deal with a variety of dynamic resources available in a distributed environment; thus, special mechanisms must be designed and incorporated

*This research was supported in part by the National Science Council,Taiwan, R.O.C., under Grant NSC 95-3114-P-001-007-MY3 and NSC99-2631-H-001-024.

[a] Corresponding author: hwwei@iis.sinica.edu.tw;
Institute of Information Science, Academia Sinica, No 128, Section 2, Academia Road, Nankang, Taipei, Taiwan, R.O.C;
Phone: 886-2-2788-3799 ext.2471; Fax: 886-2-2782-4814

into the system to satisfy the specific requirements of the environment.

Some researchers, such as Zanikolas [6] and Massie [14], have observed that designers typically need to consider several features before implementing a monitoring system. According to our survey, the key design features of monitoring systems include scalability, extensibility, portability, robustness, manageability, reasonable overhead, and security. After conducting a thorough survey, we chose the Nagios open source program for our implementation. Nagios is utilized by several monitoring systems, such as GridICE and EGEE [15][16][17], and is supported by a strong open source community, e.g. NRPE, NDOUtils, and PNP [17][18][19]. However, Nagios does not support the SRB/iRODS system or the graphical visualization of the Nagios front-end website; instead it just provides regular notifications via E-mail. Besides, the Nagios monitoring system causes malfunctions and stops entirely if an error occurs in one of the monitoring hosts. This problem could be avoided if the system contained an effective backup mechanism, because a backup monitoring host would take over immediately and the service would not be interrupted.

Nagios is integrated with the monitoring architecture, which oversees the SRB/iRODS system and servers in the distributed environment. The SIAM system, which works independently of the computing system, tracks the activities of servers and the computing system, and uses several real-time notification services to inform system administrators when faults occur. Since SIAM utilizes Nagios open source software, it can be extended and maintained based on the requirements of the system components and services. As well as being extendable, it is readily available, efficient, and easy to integrate. SIAM provides an appropriate infrastructure for monitoring a data system environment; hence, it can easily detect the real-time status of the servers and systems. It also contains a fully tested fault-tolerance mechanism and only incurs a small additional overhead. When a system malfunctions suddenly or shuts down, SIAM prevents a total failure by enabling the system to reboot and continue operations immediately. In this paper, we implement a monitoring system based on Nagios for a grid environment, and test its ability to extend the monitoring function.

The remainder of this paper is organized as follows. In Section 2, we review existing monitoring system architectures. In Section 3, we describe the proposed extension of the Sinica SRB/iRODS Monitoring System; and in Section 4, we evaluate the system's performance. Section 5 contains some concluding remarks.

2 Survey of related software

In this section, we compare four widely used monitoring systems, and discuss their limitations. We discuss Nagios in more detail because it provides the basis for our system. Three known extensions of Nagios are also considered. Monitoring systems like Ganglia, Cacti, Hawkeye, and Nagios provide basic functionality for monitoring hosts, services, and resources. We describe those systems below.

Table 1. The comparison of different monitoring system.

Monitoring System	**Advantages**
Ganglia	Scalable architecture (clusters in particular) Graphic support Basic historical data analysis
Cacti	Excellent graphic displays Web management interface
Hawkeye	Notification mechanism Multiplatform Possible custom-made sensors
Nagios	Excellent extensibility Notification mechanism Low overload
Monitoring System	**Disadvantages**
Ganglia	No web management interface Complicated system settings
Cacti	Poor extensibility No notification mechanism
Hawkeye	Poor front-end The system is under-developed
Nagios	No graphic display support No web management interface
Monitoring System	**Key Design Features**
Ganglia	Scalability Robustness Reasonable Overhead Portability
Cacti	Manageability Robustness Overhead
Hawkeye	Scalability Extensibility Overhead
Nagios	Scalability Extensibility Robustness Security Overhead

Ganglia, an open source distributed monitoring system developed by the UC Berkeley Millennium Project [11][14], has a hierarchical architecture and relies on a multicast-based announce protocol to monitor the states of systems. It also uses technologies like XML for data representation, PHP for web development, and RRDTool (Round Robin Database) for data visualization. RRDTool is a popular application for storing time series data in graphic form [9][20].

Cacti is a network monitoring system that presents the system performance in graphic form [12]. It utilizes the SNMP protocol to collect information from various monitoring machines, RRDTool for the graphical presentation of monitoring information on web homepages, and a MySQL database for data storage.

Hawkeye, developed by the Condor group, is designed to monitor distributed systems [8][13]. It is implemented in two stages. First, using the Condor ClassAd Language, Hawkeye identifies problems based on the attribute values of the resources, namely, a ClassAd. Second, the manager can collect user information and deal with user inquiries when a problem occurs. Because Hawkeye uses Condor ClassAd

Language and ClassAd as building blocks, managers find the system easy to operate.

Nagios is a widely used and scalable monitoring tool developed by Ethan Galstad in 1999 [10]. It is an open source software framework under General Public License (GPL), and it provides many standard and specialized plugins for grid systems.

Table 1 compares the advantages and disadvantages of the four systems described above. Ganglia's installation procedure is quite complicated for users; Cacti has less configuration extensibility; and Hawkeye is limited because it is under-developed. In contrast, Nagios is easier implement, more extensible and easier to maintain. Thus, we selected Nagios as our model framework. For a more extensive introduction to these systems, readers may refer to the following works [8] [9] [10] [11] [12] [13] [14].

2.1 Nagios

The Nagios architecture involves a client-server layout, so it is easy to incorporate customized plugins, and system developers can try to satisfy diverse requirements by modifying or improving the system's functionality. The main functions of the Nagios system are: monitoring the utilization of several grid middleware services and infrastructures, helping web interface-enabled users to view generated reports and monitor tuning, supporting widely used notification mechanisms (E-mail), and supporting monitoring with automatic event handlers. Nagios Process Check Logic and Nagios Remote Plugin serve as add-ons. Plugins are written in C and consist of execution scripts. Based on the hardware and software characteristics, system administrators can allow other applications as add-ons to the basic plugins. Nagios provides system administrators with five mechanisms for communicating with different types of monitoring apparatus.

2.2 Extension of Nagios

NRPE (Nagios Remote Plugin Executor) is a basic secure tool set that enables monitoring of remote hosts [17], and is used to drive Nagios Plugin applications on remote hosts.

NDOUtils is an add-on application for the Nagios system. It stores profile parameters and the monitoring records of the system's status in a database [18].

The Nagios add-on PNP application "PNP-is-not-PerfParse" provides graphical functions to display monitored information [19]. PNP analyzes Nagios' monitoring services and generates performance data. It also stores the monitoring information automatically by using RRDtool, which helps in collecting data and displaying various graphics. Graphical tools can also be configured by PNP software.

2.3 Limitations of software based on Nagios

Software based on Nagios has the following limitations:

- It is not specifically designed for SRB/iRODS systems.
- It only provides a web interface to view monitored information. It is difficult for system administrators to use because the interface lacks graphical support and it cannot be modified.
- It sends a message if an error occurs, but it is difficult for administrators to distinguish between high priority error messages that need immediate attention and less urgent warnings. That is, users need to be able to customize the levels of error messages according their needs.
- It only offers one way to receive notifications, i.e., through e-mail.
- It lacks a built in mechanism for fault tolerance.

We integrate a monitoring system to overcome these limitations.

3 Sinica SRB/iRODS monitoring system

SIAM was built for the data preservation systems developed for the Digital Archives Remote-Backup (DARB) project [21][22] as part of TELDAP (Taiwan e-Learning and Digital Archives Program) [23]. The design of the data preservation system in DARB is based on SRB/iRODS middleware. To provide monitoring services for the data preservation system, we proposed using SIAM, and based the design of our system on the features of SRB/iRODS. We describe the features and components of the SIAM system in the following subsections.

3.1 Basic features

We use Nagios as the core system in the design of SIAM and, based on the features of SRB/iRODS, we offer various components as extensions of the monitoring system. Figure 1 illustrates the utilization of SIAM in a data grid environment. We extend the Nagios monitoring service to monitor the overall data preservation system, and improve the main features of the monitoring system to provide the following functions: service availability checks, system error detections, and resource management.

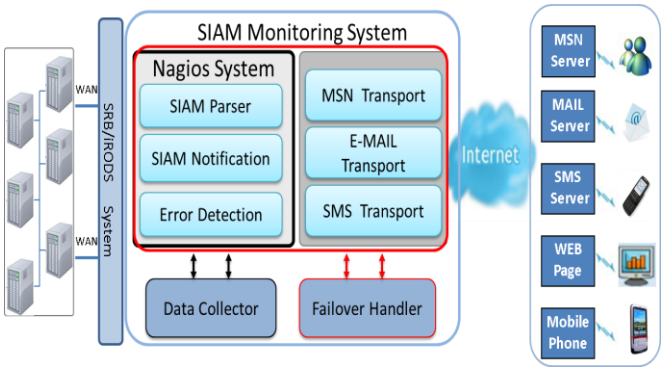


Figure 1. The configuration of SIAM in a data grid environment.

3.1.1 Service availability checks

The monitoring system periodically tests the accessibility of the data preservation server. If a login operation fails, the

monitoring application will wait for a period of time and then try to perform the login operation again. The process is repeated up to three times, after which the monitoring system immediately notifies the system administrators of the failed login attempt.

3.1.2 System error detections

The data preservation system keeps track of error messages and system processes and stores the information in log files. SIAM parses the files so that the monitoring system can identify important messages about the computing system and related components in the preservation system. When SIAM finds error messages, it identifies the relevant information and notifies the system administrators. Thus, the parser tool helps system administrators trace faults in the system and take remedial action immediately. The parser tool also allows users to define the rules to specify the tasks that should be executed if an error occurs. In summary, the parser tool is responsible for: periodically parsing the log files created by the systems and servers on the monitoring host; filtering out error messages or error strings in the log files; and recording detected errors or error strings, and sending warning messages to the system administrators.

3.1.3 Resource management

The data preservation system provides users with a logic space resource to store data. It uses a disk array as a storage space, and SIAM monitors the used capacity of all disk partitions simultaneously. The number of monitoring disks in the SIAM system depends on the size of each disk array. If the used disk space exceeds a pre-defined limit, the system will display a "Warning" or "Critical" signal. Users can customize the threshold value of each used disk space and the displayed signals according to their needs.

3.2 Advanced features of SIAM system

3.2.1 Real-time notification services

In general, Nagios does not support the SRB/iRODS system, and it only provides notifications via E-mail message. In contrast, our monitoring system utilizes various communication protocols to provide a notification service, e.g., e-mail, mobile phones, web pages, and other on-demand services. SIAM can track the status of all systems and servers in the cloud, and detect errors that occur in the data preservation system. It then alerts system administrators via real-time cloud notification services. SIAM enhances the notification mechanism by supporting other real-time cloud notification services, such as Windows Live Messenger (MSN) and Short Message Service (SMS) on mobile phones. As defined in the system configuration file, when SIAM discovers an error message, the monitoring system orders the MSN robot to transmit a message to the Microsoft MSN server via the Microsoft Notification Protocol (MSNP), and send a message to the specified MSN account. It can also use Perl script to send a message to the system administrator's mobile phone. SIAM ranks the levels of error messages by their importance, and sends corresponding notification messages to the system administrators. For example, when a system server fails or shuts down unexpectedly, SIAM dispatches a critical message to the administrator's mobile phone immediately. At the same time, the system administrator will receive an error message vial MSN or e-mail. These real-time notification services help administrators manage their systems effectively.

3.2.2 Fault-tolerance scheme

The SIAM monitoring system works independently of grid systems and provides a fault-tolerance mechanism to improve the reliability of monitoring services. Since the Nagios system does not support such a mechanism, SIAM implements a failover handler as the fault-tolerance scheme, as shown in Figure 2. The failover handler operates as follows. First, all files are backed up from the SIAM master host to the slave host, which is monitored by the SIAM monitoring system, as shown in steps 1 and 2. After installing the MySQL database between the master host and the slave host, both hosts execute their MySQL replication applications via the database. This ensures that the information backed up between the two hosts is consistent (step 3). If SIAM detects a critical fault or the master host fails, the slave host will send notification messages to the system administrators by e-mail, MSN, or SMS (step 4). The SIAM backup file is then decompressed automatically, and the slave host is substituted for the master host. Specifically, the slave host reboots and takes over as the new master host (steps 6 and 7). In this way, the failover handler ensures that 1) the monitoring service is not interrupted; 2) system administrators receive warning messages immediately so that they can take remedial action; and 3) data is not lost in the event of a serious system failure. The use of master and slave hosts results in a lower overhead and enhanced scalability in the distributed system.

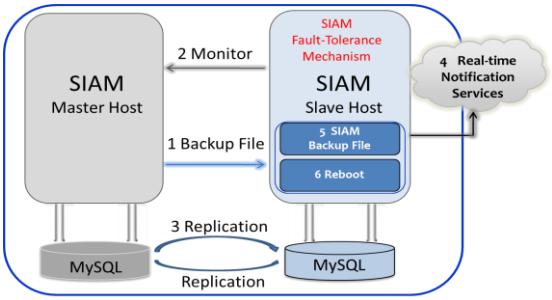


Figure 2. The SIAM fault-tolerance framework

3.3 SIAM system components

SIAM is designed for an SRB/iRODS-based data preservation system. In this subsection, we present the components of SIAM by mapping each one to a specific monitoring phase. The system contains five levels, and utilizes the Nagios Core and the standard interface to display data about various resources, services, and hosts (Figure 3).

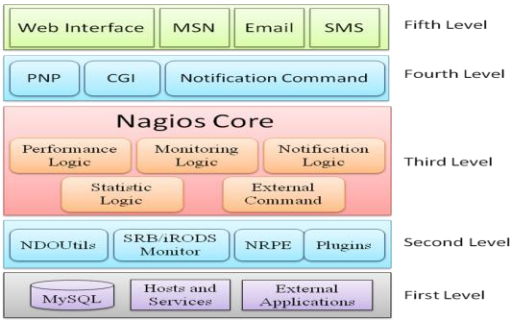


Figure 3. The components of SIAM monitoring system.

Level 1: This level stores the information gathered from hosts, services, and external applications in the database. Thus, it provides a pool of historical and periodical monitoring information, which is stored in the database. Note that the SIAM system, which can be deployed as independent server machine in a distributed environment, records the activities of the server, database, and data preservation system and notifies system administrators when errors occur.

Level 2: Level 2 contains the main applications and communication protocols that support data transfer. To ensure that users can connect to the SRB/iRODS server via a remote network and transfer data, the monitoring application must connect with the SRB/iRODS system by using remote access. Consequently, we adopt the Nagios Remote Plugin Executor (NRPE) to communicate with remotely monitored machines. The monitoring system also utilizes the NDOMOD Event Broker module and the NDO2DB Daemon in NDOUtils, which is an add-on application for the Data Collector. The NDOMOD module extracts data from Nagios and converts it to TCP socket format. The NDO2DB Daemon then stores the data in a database.

Level 3: The third level of the structure is the system kernel. The Nagios Core has five interactive components: Performance Logic, Monitoring Logic, Notification Logic, Statistic Logic, and External Command. Based on the Nagios configuration file for implementing applications, the system ensures that operations conform to the appropriate system settings.

Level 4: This level contains an application interface that communicates with the Nagios Core to control the monitoring services, display monitoring information on web pages, and send notification messages via various communication channels. Here, the monitoring system uses the graphical interface to present system statistics through the PNP application. If the monitored data, such as the CPU load and disk utilization, can be displayed and checked on the screen, system administrators can follow changes in each system via the monitoring service. SIAM enables clients to manage information without installing specific software.

Level 5: The fifth level contains an application interface that communicates with the Nagios Core to control the monitoring services, display monitoring information on web pages, and send notification messages via various communication channels. The system's web page displays the current monitoring information of the entire system. Furthermore, users can use the notification services via a Web interface and receive notifications via e-mail, MSN, or SMS.

4 Experiments

In this section, we describe the experimental environment and implementation of the SIAM system. We also demonstrate the scalability of the components in the monitoring system and evaluate the system's performance.

4.1 Experimental environment

The experimental environment contains a master host, a slave host, and other nodes that are monitored by the master host in a distributed environment. Each monitoring host is equipped with a Xeon 3.50GHz CPU and 4G memory, so its computing power is sufficient to handle a large number of nodes. In a monitoring environment, the hardware of each node includes IBM System x3650, Cisco Catalyst 3750 switch, APC Smart-UPS RT 7500 UPS, and disk array 10TB~80TB. In addition, different types of software, such as SRB, iRODS, and Oracle10g R2, are installed on each node.

4.2 SIAM monitoring system demonstration

In this section, we consider the key functions of the SIAM monitoring system, namely, service availability checks, resource allocation management, system error detection and the failover handler mechanism. As shown in Figure 4, when a provided service in the "th" node cannot be accessed, a CRITICAL message is displayed in the seventh row of the SIAM web page. The user can then click on the hyperlink "th," to obtain more detailed information, such as the service name, error timestamp, and status description (see Figure 5). Figure 6 shows an example of a CRITICAL error message generated by SIAM when monitoring the grid system installed on the "ncl" host. The message is forwarded to the e-mail and MSN accounts of the system administrators via the real-time notification system. Figure 7 displays information about the resources used by the monitored nodes, e.g., the CPU load, disk utilization, and status of the Oracle database.

The SIAM system's user-friendly web interface provides general host information, service information, and other data, as shown in Figure 8(a). It enables system administrators to configure SIAM, and provides important information for users, including the monitoring status of each category, a list of executed applications in the system, and a message and notification history record. If the master host fails to monitor distributed nodes, the slave host automatically takes over the monitoring duties and acts as the master host. To implement the failover handler, the slave host first detects the system's status through the CRITICAL error messages provided by the service availability check function. Next, the system administrators are sent a real-time cloud notification via e-mail, MSN and SMS. Then, the slave host replaces the master host and takes over the monitoring services, as shown in Figure 8(a) and (b).

Figure 4. Services availability checks.

Figure 5. SMS, MSN and E-mail notification dialogue.

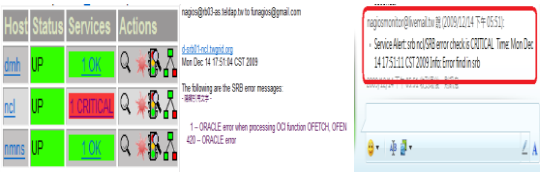


Figure 6. Real-time notification message dialogue.

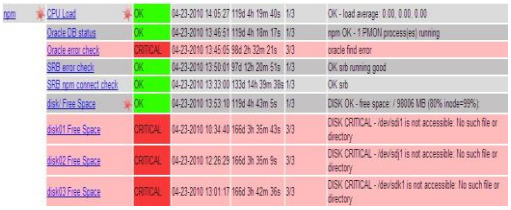


Figure 7. Resource management.

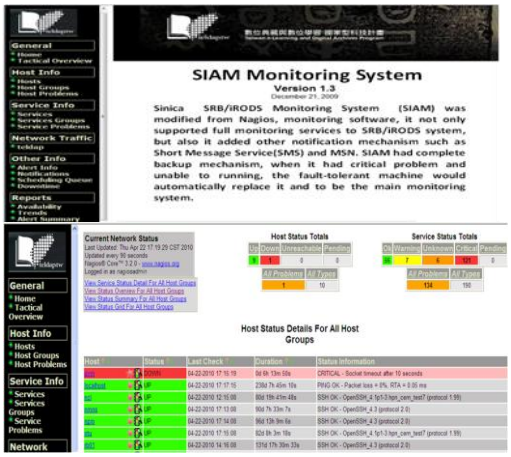


Figure 8. (a) The web page provided by the master host. (b) The web page provided by the slave host after failover.

4.3 Performance evaluation

Several important factors must be considered when evaluating the performance of SIAM, including resource utilization, error notification time, and system robustness. Resource utilization means the amount of the monitoring host's operating space that SIAM occupies while executing tasks. The notification time, which is set by the SIAM system, is the time required to dispatch notification messages when a system server fails. Robustness refers to a slave host's ability to take over from a master host in the event of a failure.

The first experiment considers the resource utilization of SIAM. In the event that the master host generates an unexpectedly large number of CRITICAL messages (such as 50,100, 200, or 300, as shown in Table 2), the operation consumes less than 1.0% of the CPU resources and 0.1% of the memory. In Figure 9, we plot the percentage average CPU usage and memory usage against the number of messages. The graph shows that, although the volume of messages increases significantly, the average resource consumption is relatively small. Furthermore, since monitored nodes use the Nagios NRPE command, SIAM occupies less system space, which in turn reduces the system overhead.

Table 2. The simulation results of resource utilization.

Monitoring Host	Resource Utilization			
	50 Critical Messages	100 Critical Messages	200 Critical Messages	300 Critical Messages
CPU Utilization	0.3%	0.3%	0.5%	0.7%
Memory Utilization	0.1%	0.1%	0.1%	0.1%

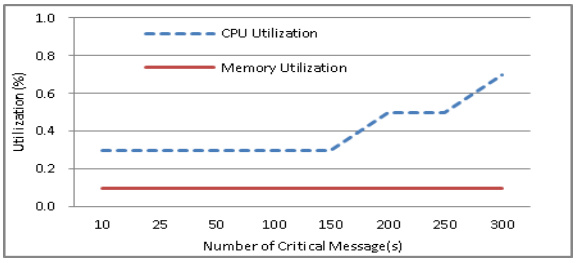


Figure 9. SIAM resource utilization.

The second experiment considers the notification time. One advantage of SIAM is that it can be customized to send out messages at different rates. In this experiment, the monitoring time interval is set at 60 minutes; that is, the system only checks for errors every 60 minutes. When the SIAM host detects a system failure in a monitored host, depending on the severity of the error, SIAM should notify the system administrators as user's required after the failure is detected. For example, if SIAM is configured to notify system administrators within 3 minutes of detecting a failure and the monitoring time interval is 60 minutes, then, in the worst case, SIAM will dispatch an alert message 62 minutes and 59 seconds (3779 seconds) after an error occurs. In the best case, where an error occurs exactly at the beginning of an interval, SIAM will respond within three minutes (180 seconds). When SIAM dispatches an alert, system administrators are notified immediately (in less than one minute) via real-time cloud services, such as MSN, SMS and e-mail. Generally, these three methods are equally fast. SMS is particularly convenient since mobile devices are portable, so administrators can receive notifications anytime, anywhere. Table 3 summarizes the notification times and notified service transfer times in the experiment.

The third experiment examines the robustness of the failover scheme in terms of the failover time (Table 3). Here, we assume that the monitoring time interval is ten minutes,

and the degree of urgency of the error message is set at level A. In SIAM, system damage is classified into three levels (A, B, and C) with A being the most critical level. If the SIAM slave host detects the failure of the SIAM master host, it will notify the system administrators within three minutes. Meanwhile, the slave host will take over the monitoring services of the master host and reorganize the monitoring system. In our experiment, the entire process took approximately twelve seconds (system failover time). Overall, the results of the three experiments show that the SIAM monitoring system is robust and it can provide real-time notifications with a low overhead.

Table 3. Simulation of the SIAM system under various execution scenarios.

Experiment	Execution Scenarios	Time (seconds)
Notification Time	Worst condition	3779
Notification Time	Best condition	180
Transfer Time of Notified Services	MSN, E-mail, SMS (average)	7
Failover Time	Level A of error message	12

5 Conclusions and future work

In this paper, we discuss how SIAM enhances the monitoring mechanisms of the SRB/iRODs system by incorporating several open source components of Nagios. We find that SIAM provides clients with more flexibility and greater control over their monitored networks because it offers both real-time notification services and a fault-tolerance mechanism. SIAM utilizes multiple communication modes when alerting administrators of system errors, all of which are customizable according to their severity. SIAM adds functionality without adding unnecessary infrastructure.

Furthermore, the SIAM system's fault-tolerance mechanism helps prevent data loss in the event of a serious system failure. The use of master and slave hosts results in lower overheads and enhanced scalability of the distributed system. Combined, these features ensure not only this minimized overhead and lower costs, but also increased robustness and alert administrators. In the future, we will explore more advanced applications of the SIAM system, including security auditing and resource performance tuning. We also plan to extend the functionality of SIAM to fit with other data transfer systems or protocols and incorporate additional cloud computing services.

6 References

[1] J. Gantz, C. Chute, Al. Mafrediz, S. Minton, D. Reinsel, W. Schlichting, A. Toncheva, "The Diverse and Exploding Digital Universe: An Updated Forecast of Worldwide Information Growth through 2011," white paper, International Data Cooperation, Framingham, MA, 2008.

[2] SRB: http://www.sdsc.edu/srb/index.php

[3] iRODS: http://www.irods.org

[4] C. Baru, R. Moore, A. Rajasekar, M. Wan, "The SDSC Storage Resource Broker," CASCON'98 Conference, Nov.30-Dec.3, 1998, Toronto, Canada.

[5] R.W. Moore, "Managing Large Distributed Data Sets Using the Storage Resource Broker," ITEA Journal of Test and Evaluation, 2007.

[6] S. Zanikolas, R. Sakellarios, "A taxonomy of grid monitoring systems," Future Generation Computer Systems, vol. 21, 2005, pp. 163-188.

[7] A. Cooke, A. Gray, W. Nut, A. Cooke, A. Gray, W. Nutt, R. Cordenonsi, R. Byrom, L. Cornwall, A. Djaoui, L. Field, S. Fisher, S. Hicks, J. Leakey, R. Middleton, A. Wilson, X. Zhu, N. Podhorszki, B. Coghlan, S. Kenny, D. O'Callaghan and J. Ryan, "The relational grid monitoring architecture: mediating information about the grid," Jounal of Grid Computing, vol. 2, Dec. 2004, pp.323-339.

[8] X. Zhang, J. Freschl, and J. Schopf, "A Performance Study of Monitoring and Information Services for Distributed Systems," In Proceedings of the 12th IEEE International Symposium on High-Performance Distributed Computing (HPDC-12), June 2003.

[9] S. Zdenko, R. Branimir, "Monitoring systems: Concepts and tools," 2004.

[10] Nagios:http://www.nagios.org

[11] Ganglia: http://ganglia.sourceforge.net.

[12] Group, T. C. CACTI: http://www.cacti.net/what_is_cacti.php

[13] Hawkeye :http://www.cs.wisc.edu/condor/hawkeye

[14] M.L. Massie, B. N. Chun, D. E. Culler, "The ganglia distributed monitoring system: design, implementation, and experience," Parallel Computing, vol. 30, 2004, pp. 817-840.

[15] S. Andreozzi, N.De Bortoli, S.Fantinel, A.Ghiselli, G.Tortone, C.Vistoli, "GridICE: a monitoring service for the Grid," Future Generation Computer Systems Journal, 2005, pp. 559-571.

[16] E. Imamagic, D. Dobrenic, "Grid infrastructure monitoring system based on Nagios," High Performance Distributed Computing Proceedings of the 2007 workshop on Grid monitoring table of contents, 2007, pp. 23-28.

[17] E. Galstad, NRPE Documentation

[18] Galstad, E., NDOUTILS Documentation Version 1.4

[19] PNP4Nagios.,Retrieved from http://docs.pnp4nagios.org/pnp-0.4/start

[20] RRDtool: http://oss.oetiker.ch/rrdtool

[21] DARB : http://rempte-backup.teldap.tw

[22] Tsung-Tai Yeh, Hsin-Wen Wei, Shin-Hao Liu, Pei-Chi Huang, Tsan-sheng Hsu, Yen-Chiu Chen, "The Development of Digital Archives Management Tools for iRODS," Proceedings of iRODS User Group Meeting 2010.

[23] TELDAP: http://www.teldap.tw/en/

Using Matrix indexes for Resource Discovery in Grid Environment

Leyli Mohammad khanli
Assistance Professor
CS Department
University of Tabriz
Tabriz, Iran

Saeed Kargar
M.S. student
Tabriz Branch
Islamic Azad University
Tabriz, Iran

Ali Kazemi Niari
M.S. student
Tabriz Branch
Islamic Azad University
Tabriz, Iran

E-mail address: l-khanli@tabrizu.ac.ir
E-mail address: saeed.kargar@gmail.com
E-mail address: a.kazemi.n@gmail.com

Abstract

There are many resources and services in grid environment which may be distributed geographically. So, resource discovery methods with lower update and discovery costs are of vital significance. The centralized methods have the advantage of improved management of nodes, but in a grid with large number of nodes and resources, a bottleneck is caused in the server, which reduces the system efficacy significantly. Decentralized methods proposed till now are not able to discover the required resources of the users and update their environment without any additional traffic and referral to unnecessary nodes.

In this paper, we use a matrix in each node to reserve the local resources and the information of children to access the requested resources of the user. In our algorithm, each node uses indexes of resources in its matrix to present available resources to the parent node. The user delivers the indexes of the resource, which contains the least bits, to one of the tree nodes. The main advantage of our method is that the unnecessary and extra nodes are not visited in both discovery and update phase; i.e. not only reduces the discovery cost but also lowers the update costs. Our proposed approach is an efficient and easy method.

The simulation results indicate that our method is the most optimal one compared to other methods considering the visited nodes in the discovery and updating as well as the lower traffic.

1. Introduction

Today, a huge number of heterogeneous computational resources which are geographically distributed can be connected using grid. In the grid environment, the numbers of users and requested resources, variety of resources, number of available nodes etc, are large, so the centralized methods [1-3] are not suitable for such environments [6].

The other methods are distributed approaches. The algorithms which use such methods face difficulties in the grid with wide environments. Flooding-based and random-based are amongst these methods. Such methods are used in most resource discovery algorithms but with large number of resources and types in the grid environment, problems increase. Some recently distributed methods use tree structures in their algorithms. They are able relieve most disadvantages available in the previous methods. "A resource discovery tree using bitmap for grids" [4] and "FRDT" [5] are amongst these methods. In the current methods, some unnecessary visited nodes cause extra traffic and reduce system efficacy in discovery or updating phase.

In this paper, like [5], we use a weighted tree for resource discovery. We use these weights as a unique path for resource discovery. Furthermore, we use a matrix in each node to reserve the local resources and the weight of children to access the requested resources of the user.

Below are presented the works done in the field of resource discovery in grid in section 2. Section 3 discusses proposed method in detail. Section 4 includes simulation results, comparing our method with previous approaches and their results. Finally, section 5 presents conclusion and further works to improve the current method.

2. Related work

In recent years, increasing computation needs have made researchers create a huge processing environment by connecting distributed resources, i.e., the current link and its infrastructure appear as a single powerful processing system with high storage capacity. It is likely that large number of heterogeneous computational resources connect to each other through personal computers to powerful clusters [7]. So, resource discovery methods are of vital importance. Several

methods have been suggested for resource discovery which will be discussed briefly.

One of these methods is matchmaking in which a matchmaking service finds a match between requests and entities. Most methods used this framework [14,15].

Yuhui Deng et al. [8] suggested a peer to peer model for resource discovery which uses ant colony optimization (ACO). The main idea of this method was inspired from ants, which search their environment for food.

Juan Li. [9] suggested a resource discovery method based on the semantic links. It also used a routing algorithm based on RDV Routing Table (RDVT).

Xue-Sheng Qi et al. [10] suggested a mechanism which is able to find multi-accessible resources and choose one of them, which uses a table. When a user wants to use the resource, reservation table will be checked. If the desired resource doesn't exist, it will be added to the table and the resource will be reserved.

Chang et al. [4] used a tree structure for resource discovery in the grid. They used several bitmaps for resource discovery and update the position of which just relates to a special attribute of a resource. User's request is transformed into the bitmaps format and delivered to the nodes available in the grid. Requested bitmap positions will be AND with local resource bitmap (containing information on the local resources of node) and if no match is found, it will be AND with bitmap related to children (index bitmap). If again no match is found, the request will be sent to all children by that node (except the children the request came from), otherwise the request is delivered to its parent.

In the previous work [5], we proposed a resource discovery method based on a weighted tree, and used formats which the information of the resources reserved as a footprint in this formats and if a request reaches to any node, we can directly access to the target node without referral to unnecessary nodes.

In this paper, like [5], we use a weighted tree and propose a matrix to reserve weights available in the tree. In the current method, we only use indexes of matrix for transmitting information between tree nodes, in which negligible information will be transformed between nodes during preparing, discovery and updating phases.

Also using weighted tree structure and with the information reserved in matrix, requested resources will be discovered without any referral to unnecessary nodes. The update cost would be lower than others.

3. Our proposed mechanism

Grid environment is a dynamic and vast area with various resources. So, the resource discovery mechanisms are of vital significance. As mentioned in previous section, there are many methods in this field. Most of them use flooding-based and random-based due to their simplicity. In these methods, we have no information about the location of resources in the environment, so requests may pass extra and unnecessary paths. They impose the traffic in the vast area of the grid, so the efficacy of the system reduces significantly.

In [4], which contains information about resources available in the grid, again user's requests are sent to unnecessary and extra nodes, because the information has no knowledge of the resource location. FRDT method [5], does not refer to unnecessary and extra nodes in its resource discovery path, but its update costs are high because no comprehensive information of children is available.

Here, we introduce a resource discovery mechanism based on a weighted tree. In the current method, a matrix is available in all nodes in which when a request reaches a node in the grid, we can directly and without referral to unnecessary nodes access the desired resource, using information registered in matrix. Through update path (if the resource used by the user), the least number of nodes are visited and the network traffic reduces significantly. The matrix is simple to build and update. When we look for a resource in the tree or have the information of a resource in matrix, just the indexes of the resource position in matrix is transferred. Thus, unnecessary traffic in the grid significantly reduces and the system efficacy increases.

3.1. Description of matrix

Various computational, communicational and informational resources can be shared in the grid, and any of these resources can have different attributes (for example attributes of operating system are Linux, Unix, Seven, XP etc). Therefore, the number and variety of the resource available in the grid would be large. Resource discovery in the grid environment would be so critical considering the definition that user has in his mind.

Suppose there are 3 kinds of resources in the grid environment and any of these resources have 4 attributes. First, a matrix placed in all nodes and its contents (regarding the resources available in nodes) will be changed during preparing and updating phases. An example of this matrix is shown in Fig. 1(a), in which the resources available in an imaginary grid written in any positions of matrix.

In Fig. 1(b), the mentioned matrix located in one of the nodes, indicates some numbers considering the resources that are shared. Number 1 in any position of matrix shows the availability of resource in current node. If the node does not have the resource (or does not exist in its children and generations) number 0 will be replaced in

the related position. Suppose that our tree in Fig. 1(b) contains Resource 1/Attribute 1, Resource 2/Attribute 2 and Resource 3/Attribute 4 where number 1 replaces them, but in the rest of positions number 0 will be placed. Other numbers may be added to them, which will be discussed in detail.

	00	01	10	11
Resource 1 (00)	Attribute 1	Attribute 2	Attribute 3	Attribute 4
Resource 2 (01)	Attribute 1	Attribute 2	Attribute 3	Attribute 4
Resource 3 (10)	Attribute 1	Attribute 2	Attribute 3	Attribute 4

(a)

	00	01	10	11
Resource 1 (00)	1	0	0	0
Resource 2 (01)	0	1	0	0
Resource 3 (10)	0	0	0	1

(b)

Figure 1. An example of matrix (a) before setting local information; (b) after setting local information.

3.2. Preparing phase

Our resource discovery algorithm is located in a weighted tree the links of which have logical relationships between the nodes; i.e. the nodes can create the links by holding the IP address of parents and children. Weights in our tree allocate a binary number to any child. Every node allocates a unique binary number to its children as in [5]. In our method, any node allocates a binary number to its children through the following formula:

$$Y=\begin{cases}\lceil \log_2^n \rceil \text{ bits} & \text{if } n>2\\ 2 \text{ bits} & \text{if } n<=2\end{cases} \quad (1)$$

As mentioned previously, our method uses a matrix in its nodes, which is available in every node connected to our grid. The node fills positions 0 and 1 according to the resources which it shares. So, only the local information of nodes will be recorded in matrixes. An example is shown in Fig. 1(b), in which the node only records their local information in matrix, and no information of the resources is available in their children. In order for our resource discovery tree to take its final form and the algorithm used by the tree, the nodes should record the information of their children and generations in their matrixes so that the requested resource of the user can be discovered.

After replacing the information of its children in matrix, the node should deliver the information to its parent node. In our proposed algorithm, every node sends the indexes of positions that contain the information (independent of how information is recorded in that place) and then sends it to its parent node in the form of several bits.

An example is given here. According to our approach (sent part), nodes send indexes of its local resources to their parent immediately. Then they send the index of matrix belonging to the children with a delay (delay must not be more than a threshold considered for the sent part). Every node in the receiving part, first records information related to its children and then its generations. During the resource discovery, the request is sent to paths which are close to the node. Node G and H (being leaf nodes), have only the information of their local resources. Node G, according to Fig. 2, contains Resource 1/Attribute 3, Resource 2/Attribute 1 and Resource 3/Attribute 2. So, it sends the information of the indexes of these resources ((00,10),(01,00),(10,01)) to its parent node i.e. D. In node H, the related indexes will be sent to node D. Node D writes the related weights in their related positions, considering which child owns the weight.

For example, node A in the position of (00,00), because it contains the local resource (bit 1), attaches the weight of the children that received the indexes from them (i.e. 10010). In Fig. 2, we show a tree after gathering information of the children from their parent nodes.

3.3. Resource discovery in proposed scheme

When a user requests a resource in the grid environment, it should be delivered in a particular format to one of the nodes. In our proposed approach, this format will be determined considering the size of matrix and extra one bit for the defining of discovery (bit 1 would be discovery and bit 0 would be update).

If the kind of resources in our grid is 12, so according to Fig. 1(a), our matrix contains three rows and four columns. So, in discovery, to determine the type of resource, we require 2 bits for determining the row and 2 bits for columns and another bit for determining the resource discovery request. The one bit adds the right of these bits of the resource discovery which would be 1 (i.e. (row, column, 1)).

When a request in the format of (r,c,1) reaches any node in tree, observing number 1 in the right end of the received form, means that the received form would be a resource discovery request. Therefore, immediately refers to its matrix, looking for information in index (r,c) of own matrix, first the contents of the related position are separated, and in the size of Y bits (formula 1) the information is separated from right to left. If a bit 1 remains in far end, it means that the node locally contains current resource, then the node reserves the requested resource and sends a success message to a user that has sent this request.

Otherwise, if no local resource exists, it removes Y bits through left to right (weight of one of the children), then sends the resource discovery request to the related child.

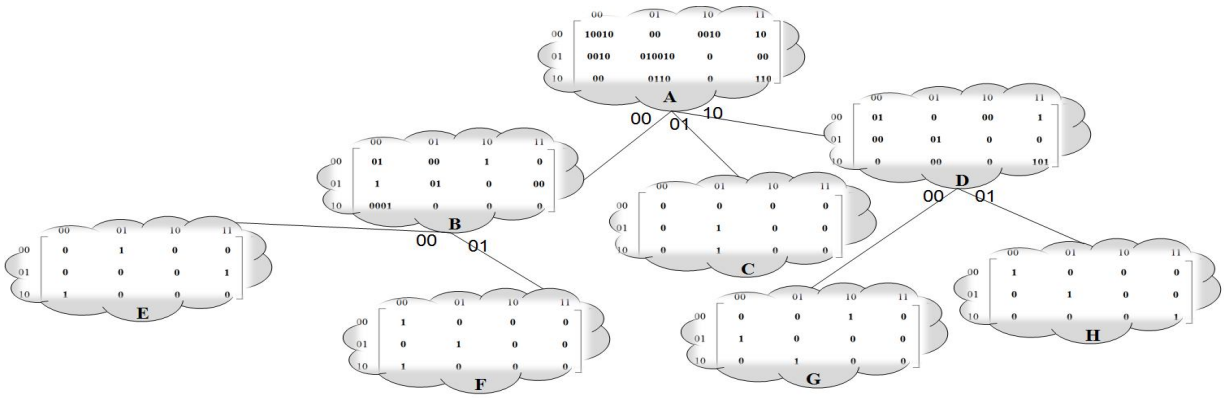


Figure 2. A sample of tree after preparing phase.

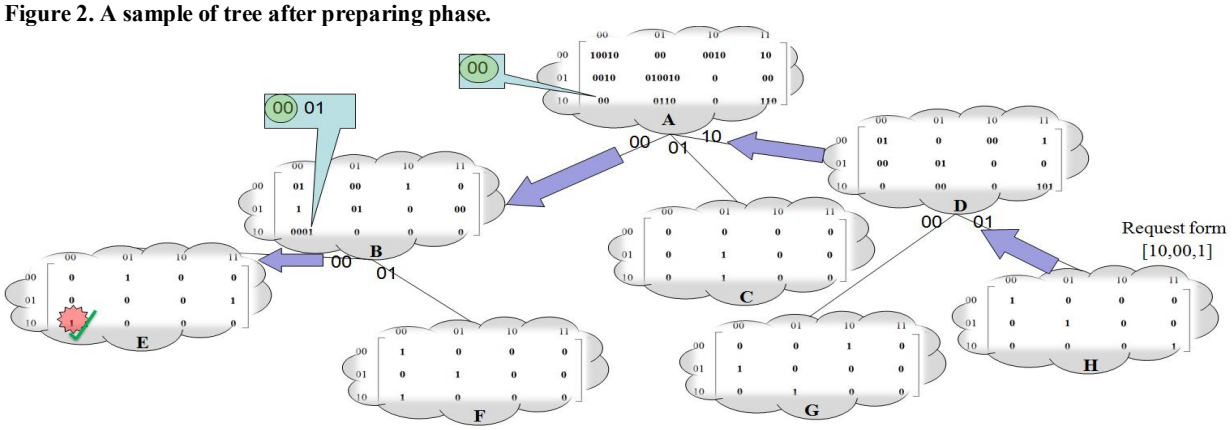


Figure 3. An example of resource discovery.

In the third state, if the position contains 0, it means that the desired resource does not exist in the node nor the children and generation. So, the request will be sent to the parent node. We show an example of resource discovery in Fig. 3, which the requested resource discovered for user in node E.

As we observed, the correct path to reach any resources in the tree can be found without keeping a lot of information for every resource in the node. For every node and for any resource, only one binary number with short length, which is the weight of one or more children, is kept. Therefore, we can directly reach to the target node without visit extra and unnecessary nodes.

Preventing the large volume of reserved data in matrix, every node keeps only one step of the path to reach the target. For example, when a resource has ten level distances from a node, only it keeps one step, instead of keeping the whole path in mind, which is the weight of one edge of its children. In fact, the path to a particular resource is distributary reserved in the tree, so the volume of the information in every node is reduced, and the data management would be facilitated. One of the other abilities of our method is that every node can send the request to several different paths (knowing that the resource is available in some of its children) and finds a resource from different places and reserves it. Therefore, the user chooses and uses one of the resources.

3.4. Updating

Update phase is part of the resource discovery phase which both should be considered as a single phase (discovery & update phase). When a resource in the grid environment is discovered and made accessible to the user, it will not be usable for others until current user stops using it. As in updating the resource discovery, when a change occurs in one of the resources of the node, it should be informed to its parent in a special format, and the required changes are done by the parent node. The size of the format will be determined considering the size of matrix, and an extra one bit for the defining of the update operation (bit 0 will be added to the right hand of the format).

If a resource is removed from the list of that node, the mentioned node in its matrix deletes bit 1 from the related position to that resource. When no information is available in that position (i.e. bit 0), the node sends the index of the place in the update form to its parent.

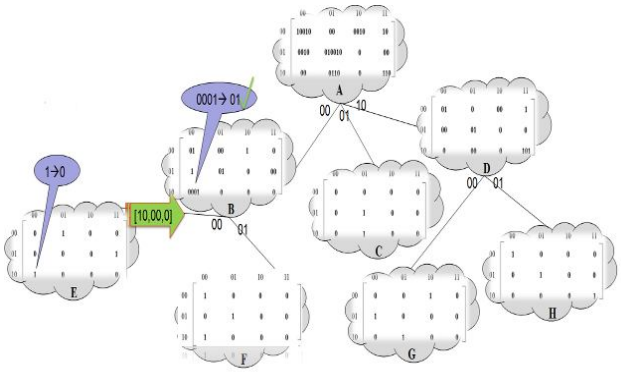


Figure 4. An example of update.

Otherwise, if no information is available (weight of edge of children), updating will finish in this step, and there will be no need for sending the update form to the parent. If any node in our resource discovery tree receives a form in the format of (r, c, 0), it refers to the index (r, c) in its matrix and deletes the weight from index (r, c) of matrix which the update form has received from it. If any number remains except 0, update will finish, otherwise, update form will be sent to the parent node.

As shown in Fig. 4, suppose that a resource with index (10,00) is lost in node E with any reason, so the node deletes the bit 1 in the related index in its matrix. Since the result is 0, the node sends the update form to its parent (B). Receiving the form, node B deletes 00 from index (10,00) in its matrix (since the update form is received from child with weight 00). Because the result is a number except 0 (0001→01), updating phase finishes. Another condition is that an attribute of the resource changes (size of disk space changes from 160G to 120G) or a new resource is added. Here, the node should send the previous attribute (160G) in updating format to its parent node so that the resource could be deleted from its information. Then, new added resource (120G) informs to the parent node. Our updating algorithm is simple and low cost.

4. Simulation results

The simulations were done in MATLAB environment, in which different nodes of the tree are placed in their position and then the available resources were randomly distributed in every level by requesting randomly in different levels. We performed our experiment on trees with height of 4 as in [4, 5, 11], where we make our grid similar to a real world grid system.

In the first experimental tests, number of nodes that were visited in the resource discovery in our method, FRDT [5] and tree method [4] was compared with different nodes in a tree with the height of 4. Fig. 5 shows the results of this experiment.

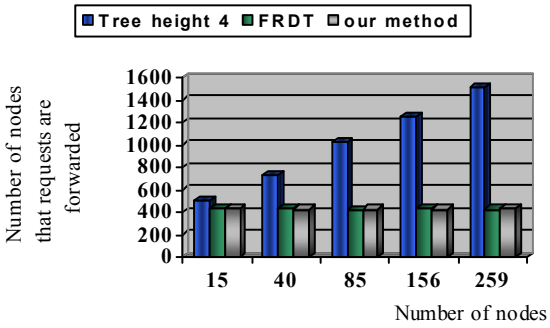


Figure 5. The number of nodes visited in the resource discovery trees with height 4 for 100 requests.

In the second experimental tests, we compared four methods of flooding-based, MMO [12,13], tree method and FRDT with ours, in which the number of visited nodes were in trees with the height of 4. As shown in Fig. 6, our method would have a constant value (since the tree height is constant).

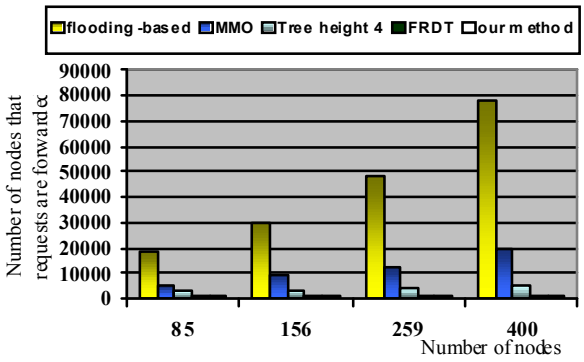


Figure 6. Average number of nodes that requests are forwarded to during resource discovery in trees with height 4 for 300 requests using different method.

In the next experimental tests, we compared the update of our method with tree method and FRDT in which the visited nodes are shown in trees with the height 4 to update the tree following 1000 requests sent to the resource discovery. The cost of our update method is lower than others (Fig. 7).

In the fourth experimental tests, the total number of nodes visited for the resource discovery and tree updating are shown for three methods. As shown in Fig. 8, the total number of nodes visited in our method would be lower compared to other methods. So, our method would be more efficient than other ones.

In the last experimental tests, the number of links occupied during the resource discovery and update in three methods shown for 1000 requests. In Fig. 9, the number of links occupied in our method is lower than other methods and produces lower traffic.

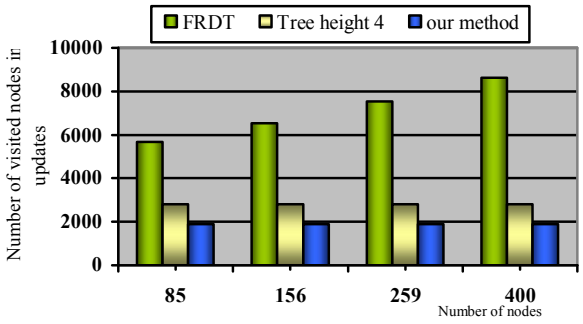


Figure 7. The number of nodes visited in updating in trees with heights 4 for 1000 requests.

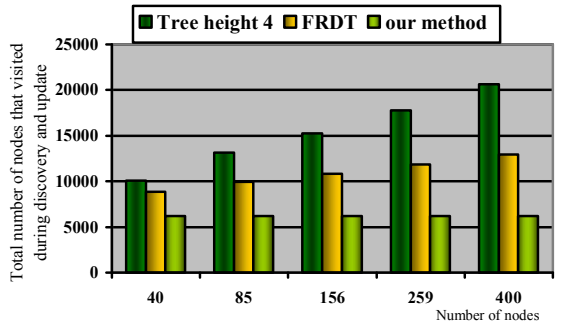


Figure 8. Total number of nodes visited in the resource discovery and updating for 1000 requests on trees with height 4.

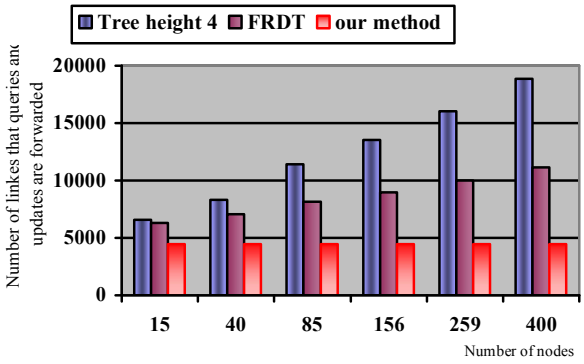


Figure 9. The number of links occupied during the resource discovery and updating for 1000 requests.

5. Conclusions and future work

In this paper, we proposed a mechanism for resource discovery and updating which used matrix indexes, in fact, the indexes with much lower volumes were transmitted amongst nodes. In the current method, we will be able to discover and reserve multi-resources, and the user can choose and use the best amongst the reserved resources. The main advantage of our method is that the unnecessary and extra nodes are not visited in both discovery and update phase; i.e. not only reduces the discovery cost but also lowers the update costs.

In the future works, we should improve our method such that the user could request multi-resources with multi-attributes simultaneously in a single format and lower space and the mechanism discovers the resources from one node and make it accessible to the user. If we could consider other factors like resource cost besides the resource interval in our algorithm, our method will significantly improve.

6. References

[1] F. Berman, et al., Adaptive computing on the grid using AppLeS, TPDS 14 (4) (2003).
[2] A. Chien, B. Calder, S. Elbert, K. Bhatia, Entropia: Architecture and performance of an enterprise desktop grid system, J. Parallel Distrib. Comput. 63 (5) (2003).
[3] C. Germain, V. Neri, G. Fedak, F. Cappello, XtremWeb: Building an experimental platform for global computing, in: Proc. of IEEE/ACM Grid, December 2000.
[4] R-.S .Chang, M-.S .Hu, A resource discovery tree using bitmap for grids, Future Generation Computer Systems 26 (2010) 29-37.
[5] L.M. Khanli, S. Kargar, FRDT: Footprint Resource Discovery Tree for grids, Future Gener. Comput. Syst. 27 (2011) 148–156.
[6] K. Czajkowski, S. Fitzgerald, I. Forster, C. Kesselman, Grid information services for distributed resource sharing, in: Proc. 10th IEEE International Symposium on High-Performance Distributed Computing, HPDC-10, August, 2001, pp. 181_194.
[7] Ian Foster, Carl Kesselman, The Grid 2: Blueprint for a New Computing Infrastructure, Morgan Kaufmann Publishers Inc., San Francisco, CA, 2003.
[8] Yuhui Deng, FrankWang, Adrian Ciura, Ant colony optimization inspired resource discovery in P2P Grid systems, J Supercomput (2009) 49: 4–21
[9] J. Li, Grid resource discovery based on semantically linked virtual organizations, Future Gener. Comput. Syst. 26 (2010) 361–373.
[10] X.S. Qi, K.L. Li, F.J. Yao, A time-to-live based multi-resource reservation algorithm on resource discovery in Grid environment, in: Proceedings of the 2006 1st International Symposium on Pervasive Computing and Applications, August 3_5, 2006, pp. 189_193.
[11] C. Mastroianni, D. Talia, O. Versta, Evaluating resource discovery protocols for hierarchical and super-peer grid information systems, in: Proceedings of the 15th EUROMICRO International Conference on Parallel, Distributed and Network-Based Processing, PDP'07, February 7–9, 2007, pp. 147–154.
[12] M. Marzolla, M. Mordacchini, S. Orlando, Resource discovery in a dynamic environment, in: Proceedings of the 16th International Workshop on Database and Expert Systems Applications, DEXA'05, September 3–7, 2005, pp. 356–360.
[13] M. Marzolla, M. Mordacchini, S. Orlando, Peer-to-peer systems for discovering resources in a dynamic grid, Parallel Comput. 33 (4–5) (2007) 339–358.
[14] K.I. Karaoglanoglou, H.D. Karatza, Resource Discovery in a dynamical grid based on Re-routing Tables, Simulation Modelling Practice and Theory 16 (2008) 704–720.
[15] Simone A. Ludwig, S.M.S. Reyhani, Introduction of semantic matchmaking to Grid computing, J. Parallel Distrib. Comput. 65 (2005) 1533 – 1541.

Minimization of Security Alerts under Denial of Service Attacks in Grid Computing Networks

Syed Raheel Hassan[1], **Jasmina Pazardzievska**[2], **and Julien Bourgeois**[1]
[1]Laboratory of Computer Science, University of Franche-Comte (UFC), Leprince-Ringuet, Montbeliard, France
[2]Faculty of Electrical Engineering. and IT, University Ss. Cyril and Methodius, Skopje, Republic of Macedonia

Abstract— *Grid computing networks aggregate huge computing power that they need for solving different scientific problems. This power can be used for attacking the grid's components as well as outside computers. Attacks such as the Denial of Service (DoS) could be used to target user machines, servers, and security management solutions to sabotage the normal operations of the grid computing network. In this paper the design of the grid SOC (GSOC) which minimizes the huge security alerts generated under network attacks will be discussed. GSOC performance has been compared with the DSOC and its attack detection capabilities with Snort and some experiments are presented using Grid'5000 network.*

Keywords: GSOC, DoS in Grid Computing Networks, Minimization of Security Alerts using GSOC.

1. Introduction

In recent years, different multi-administrative domains started working together as one grid network. The emergence of different organizations has made the grid computing network vulnerable to many network attacks. Due to the nature of the grid an attacker can use the grid computational power to target any administrative domain attached to the grid network for example Distributed Denial of Service (DDoS) attacks. When an attacker launches an intensive DDoS attack on the network the IDS starts generating many security alerts. It starts sending these alerts to the central database. This huge number of security alerts can create bottlenecks in the network and uses lots of disk space. Due to these intensive attacks the IDS can becomes so overloaded and therefore turning unstable. This instability results in the creation of many security alerts or some-times in false positives. Both the instability and the huge number of security alerts that administrator has to manage, give the attacker a fair chance to perform malicious activities. The instability of an IDS is due to multiple reasons. The most common ones observed are due to disk space failure, database failure and system process queue overloading. Intrusion detection and prevention systems (IDPSs) have been introduced to help the network administrator thwart possible network attacks. At present, IDPSs are also struggling to efficiently protect multi-administrative domain networks which can change their size dynamically. They hardly achieve this goal and reducing number of false positives while maintaining performance is still an issue [1]. The remaining parts of the paper are organized as section 2 presents the related work, in section 3 the architecture of GSOC has been discussed along with its components. Experiments are to be found in section 4 and the conclusion in section 5 .

2. Related Work

Protect grids from DDoS Attacks by Yang Xiang and Wanlei Zhou [2] proposed a distributed defense system for detecting DDoS attacks. This system requires access to the routers of each site. They performed the tests on the SSFNet (Scalable Simulation Framework) [3] which allowed them to capture and analyze all the network traffic between different sites. Their solution lacks practical implications because the access to routers and the capturing of network traffic of external sites is not possible in real active grid networks.

Security for Grid Service by Von Welch et al. [4] was the work done to upgrade the Globus Toolkit version 2 (GT2) so that it became the Globus Toolkit version 3 (GT3). It was the first implementation of the Open Grid Service Architecture (OGSA). OGSA was first suggested by Ian Foster in [5]. The OGSA provides heterogeneous systems with interoperability in order to communicate with different types of resources. The technical documentation of the OGSA 1.5 version which is available at [6] recommends to use intrusion detection systems for handling DDoS attacks on grid services. The OGSA does not provide any mechanism with how to counter DDoS attacks from the trusted user.

Predation and the cost of replication: New approaches to malware prevention by Richard Ford et al. [7] have used a ++shield program which was a modified version of the shield program. The Shield was developed by Wang et al. [8] It limits Malicious Mobile Code (MMC) in the network. In their experiments of shield heuristic simulation they have used the improved version of shield that was installed by default in all machines. If any machine has been attacked, the victim machine blocks the attack attempts by returning a magic number into the TCP headers or the

packet payload. This technique was useful to overcome DoS attacks but could not handle DDoS attacks. The DDoS uses multiple sources such as the attacker with mock IP addresses. Therefore even if the attacked machine keeps blocking the requests, it cannot handle DDoS attacks.

Distributed Security Operation Center (DSOC) was proposed by Ganame et al. [9]. It shows better stability in multi-site networks by detecting DDoS attacks. When the DSOC was deployed in the grid computing networks, it did not give consistent results. It does not handle the grid specific properties namely, (i) The grid network, a combination of different administrative domains, each of them composed of multi-site networks. (ii) In grid network a high number of nodes collaborate with one another. Therefore the size of the network is increasing and decreasing dynamically. (iii) In grid network a view of the security events of external networks is unavailable. (iv) The DSOC under DDoS attacks in a grid network needs much more disk space as it does not have time-based correlation modules.

Keeping the above-mentioned issues in view, the GSOC has been proposed by Bourgeois and Hassan [10]. It overcomes the limitation of the DSOC. The GSOC has two levels of correlation namely, basic and advance which help the GSOC to detect more sophisticated and distributed attacks. Due to this two-step correlation, the GSOC reduces the size of logs at both the collector and the analyzing ends. The aim of our work is to develop a security operation center dedicated to multi-administrative domain networks.

3. Grid Security Operation Center (GSOC) Architecture

In this section the components of the GSOC are explained with the importance of correlation in detecting complex attacks. The GSOC is based on the concept of separate boxes [11] that perform a specific task. The GSOC has four main components which are an event-generating box (EBox), a collecting box (CBox), a Local Analyzer (LA) which consists of a database box (DBox) and an alert-analyzing box (ABox) and a Global Analyzer which contains a global intrusion data base called (gidb).

3.1 Correlation

The main purpose of correlation is to analyze complex information sequences and to produce simple, synthesized, real-time alerts. The GSOC introduces two-level correlations: (i) Basic Correlation (BC) and (ii) Advanced Correlation (AC). This two-level hierarchy reduces the network traffic between the GSOC components and causes an easier detection of complex intrusions. The CBox has the role of performing basic correlation, whereas the LA is responsible for advanced correlation. The main purpose of BC is to reduce the network load between the GSOC modules; therefore attack detection is easier to perform. BC does not have the ability of detecting distributed denial of service or strong brute force attacks. The CBox is capable of detecting only Weak attacks. For example, if two attackers are simultaneously attacking one target sensor in an AD, performing a DDoS or strong brute force attack, the CBox will report one alarm for a DoS attack and one alarm for a weak brute force attack, originating from two different attackers. The task of deciding whether it is a DDoS attack or any other kind of strong attack is dedicated to the LA, more specifically to the ABox.

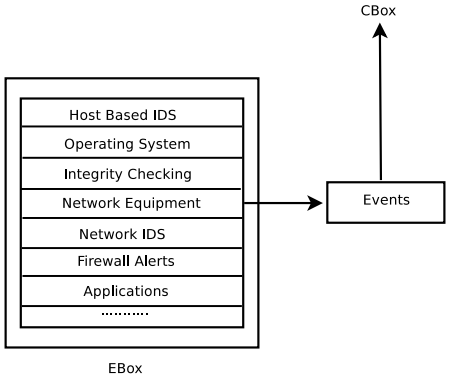


Fig. 1: EBox Design

3.2 Event-Generating Box (EBox)

The EBox is a component in grid network that generates events (see figure 1). These events could be of two types. One from the sensors which generates data due to any operation performed on them, this includes operating systems, firewalls, routers, switches, wireless HUBs or RADIUS servers. The second type generates events when a specific state or a threshold value occurs in different network management systems (NMSs). These NMSs are very useful for detecting distributed denial of service attacks by continuous checking system availability via ping or snmp [12]. These events are then forwarded to the CBox.

3.3 Collecting Box (CBox)

The CBox is a log-collecting module that collects logs from different EBoxes. One CBox is enough for one local site of an administrative domain. More than one CBox can be deployed in one site if the number of generated events are too high. Every EBox has a different format for reporting the event. Therefore the CBox collects this raw information from different protocols shown in figure 3. The dispatcher that plays an intermediary role is placed between the event-receiving protocols and the application modules. The application modules are the modules in the CBox which contains the possible attack lists. The dispatcher searches for these reported events from the EBox and tries to match them within available application modules like Linux, Windows and XtreemOS. When the reported event matches

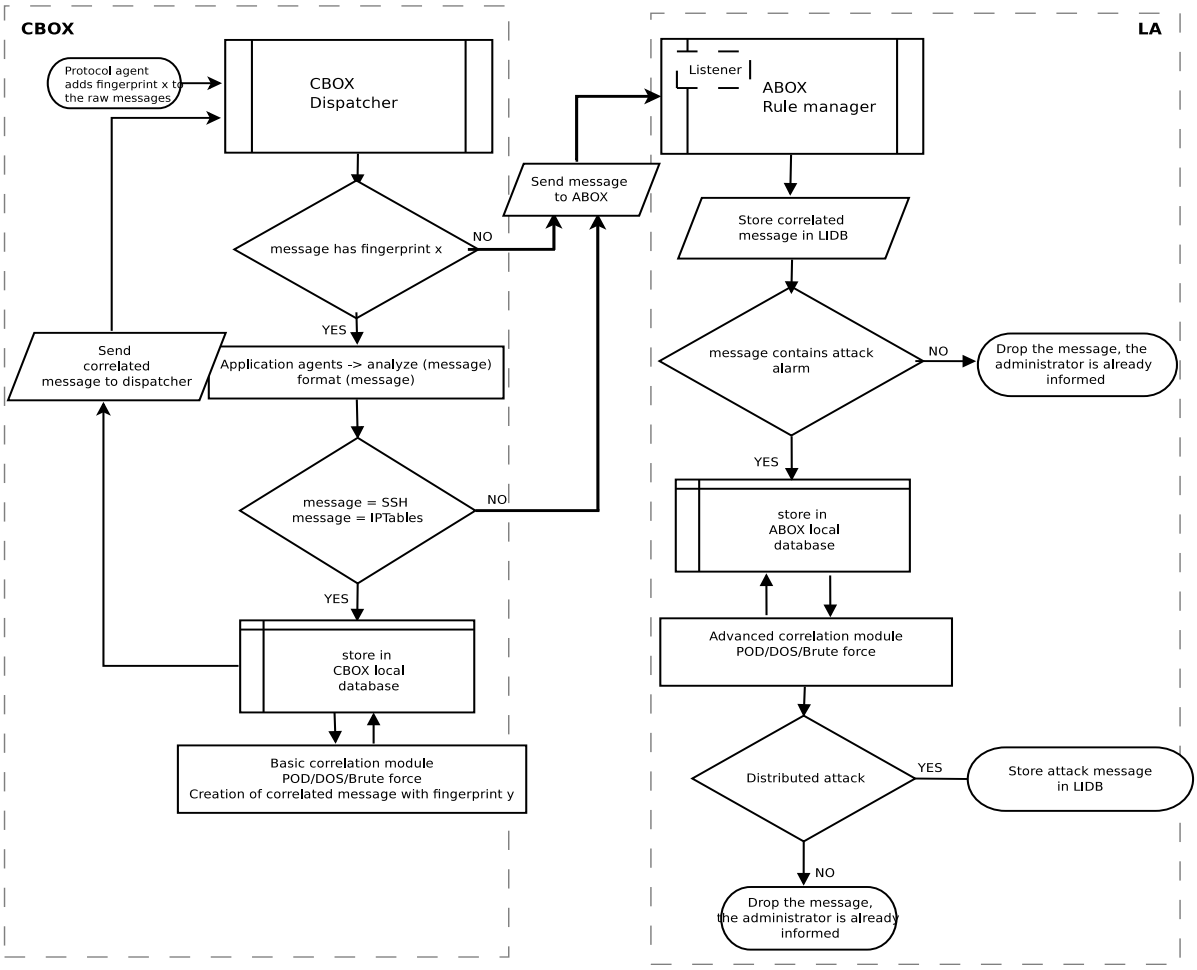


Fig. 2: Basic and Advanced Correlation flow chart

any defined attack template, it is then arranged in an internal format before it is sent to Basic Correlation.

3.3.1 Basic Correlation

The left part of figure 2 is the detailed explanation of basic correlation. The basic correlation module can be thought as a message marker. Each message is labeled depending on its contents, it checks if a message is containing an attack alert or it is a regular message. Each raw message sent from the EBoxes and received by the event-receiving protocols at the CBox is labeled with fingerprint (eg: fingerprint x). This fingerprint points out to the dispatcher that this message should be first analyzed by the application modules and if supported rule is found then it will be formatted. The dispatcher inspects whether these formatted messages are the ones that the administrator is interested in correlating (message originating from ssh session or message from IPtable rules at the EBoxes). If this condition is true, these kinds of messages are stored in a local database for a very short period of time (at most one minute). If this condition is not true, the CBox forwards that message to the LA (specifically ABox) in order to display a global view of the whole network. After the basic correlation new fingerprint (eg: fingerprint y), different than the one added to the raw messages is applied to the stored messages in the local database. This fingerprint tells the dispatcher that this message has already correlated and should be sent to the LA for further analysis. Afterwards, only the correlated messages are stored in the local database and transferred to the dispatcher which further forwards them to the LA. At this stage the messages that contain an attack are forwarded to the LA as well as those that are not containing any attacks. The communication between the CBox and the LA is over socket protocol.

3.4 Local Analyzer (LA)

The Local Analyzer is composed of two modules (i)alert analyzing box (ABox) and (ii)database box (DBox) (see figure 4). The *ABox* job is to report the alerts of the messages received from the CBox. All the CBoxes from the multiple

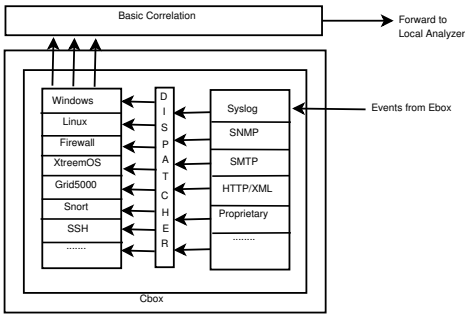


Fig. 3: CBox Design

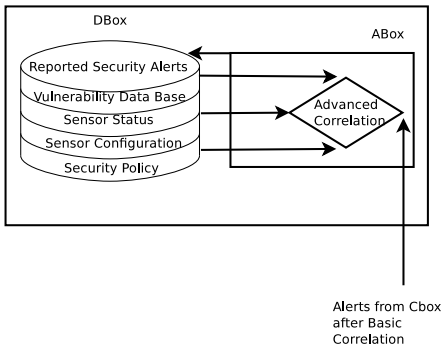


Fig. 4: DBox and ABox Design

local sites of an administrative domain send their alerts after basic correlation to the ABox. The ABox then receives these alerts and further correlates for strong brute force, strong ping of death and distributed denial of service attacks. The ABox warns the grid administrator with low, medium, and high-level alerts. These three types of alerts are created by the administrator using the GUI of the GSOC. These alerts are then saved in the DBox. The *DBox* holds information like *Security Policy* which contains all the rules created by an administrator for example password cracking attempts, administrative rights gaining attempts, log erasion etc. *Sensor Configuration* which holds all the information related to a node, for example what type of operating system is used on a node, its kernel version number, which services are running. *Sensor Status* shows whether the node is working or not. *Vulnerability Database* which holds vulnerability from common vulnerabilities and exposures [13]. *Reported Security Alerts* are the alerts which are identified as attacks and these alerts are saved permanently in the database.

3.4.1 Advance Correlation

The right side of figure 2 explains the advanced correlation at the LA, more specifically the ABox. When the CBox starts sending messages to the ABox, the listener module at the ABox accepts the correlated messages from the CBox. When a correlated message arrives at the ABox, a rule manager checks if the network administrator is interested in monitoring information about the sensor included in the message. If this is true, the message is stored in the lidb database and reported to the administrator. However, the administrator still does not know whether there is a strong attack on any of the sensor. For this reason, the messages that contain an alarm for an attack in itself are also stored in a aboxlocal database for a short period of time (at most one minute, just like the local database at the CBox) until the advanced correlation finishes its task. The messages without alarms are dropped, because the administrator has already been informed. The operations for performing the advanced correlation task are (i) target and (ii) time correlation. This module counts the number of notifications for the same target from different sources (attackers) within the time interval. If there are more than one attackers that assaults the same target (sensor) in the same unit of time, a strong attack alert is stored in the lidb database. At last one alert will be displayed at the GUI of the GSOC.

3.5 Global Analyzer (GA)

The Global analyzer is the backup of the LA and lidb. It starts working if the LA and lidb are under an intensive distributed denial of service attack and if the LA stops processing security alerts from CBoxes.

4. Experiments

In this section the comparison of the GSOC with Snort and the DSOC under different network attacks has been discussed. Here Snort has been taken to measure the attack detection capability of the GSOC. In practical Snort could be used as an input to the GSOC. The DSOC has been taken to measure the performance in terms of number of alerts generated when the victim is under attack. Graphical representations have been used to show the efficiency of the GSOC in relation to others. To calculate the efficiency the number of alerts generated individually by the GSOC, DSOC and Snort in one hour has been taken as a parameter. Figure 5 is the general diagram of Grid'5000 network where the GSOC was deployed for the experiments, the details of the network are available at [14]. The deployment of GSOC for experiments was first a CBox at Orsay, a second one at Grenoble and a third CBox at the Rennes site. These three CBoxes send their logs to the LA when the two attackers simultaneously start attacking the victim machine at the Rennes site. The security alerts of the victim machine were forwarded to the LA at the Nancy site where the administrator could take action to block the attackers. In the experiments some of the log messages sent from the victim machine to the CBox machine were dropped due to network congestion because the UDP protocol had been used for sending and receiving the logs via rsyslog.

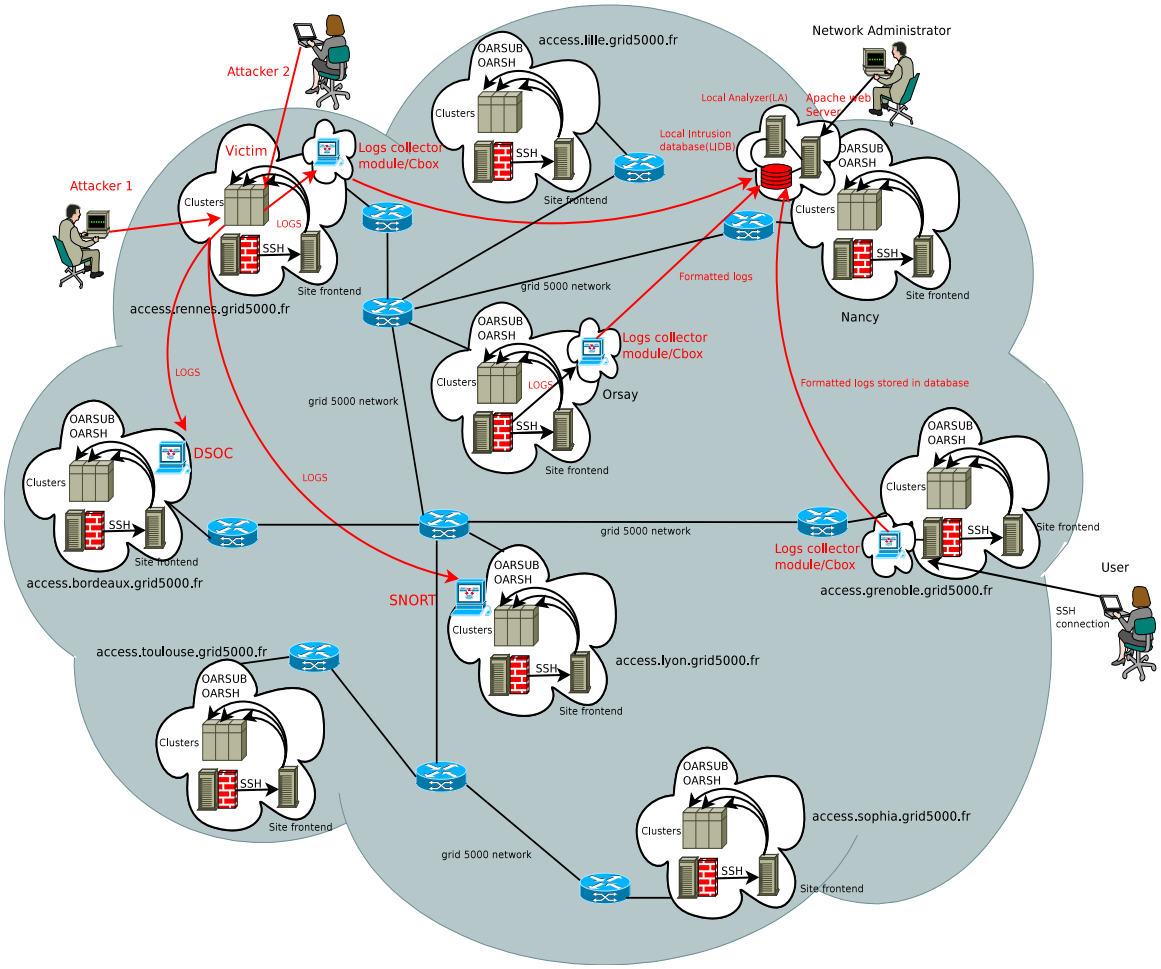


Fig. 5: GSOC in Grid'5000 Network

4.1 GSOC Behavior Under Brute Force Attack

This is a brute force attack test scenario for the GSOC. The THC Hydra [15] has been used for launching a brute force attack. It has used a password file, which is a dictionary of passwords. This dictionary contains 8048 passwords in one file. In this test, two attackers (attacker1 and attacker2) are performing the attack (see figure 5) on a target machine called victim. The GSOC detects the attack and generates one alert after one minute which has been shown on the GUI of the GSOC as a *weak attack*. This one alert contains all the necessary information which includes the IP address of the sources, the start and end time of the attack which is equal to the elapsed time of one minute, the user's name attacker 1 or attacker 2 by which the attack has been launched, target IP addresses and the number of attempts made by each attacker. This information is very helpful for the network administrator to stop the expansion of attacks. This is the role of basic correlation that has been added to the GSOC which minimizes the log messages to save disk space, minimizes database size, minimizes network bandwidth before sending these correlated alerts to the LA. The behavior of GSOC under brute force attack can be seen in figure 6. The LA receives the correlated messages from multiple CBoxes and further correlates them to see if the other sensors from other sites are also targeting the same sensor or a group of sensors. If this is the case then another alert is generated and displayed at the GUI of the GSOC which shows *strong attack*. This alert also contains the similar information mentioned above which helps the administrator to look further into other sites to detect the source of the attack.

4.2 Comparison of GSOC with Snort and DSOC Under Brute Force Attack

Figure 7 shows that the GSOC is much better than the DSOC and Snort. GSOC basic and advance correlation has reduced the number of generated alerts while keeping the same information intact. The GSOC has reduced the disk space, the database size and the network bandwidth; in addition, it can detect more sophisticated attacks.

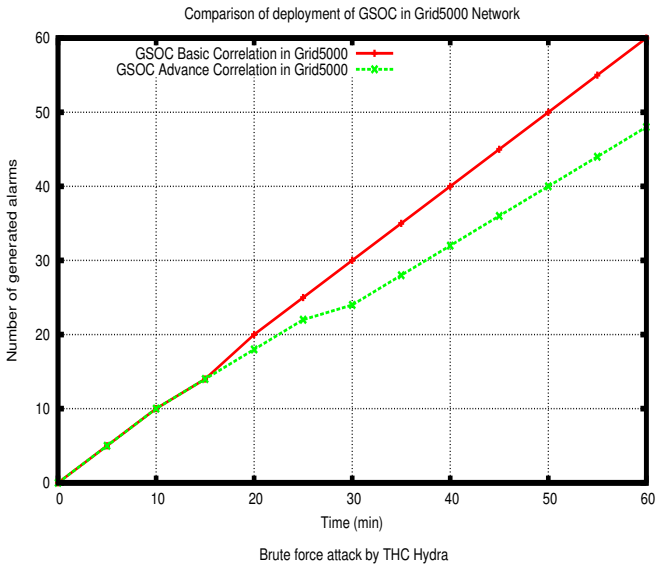


Fig. 6: GSOC behavior under brute force attack

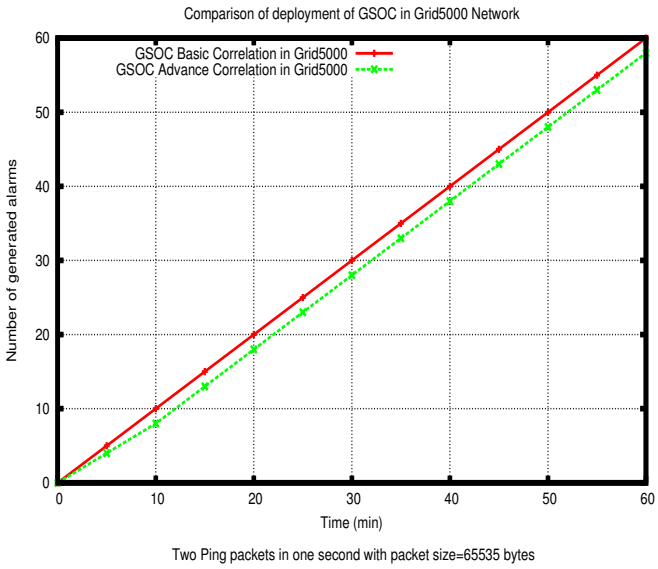


Fig. 8: GSOC behavior under Ping of Death Attack

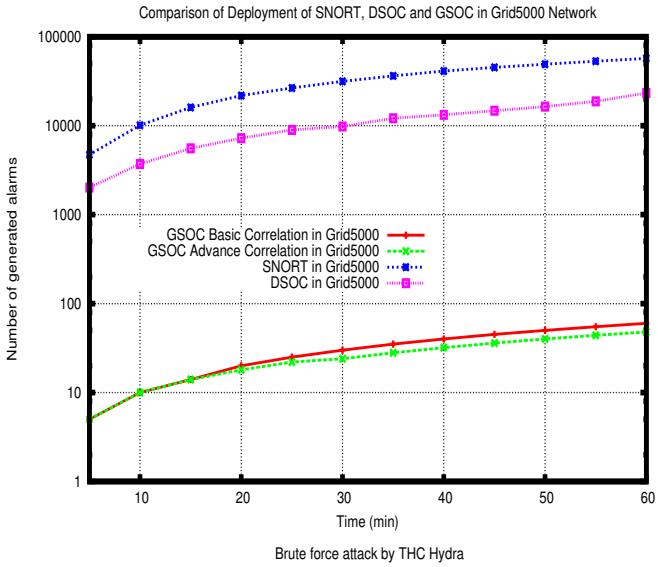


Fig. 7: GSOC v/s Snort and DSOC under brute force attack

4.3 GSOC Behavior Under Ping of Death Attack

This is a ping of death (PoD) attack test scenario for the GSOC. In order to detect a DDoS/DoS attack, ping packets bigger than (the size) of 85 bytes will be discarded. IPtable rules have been used to log an alert if any packet bigger than 85 bytes has been received by the sensor. In the code of the GSOC when the CBox script executes the IPtables, rules have been added automatically . The IPtable rules that have been used are as follows:

Iptables -A INPUT -d0/0 -s0/0 -p icmp -m length -length 85: -j LOG -log-prefix "PING OF DEATH"
Iptables -A Iptables -A INPUT -d0/0 -s0/0 -p icmp -m length -length 85: -j DROP.

This means that each ICMP packet greater than 85 bytes will be reported. After (performing) the attack using the commands mentioned below,

Attacker 1: ping -i 0.5 -s 65507 IP Address of the Victim
Attacker 2: ping -i 0.5 -s 65507 IP Address of the Victim
the results can be seen in figure 8.

4.4 Comparison of GSOC with Snort and DSOC Under Ping of Death Attack

The comparison clearly shows the performance of the Snort, DSOC and GSOC. The GSOC is much better in terms of generating lesser number of alerts. The alerts that are stored in the local database at the CBox are deleted after one minute and only the correlated messages are stored locally and transferred to the LA. This helps to control the size of the disk and database. (See figure 9).

5. Conclusion

To counter attacks like brute force, denial of service and distributed denial of service which have been discussed in this paper, the GSOC has generated few alerts compared to the DSOC. The GSOC minimizes and correlates security alerts and gives the administrator a concise and accurate security report. The results in comparison to the DSOC and the Snort are presented in section 4. The graphs in the

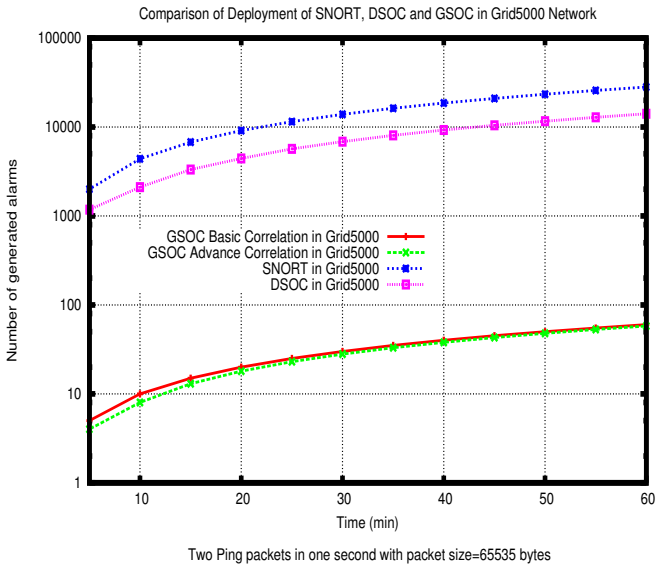


Fig. 9: GSOC v/s Snort and DSOC under Ping of Death Attack

experiments show that the GSOC generates accurate security alerts after correlating a comprehensive event record from one or multiple attackers. This correlation of security alerts makes the GSOC resistant to intensive distributed denial of service and brute force attacks.

Acknowledgments

Thanks to the Laboratory of Computer Science University of Franche-Comte France, the Higher Education Commission and Quaid-e-Awam University of Engineering, Sciences and Technology Pakistan for supporting our work financially. The Grid'5000 network for providing us with the platform to perform tests.

References

[1] W. Kanoun, N. Cuppens-Boulahia, F. Cuppens, S. Dubus, and A. Martin, "Success likelihood of ongoing attacks for intrusion detection and response systems," in *Proceedings of the 2009 International Conference on Computational Science and Engineering - Volume 03*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 83–91. [Online]. http://portal.acm.org/citation.cfm?id=1632709.1633494

[2] Y. Xiang and W. Zhou, "Protect grids from ddos attacks," in *GCC*, 2004, pp. 309–316.

[3] "Scalable simulation framework (ssf): A public-domain standard for discrete-event simulation of large, complex systems in java and c++," 2010. [Online]. http://www.ssfnet.org/homePage.html

[4] V. Welch, J. Gawor, C. Kesselman, S. Meder, and L. Pearlman, "Security for grid services," in *In Twelfth International Symposium on High Performance Distributed Computing (HPDC-12)*. IEEE Press, 2003, pp. 48–57.

[5] I. Foster, C. Kesselman, J. M.Nick, and S. Tuecke, "The physiology of the grid: An open grid services architecture for distributed systems integration," 2002. [Online]. http://www.globus.org/alliance/publications/papers/ogsa.pdf

[6] "The open grid services architecture, version 1.5," Available from: http://www.ogf.org/documents/GFD.80.pdf, 2002-2006.

[7] R. Ford, M. Bush, and A. Bulatov, "Predation and the cost of replication: New approaches to malware prevention?" *Computers & Security*, vol. 25, no. 4, pp. 257–264, 2006.

[8] H. J. Wang, C. Guo, D. R. Simon, and A. Zugenmaier, "Shield: vulnerability-driven network filters for preventing known vulnerability exploits," *SIGCOMM Comput. Commun. Rev.*, vol. 34, pp. 193–204, August 2004. [Online]. http://doi.acm.org/10.1145/1030194.1015489

[9] A. K. Ganame, J. Bourgeois, R. Bidou, and F. Spies, "A global security architecture for intrusion detection on computer networks," *Computers & Security*, vol. 27, no. 1-2, pp. 30–47, 2008. [Online]. http://dx.doi.org/10.1016/j.cose.2008.03.004

[10] J. Bourgeois and S. R. Hassan, "Managing security of grid architecture with a grid security operation center." in *SECRYPT'09, Int. Conf. on Security and Cryptography, Milan, Italy*. INSTICC Press, July 2009, pp. 403–408.

[11] S. Northcutt and J. Novak, *Network Intrusion Detection*, third edition ed., ser. ISBN: 0-73571-265-4. New Riders, 2002, september.

[12] J. Bourgeois, R. Bidou, and F. Spies, "Towards a global security architecture for intrusion detection and reaction management," in *Proc. of the 4th Int. Ws. on Information Security Applications, WISA 2003*, ser. LNCS, K. Chae and M. Yung, Eds., vol. 2908, Jeju, Corea, August 2003, pp. 129–142.

[13] "Common vulnerabilities and exposures is a dictionary of publicly known information security vulnerabilities and exposures," 2010. [Online]. http://cve.mitre.org/

[14] "Grid'5000 is a scientific instrument for the study of large scale parallel and distributed systems." 2010. [Online]. https://www.grid5000.fr/mediawiki/index.php/Grid5000:Home

[15] V. Hauser, "The hacker's choice, a very fast network logon cracker which support many different services," 2010. [Online]. http://freeworld.thc.org/

SESSION

GRID AND CLOUD: APPLICATIONS + ALGORITHMS

Chair(s)

TBA

On-Demand Virtual Cluster in Cloud WebOS with Fault Tolerance Module

Yi-Lun Pan [1], Chang-Hsing Wu[1], Hsi-En Yu[1], Hui-Shan Chen[1] and Weicheng Huang[1]
[1]National Center for High-Performance Computing, Hsinchu, Taiwan
E-mail：serenapan@nchc.org.tw, hsing@nchc.org.tw, yun@nchc.org.tw, chwhs@nchc.org.tw, whuang@nchc.org.tw

Abstract – *As virtualization technologies become more prevalent, each Cloud user encounters the problem of building his/her own virtual cluster with less friendly interface of virtual resource management. To help resolving this issue, an On-Demand Virtual Cluster in Cloud Web-Based OS feature has been developed by the Pervasive Computing Team in the National Center for High-performance Computing (NCHC). The On-Demand Virtual Cluster system exhibits the ability to reconfigure itself to adapt to the changes in the Cloud environment. And, it can discover, diagnose, and monitor Cloud computing resources automatically. An On-Demand Virtual Cluster system embedded in the Cloud WebOS is developed. An extremely lightweight approach helps the acquisition of virtual computing services to provide dynamically creation mechanism. The approach leverages virtualization techniques combined with cluster queuing system and transparently the fault tolerance capability to the parallel HPC application.*

Keywords: Virtualization Techniques, WebOS, Virtual Cluster, and Fault Tolerance.

1 Introduction

In Cloud computing environment, there are various important issues, including information security, virtual computing resource management, routing, fault tolerance, and so on. Among these issues, the virtual computing resource management has emerged as one of the most important issues in few years. As virtualization technologies become more prevalent, each Cloud user encounters manually to build wanted/specified virtual cluster with the non-straightforward interface of virtual resource management. To contribute to the issue, an On-Demand Virtual Cluster in Cloud WebOS (Web-Based Operation System) platform feature has been developed by the Pervasive Computing team in the National Center for High-performance Computing (NCHC).

The Cloud WebOS platform provides a new service paradigm [1]. The infrastructure of WebOS is the intention to offer a seamless and unified access to geographical distributed resources connected via Internet. It can supply most operation system services [2]. The current implementation also adopts the *Asynchronous JavaScript* and *XML* (AJAX) as base of the proposed Cloud WebOS platform. The major task of Cloud WebOS platform and designed On-Demand Virtual Cluster system can be easily customized and configured based on the end users needs. The On-Demand Virtual Cluster system exhibits the ability to reconfigure itself to the changes in the Cloud environment. And, it can discover, diagnose, and monitor Cloud computing resources automatically. Meanwhile, we also developed several Cloud widgets in Cloud WebOS platform for controlling virtual cluster and virtual machines.

Furthermore, an efficient fault tolerance mechanism is indispensable that provides supports for long-running parallel and message passing interface (MPI) jobs, in HPC or Cloud environment. As we known about the job failure rate in distributed environment, it is getting higher and higher, such as TeraGrid [3], and DAS-2 [4]. Therefore, the availability of fault tolerance mechanisms is not sufficient. To prevent valuable computation to be lost due to failures, this research combines virtualization technology to solve the kind issue. The NCHC Pervasive Computing development team not only built Cloud WebOS platform, along with the framework of eyeOS [5], but also incorporated the mechanism of fault tolerance.

This approach leverages virtualization techniques combined with cluster queuing system and fault tolerance mechanism. The main feature of this project is simplifying a lot complexity of utilizing Clouds. Then, the ultimate target is to make scientists/researchers painlessly run their jobs on Clouds automatically and transparently with the fault tolerance capability. Finally, the contribution of this paper is to provide lower barrier for using Cloud Computing Environment. The designed On-Demand Virtual Cluster in Cloud WebOS platform has become necessary to provide Cloud users with an interface that is both user-friendly and straightforward.

The rest of the paper is organized as follows. Section 2 presents related works. In Sections 3, we proposed On-Demand Virtual Cluster in Cloud WebOS, and describe the system architecture. And the following Section 4, these Research Results of the Designed Cloud Widgets and Experimental Results are presented. Finally, the conclusion and future directions are presented in Section 5.

2 Related Works

2.1 Existing Web-based Operation System (Web OS) Projects

Recently, there is a famous web application developed based on AJAX technique implementation, which is Gmail by Google [6]. Its success in the file transfer and rich text editing illuminates that the developments of Cloud WebOS platform via AJAX technique become practicable. However, it does not provide on-demand requirements for applications, especially for Cloud applications. Even though users can customize their personal homepage in Google, all the customizations are limited to the plain text and images.

Web-based Operating System (WebOS) project started at the University of California, Berkeley in 1996 as part of Network of Workstations [7]. So far, there are several typical commercial representatives of WebOS, such as FlyakiteOSX [8], Glide OS [9], XIN [10], and so on. All of these systems are online OS with Ajax and PHP techniques. However, these projects are not open source and also short of the management of distributed resources. To further enhance such an approach to meet the demands of Cloud Computing, Cloud Web-based Operating System (WebOS) platform, along with the framework of eyeOS has been developed. This development follows the spirit of open source, open standard and GNU/GPL license.

2.2 Virtualization Technologies

A wide range of methods and systems with fault tolerance have been developed in previous researches, mostly in the context of middleware where scalability was not using virtualization technologies and combining resource managers [11], [12], [13]. Our research enhances avoiding a complete restart, and retains execution of parallel and all kind of MPI jobs as nodes fail. To implement virtualization technology, there should be an additional software layer, called a virtual machine monitor or a hypervisor, inserted between the existing operating system and hardware to manage the resource and virtual machines. As an ensemble to support the execution of applications that needs physical hardware. The characteristics of virtualization technologies are described as the following:

- *Efficiency* - how much overhead is added by the virtualization technology, more overhead means less resources for virtual machines (VMs);
- *Isolation* - how well protected VMs are from one another, and how well protected the host is from the VM; better isolation means a VM can halt and catch fire without affecting the real host or other running VMs;
- *Flexibility* - the ability of the virtualization technologies to run platforms and operating systems that are different from the host, good flexibility means more choices for VM platforms and the ability to run VMs with minimal modification;
- *Manageability* - availability of tools and APIs for starting, stopping and moving VMs.

Generally, modern hypervisor implementations are divided into two categories, including Host-based and Bare-metal approaches. The host-based approach uses modified operating systems to provide virtual machine monitoring. Linux-VServer [11], OpenVZ [12], and Solaris Zones [13] are examples of the host-based approach. On the other hand, the bare-metal approach that employs small-dedicated hypervisors, which run on physical machines like operating systems in non-virtualized environments. VMware ESX server [14], Kernel-based Virtual Machine (KVM) [15], and Xen are the famous examples of the bare-metal approach.

With its success the virtual technologies of fault tolerance, we integrate virtualization technologies – KVM and WebOS technologies. This research comes up with a new and lightweight approach to acquiring virtual computing services via the proposed On-Demand Virtual Cluster.

3 Proposed On-Demand Virtual Cluster in Cloud WebOS

3.1 Research Objective

The key idea of Cloud Computing lies in its component-based nature, which are reusability, substitutability and user friendly. By integrating virtualization technologies and WebOS, we provided an approach to acquiring Cloud services via Cloud Widgets in the Cloud WebOS. This progress helps to lower the barrier for using Cloud Computing Environment. In order to develop an autonomic virtual computing resources management system based on decentralized resource discovery architecture, we implemented the On-Demand Virtual Cluster system in Cloud WebOS. This research focuses on virtual resources management with an interactive graphical user environment. Besides, it also can provide automatically and transparently the fault tolerance capability to the parallel HPC applications.

As the figure 1, it shows a high level overview of our Cloud WebOS. When the NCHC Cloud WebOS in the middle of this figure receives a Cloud job request from the users via the web browser, and then the job will be sent to the fittest virtual cluster in the backend to process via On-Demand Virtual Cluster system. The poposed system will help users to generate the fittest virtual cluster and

choose/allocate the physical resources with a graphical interface.

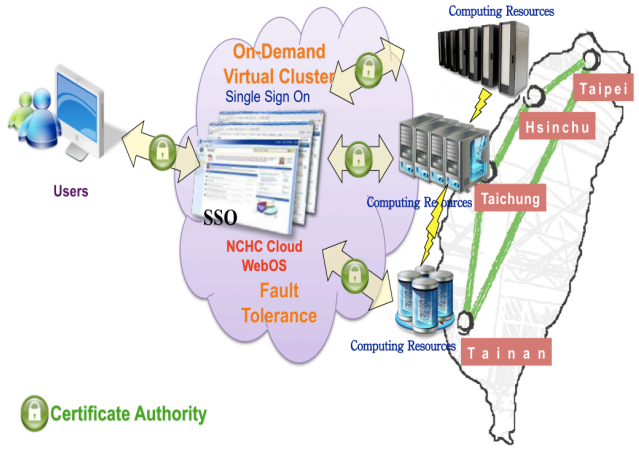


Figure 1. The Overview of Cloud WebOS Platform

3.2 Implementation and System Architecture

In this project, we combine the WebOS platform with Cloud computing resources to offer users a friendlier Cloud environment. The system architecture of the On-Demand Virtual Cluster in Cloud WebOS is sketched in the figure 2. In the Cloud WebOS, upon receiving a Cloud job request from the end users via Web Browser, the system acquires Cloud Services via Cloud Widgets, which in turn connect the Image Creator Widget, Virtual Machine (VM) Creator Widget, VM Monitor Widget, and VM Control Widget, within On-Demand Virtual Cluster system. Each of Cloud Widgets is described in **Section 4**. The proposed system helps selecting the most adaptive computing resources to create virtual clusters automatically based on the demands from the end users. These Widgets of On-Demand Virtual Cluster system in Cloud WebOS also drive the Cloud middleware to operate physical computing resources and storages.

Meanwhile, as the figure 3 is shown, the end users connect the Application Pool to get the software services, such as information security, and even Bio simulation, via Cloud WebOS easily. After connecting Application Pool, the Cloud WebOS also can integrate the public service provider, such as Amazon EC2 [16] and so on. Upon receiving Cloud job request via Cloud WebOS, the On-Demand Virtual Cluster makes communication with Cloud Middlewares, which are Data Broker, Monitoring & Reporting, and Dynamic Provisioning. The Data Broker collects data from the distributed physical sensors. The Monitoring & Reporting takes responsible for monitoring the status of physical machines and virtual machines. Finally, the Dynamic Provisioning provides the capability of resource allocation automatically, the feature of load prediction dynamically, and the fault tolerance module is activated at the same time. It improves the performance of the dynamic scheduling over conventional scheduling policies.

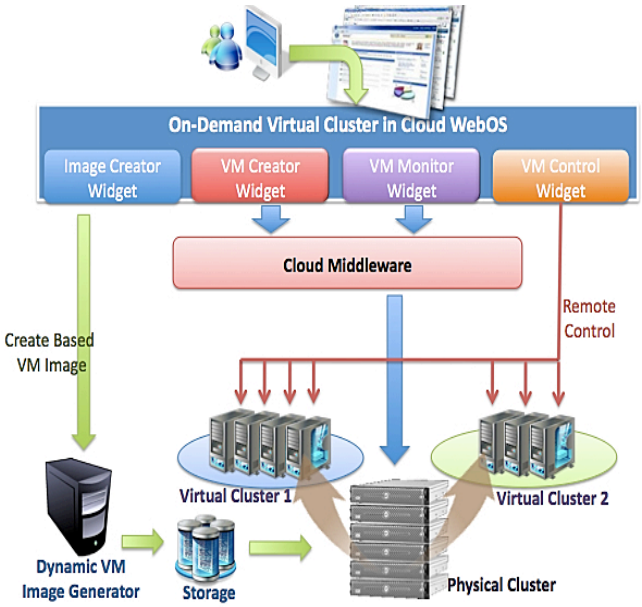


Figure 2. The System Architecture of On-Demand Virtual Cluster in Cloud WebOS

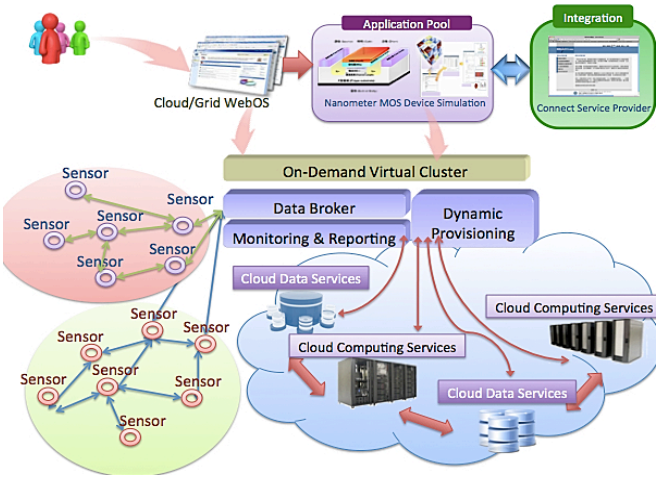


Figure 3. The Application of On-Demand Virtual Cluster

With On-Demand Virtual Cluster in Cloud WebOS, users can create a dynamic HPC cluster consisting of VMs. The scale of each virtual cluster can be determined by user's criteria. When user needs virtual cluster no longer, the cluster can be destroyed. Then the computing resources are released. The whole operation can be manipulated via web browser, because we use XML-RPC based Application Programming Interface (API). Moreover, there are two middlewares embedded into the proposed Cloud WebOS, as the following shown:

- *Integration with OpenNebula* – OpenNebula is used as central cloud management. It is responsible for finding available computing resources, creating VMs based on a selected image, and deploying the

image into the physical computing resources. It also manages unique MAC address, IP address, and virtual network (vNet) ID. Therefore, each user's cluster lives on its own vNet, in order to isolate the various virtual clusters.

- *Batch System with Torque* – It is used for the scheduling of virtual cluster and physical cluster. The fault tolerance module is embedded with this resource manager, Torque in the figure 4. This development not only makes users submit job as usual via PBS_SERVER, but also makes resource manager have additional capabilities of automatic fault tolerance and job migration with virtual technology. The detail explanation is in **Section 3.3**.

3.3 Fault Tolerance Module in Cloud WebOS

The kernel of the fault tolerance module consists of three parts, including Job Control Module, Virtual Machine Control Module (VM CTL), and Virtual Machine Monitor Module (VM Monitor). When PBS_SERVER receives a long-running MPI jobs request from users, users can activate fault tolerance module very easily. As long as adding key parameter in original PBS script, the PBS_SERVER wakes up Job Control Module. It makes communication with PBS_MOM on each computed nodes, in figure 4. And then it can deal with dispatching and scheduling job via VM CTL. The VM CTL controls KVM hypervisor to create virtual environment that we called Guest OS (Operation System). It also takes snapshot and makes checkpoint at regular intervals. The following VM Monitor, it collects real time status of Guest OS, and then reports to PBS_MOM. As such, PBS_MOM can monitor the status of job execution.

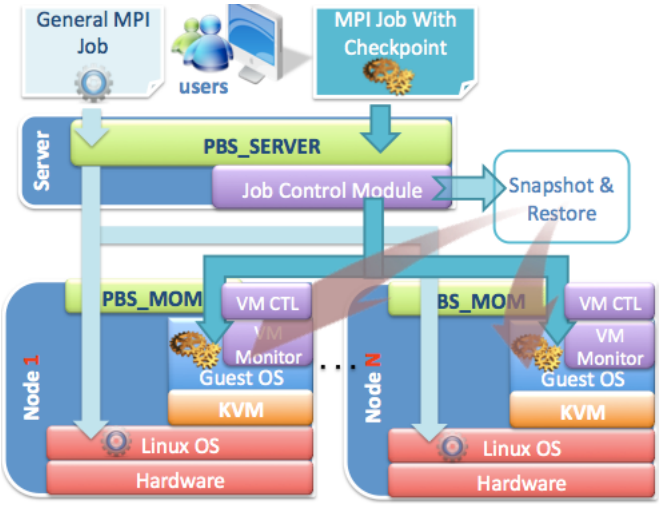


Figure 4. The Fault Tolerance Module in Cloud WebOS

If the computed node was crash or network was broken, the Job Control Module would inform VM CTL. VM CTL immediately activates these mechanisms of Checkpoint, and Live Migration. Meanwhile, a failed node is replaced with a health node where the corresponding MPI jobs are recovered from the last checkpoint. Hence, live nodes remain active while failed nodes are dynamically and transparently replaced.

Job Control Module also gets abnormal signal of jobs from VM Monitor. It removes failed jobs, and then generates new job ID to roll back the last checkpoint. Once the snapshot version selected, the checkpoint will be restored, in another computed node if necessary, to recover the failed instance. The computation will thus be continued without any interruption. Users can easily distribute and balance the workload among multiple computed nodes, thus to enhance the utilization of computing resources.

4 Research and Experimental Results

4.1 Research Results - The Designed Cloud Widgets

In addition to the basic Widgets, more advanced Cloud Computing Widgets are attempted as well. One of the most important results in this paper is that we have developed many Cloud Widgets with friendly graphical user interface, especially the On-Demand Virtual Cluster System in WebOS. The kernel of this system architecture consists of four Widgets, including Image Creator Widget, VM Creator Widget, VM Monitor Widget, and VM Control Widget. Users without much learning effort can easily manage all of these widgets. Besides, these Widgets allow users to customize and to arrange their complicated computing tasks according to their requirements.

The Image Creator Widget, in the figure 5, is to generate the customized base image and on-demand/specified application from the end users' requirements. This Widget provides a complete and integrated HPC software stack that consists of operating system, management tools, resource monitor, and even commercial package, such as the Matlab. VM Creator Widget - with the profile of virtual cluster demanded by the user provided, it will generate a specification, shown in the figure 6, which in turn is parsed by the VM Creator engine to create an On-Demand Virtual Cluster on the physical computing resources.

In the figure 7, the main task of the VM Monitor Widget is to monitor the all the status of virtual machines, Networks, and the physical hardware. In addition, this Widget makes use of the information and the status provided by the Monitoring & Reporting Cloud Middleware. The VM Control Widget is designed for such a purpose with Cloud visualizer integrated as the core of its Cloud service, as shown in the figure 8 and the figure 9.

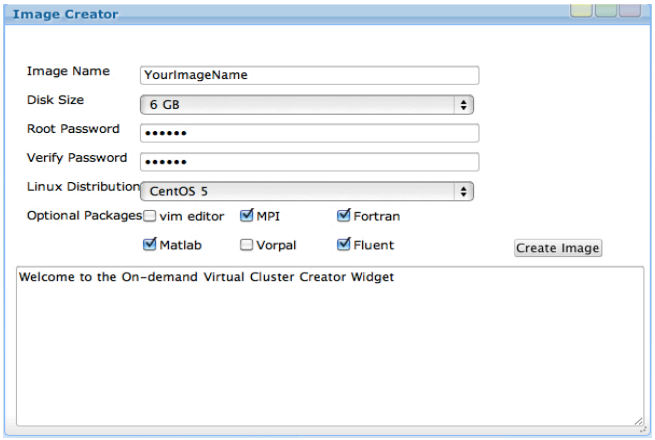


Figure 5. Image Creator Widget

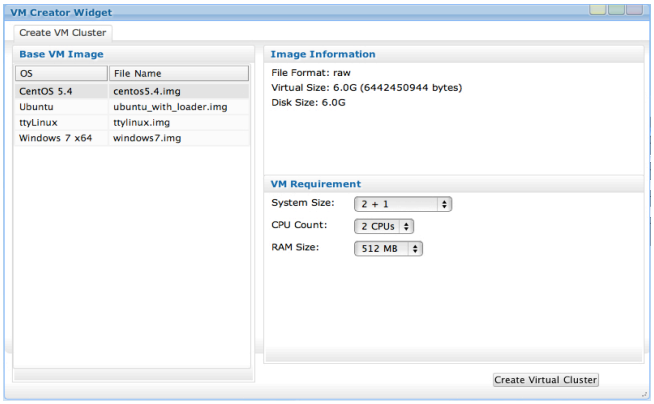


Figure 6. VM Creator Widget

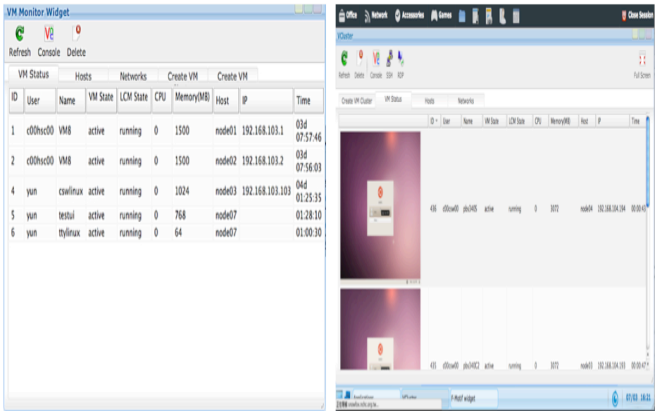


Figure 7. VM Monitor Widget

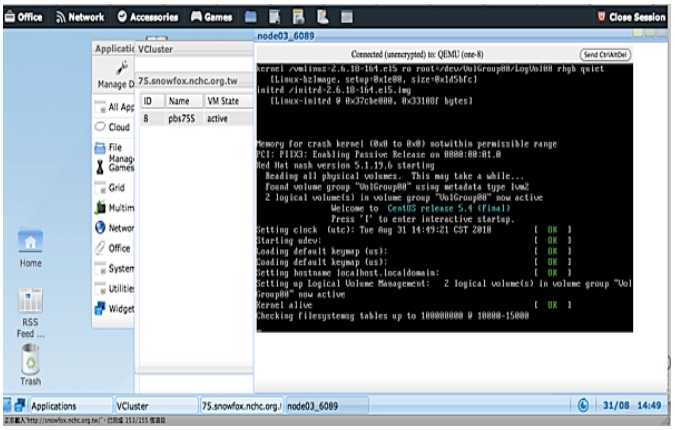


Figure 8. VM Control Widget Cloud Visualizer – Linux Booting Status

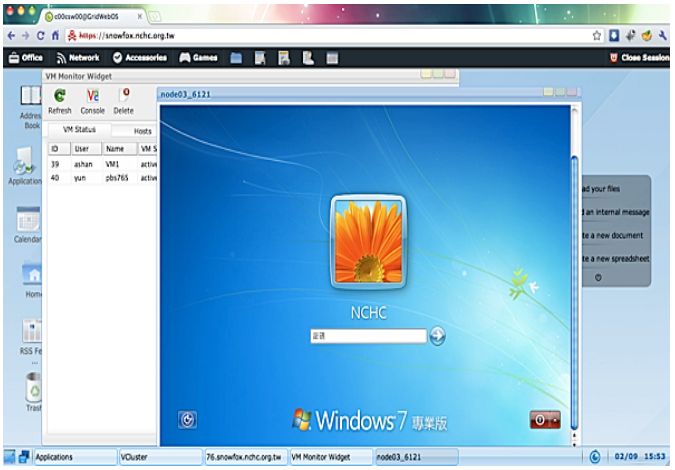


Figure 9. VM Control Widget Cloud Visualizer – Windows 7 Booting Status

We used above Cloud Widgets to implement the concept of "Carry on Cloud". There are two customized applications for biological simulation and information security simulation. The F-motif Simulation Widget provides specialized Cloud services to search and analyze the sequence of gene in real time, in figure 10. The other customized Cloud Widget about information security is called ICAS (IDS-log Cloud Analysis System) Widget. As long as user selects the ICAS base image, the Hadoop DB and virtual cluster are constructed automatically, as the following figure 11 shown.

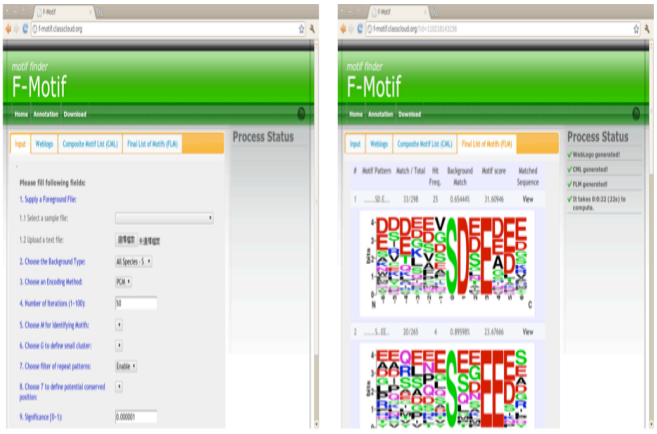


Figure 10. F-motif Widget

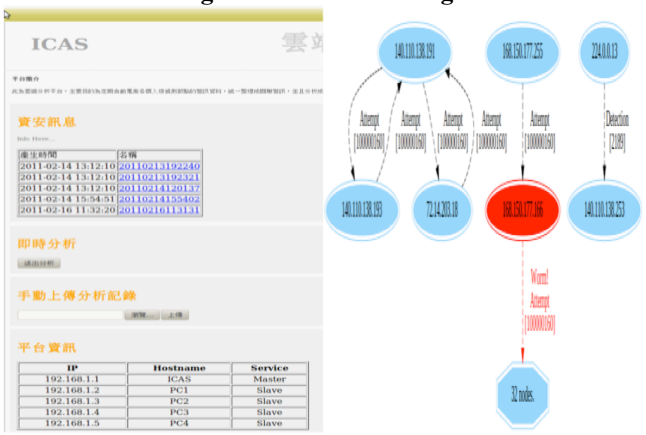


Figure 11. ICAS Widget

4.2 Experimental Results

The preliminaries of experiment are needed to set up, including the multi-sites physical computing environment, the virtual machine – KVM, Network Speed Test [17], and Disk I/O test tool - bonnie++ [18]. As shown in Table 1 and Table 2, we list the experiment parameter statement used in our experimental environments and the summary environment characteristics of NCHC computing resources.

Table 1. Experiment Parameter Statement

R	Number of real nodes
V	Number of virtual machines
RV	Number of virtual machines created in each real node

Table 2. Summary Environment Characteristics of NCHC Computing Resources

Resource	CPU Model	Memory (GB)	CPU Speed (MHz)	#CPUs	Nodes	Job Manager
Snowfox	Intel(R)Xeon(R) CPU 2.5GHz, E5420	16	2500	112	14	Torque

The following part, we discuss experimental results. There are three scenarios, including the performance of Network I/O, the performance of Disk I/O, and the performance of Message Passing Interface (MPI) program on VM cluster. Moreover, in order to improve the performance, we use the Virtio driver in the virtual machines. Virtio driver provides paravirtualized functions for network virtualization and disk I/O virtualization. The solutions of Laplace's equation are important in many fields that consist of science, fluid dynamics, astronomy and notably the fields of electromagnetism [19]. Therefore, we adopt the Laplace equation for heat conduction to be our benchmark with a MPI matrix to evaluate the performance.

In the figure 12, we found the Network speed is tackled about 166 Mb/s without Virtio, because the I/O bottleneck is between virtual machine and hypervisor. Therefore, our proposed system is activated the Virtio. The performance of Network I/O is nearly the same with native machine. In the performance of Disk I/O scenario, we compared with Virtio and without Virtio. With Virtio, it can be improved write performance about 120% and read performance about 20%, as the following figure 13 shown.

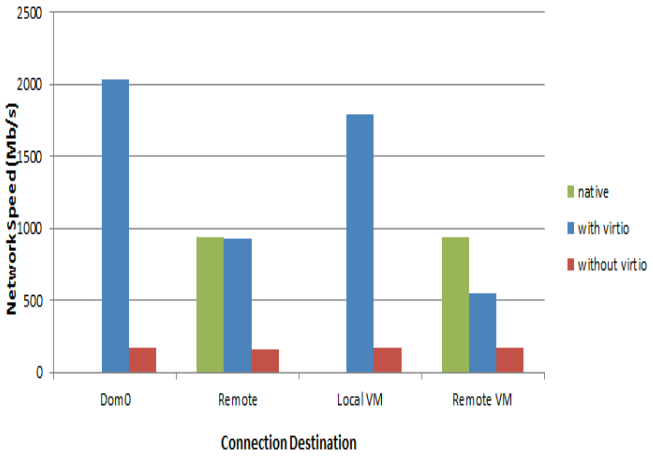


Figure 12. The Performance of Network I/O

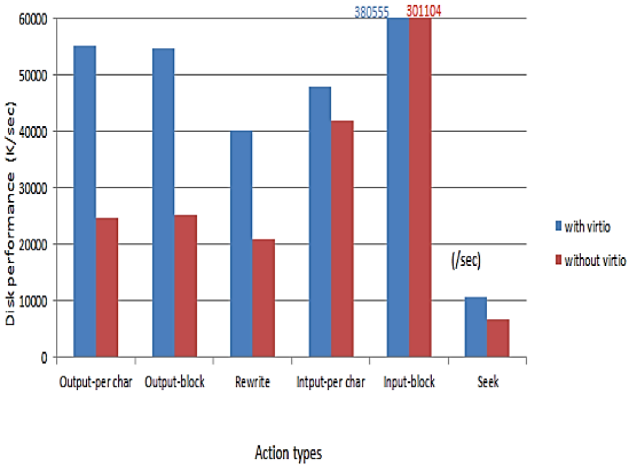


Figure 13. The Performance of Disk I/O

In the performance of MPI program on virtual machine scenario, we study the decrement of performance due to the virtualization as shown in Figure 14. We take an MPI job, which needs 8 cores as example. Each VM runs respectively on a real node, the number of processor core in VM is set as

2, and we change ”R” and ”V”. In cluster computing, if we want to integrate real nodes and virtual machine to execute a parallel application collectively, we should understand the possible decrement of performance due to the virtualization. When the number of VM increases, the turnaround time of the executed job follows to increase, the overhead amounts to 11.73% in the case of using virtual machines entirely. In order to reduce the overhead, we activate the Virtio in the virtual machine, and the overhead can be lowered to 6.56%.

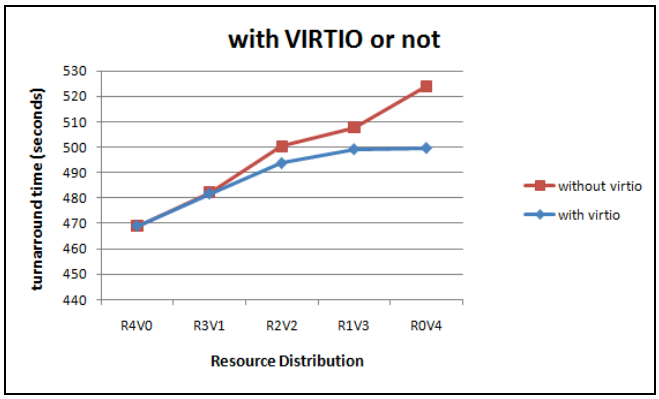


Figure 14. The Degree of Virtualization with Virtio and without Virtio

We also study the performance by the different distribution of VMs in real nodes as shown in following figure 15 and figure 16. The total number of VMs is fixed as 8, the number of processor core in VM is set as 1, and we change ”R” and ”RV”. The purpose of this experiment mainly provides a suggestion for users to distribute the virtual machines in real nodes when creating VM cluster. The experiment results show that there is best performance in the case of “R8 RV1” (8 virtual machines in 8 real nodes). When the number of virtual machines in single real node increases, the turnaround time of the executed job follows to increase. The overhead amounts to 24.61% in the case of creating 8 virtual machines in single real node. This is because there is limited memory and CPU power in single real node. The whole performance would go down greatly if the available resource of real node runs out.

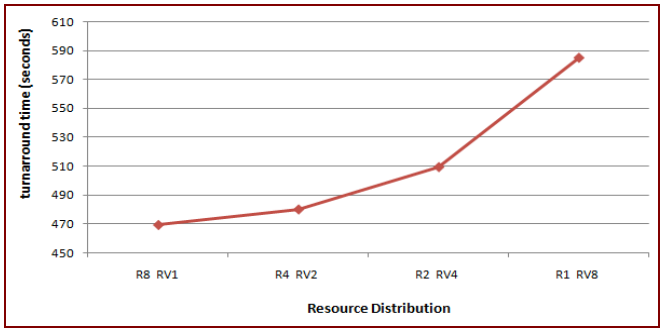


Figure 15. The Distribution of VMs in Real Nodes (1/2)

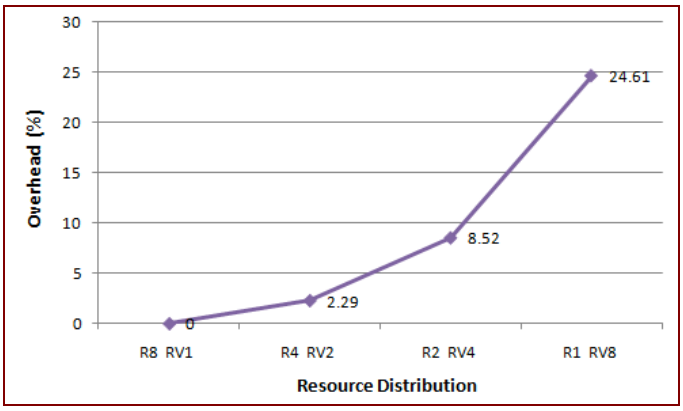


Figure 16. The Distribution of VMs in Real Nodes (2/2)

5 Conclusion and Future Work

The proposed toolkit – On-Demand Virtual Cluster system in WebOS, provides Cloud users with an interface that is both user-friendly and more straightforward. In this project, we combine the WebOS platform with Cloud computing resources to offer users a friendlier Cloud environment. The On-Demand Virtual Cluster in Cloud Web-Based OS Environment not only helps user to build virtual cluster easily and automatically, but also provides different varieties of computing environment such as Linux, Win7, and so on. This approach leverages virtualization techniques combined with cluster queuing system and fault tolerance mechanism. The main feature of this project is simplifying a lot complexity of utilizing Clouds. Then, the ultimate target is to make scientists/researchers painlessly run their jobs on Clouds automatically and transparently with the fault tolerance capability. Finally, the contribution of this paper is to provide lower barrier for using Cloud Computing environment.

The designed On-Demand Virtual Cluster in Cloud WebOS platform has become necessary to provide Cloud users with an interface that is both user-friendly and straightforward. Besides, the fault tolerance module can ensure the successful execution of a parallel HPC job for both within a cluster or under the Cloud or distributed computing environment, and thus to improve the reliability of computing resources.

6 References

[1] W. Kim, “Cloud computing: Today and Tomorrow,” *Journal of Object Technology*, 8, 2009.

[2] G. Gu, and X. Lu, “Simple Web OS System Based on Ext Framework and Cloud Computing,“ *International Forum on Information Technology and Applications*, pp. 448-450, IEEE, 2010.

[3] O. Khalili, J. He, C. Olschanowsky, A. Snavely, and H. Casanova," Measuring the Performance and Reliability of Production Computational Grids," *7th IEEE/ACM*

International Conference on Grid Computing (GRID 2006), Proceedings, pp. 293–300, IEEE, Barcelona, Spain, Sep 2006.

[4] A. Iosup and D. Epema, "GRENCHMARK: A Framework for Analyzing, Testing, and Comparing Grids," *CCGrid*, Vol. 00, pp. 313–320, 2006.

[5] http://www.eyeos.org/, 2011.

[6] http://www.gmail.com, 2011.

[7] Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau, David E. Culler, Joseph M. Hellerstein, and David A, Patterson, "Searching for the Sorting Record: Experiences in Tuning NOW-Sort," *The 1998 Symposium on Parallel and Distributed Tools (SPDT '98)*, Welches, Oregon, August 3-4, 1998.

[8] http://osx.portraitofakite.com/logon.htm, 2011.

[9] http://www.glidedigital.com/, 2011.

[10] http://www.xindesk.com/, 2011.

[11] S. Soltesz, H. P¨otzl, M. E. Fiuczynski, A. Bavier, and L. Peterson, "Container-based Operating System Virtualization: A Scalable, High-Performance Alternative to Hypervisors, " *Proceedings of ACM SIGOPS/Eurosys European Conf. on Computer Systems*, pp. 275-287, Mar. 2007.

[12] http://wiki.openvz.org/Main_Page, 2011.

[13] D. Price and A. Tucker, "Solaris Zones: Operating Systems Support for Consolidating Commercial Workloads, " *Proceedings of 18th Large Installation System Administration Conf.*, pp. 241-254, Nov. 2004.

[14] John Paul, "VMWare ESX Server Workload Analysis: How to Determine Good Candidates for Virtualization," *33rd International Computer Measurement Group Conference*, pp. 483-484, San Diego, CA, USA, December 2-7, 2007.

[15] B. Zhang, X. Wang, R. Lai, Y. Liang, Z. Wang, Y. Luo, and X. Li, "Evaluating and Optimizing I/O Virtualization in Kernel-based Virtual Machine (KVM)," *International Conference on Network and Parallel Computing,* pp. 220-231, Zhengzhou, China, September 13-15, 2010.

[16] http://aws.amazon.com/ec2, 2011.

[17] http://vmstudy.blogspot.com/2010_04_01_archive.html, 2011.

[18] http://www.coker.com.au/bonnie++/, 2011.

[19] http://en.wikipedia.org/wiki/Electromagnetism, 2011.

A Hybrid Dynamic Load Balancing Algorithm for Heterogeneous Environments

Mayuri Mehta[1] **and Devesh Jinwala**[2]
[1]Department of Computer Engineering, Sarvajanik College of Engineering and Technology, Surat, Gujarat, India
[2]Department of Computer Engineering, S. V. National Institute of Technology, Surat, Gujarat, India

Abstract - *Dynamic load balancing has become a necessity in emerging distributed environments to address the inherent heterogeneity in computing resources. The majority of traditional dynamic load balancing algorithms were developed assuming homogeneous set of nodes and suffer significant performance drop under variety of workloads. Moreover, the centralized dynamic approach limits the scalability with the load balancing unit itself becoming a bottleneck. Conversely, the decentralized dynamic approach though overcomes the above problems, suffers from increased communication overhead. Here, we propose the design of a simple yet effective hybrid dynamic load balancing algorithm that overcomes limitations of centralized and decentralized approaches and performs competitively for heterogeneous system. As per our theoretical analysis, under this new hybrid approach, a highly loaded node is expected to discover lightly loaded node in lesser time as compared to decentralized dynamic approach, minimizes the communication overhead, and is also expected to produce improved performance under variety of workloads*.

Keywords: Dynamic load balancing, distributed system, centralized load balancing, decentralized load balancing

1 Introduction

Over the last decade, distributed computing systems, in general, have been facing a complexity crisis: *increasing heterogeneity, scalability, dynamism, and interconnectivity of processing elements* that demand building high quality load balancing algorithm to achieve significant improvements in performance [1][2]. Load balancing is an operation that involves redistribution of the system workload, as evenly as possible, among the processing elements of a distributed system based on prior analysis of the existing load. The underlying focus is to achieve *optimal* resource utilization, *maximum* throughput, and *minimum* response time [3][4]. An associated aim is also to avoid system overload by using the processing power of the entire system *evenly* to smooth out periods of high congestion at individual nodes and network [5] [6].

A load balancing algorithm could be static or dynamic depending on the freshness of the input parameters used for determining the load distribution. Obviously, the latter is more responsive to changes in system workload and hence is more suitable to heterogeneous computing environments [7]. A Dynamic Load Balancing (DLB) algorithm attempts to utilize the current system state information to make effective load distributing decisions [7]. The benefits from a DLB algorithm are *higher throughput, reduced mean job response time, higher reliability, dynamism, higher resource utilization, and adaptability to fluctuations in load* as compared to the same in a static algorithm.

A DLB algorithm is further based on either centralized OR decentralized (distributed) approach [8]. In a centralized DLB algorithm, a central node acts as the load balancing node, working on the jobs from different nodes in the system [9]. However, this approach limits the scalability because the load balancing unit itself becomes a bottleneck and susceptible to failures [10]. On the other hand, in a decentralized approach, all nodes participate in load balancing [11]. This approach supports scalability and has better fault tolerance. However, the critical issue here is the inter-node communication and synchronization required – that involves an all-to-all broadcast as compared to that in the centralized approach (all-to-one and one-to-all) [12]. Thus, though the decentralized approach overcomes the limitations of the centralized approach, it raises the number of messages used for information exchanges. Therefore, the development of an efficient DLB algorithm is really critical design issue – one that must ensure a tradeoff between the scalability and efficiency.

In this backdrop, we propose the design of a simple yet effective hybrid dynamic load balancing algorithm. Our approach typically sits between the two extremes mentioned above. We exploit the established notion of divide-and-conquer for dealing with the scalability – we partition a system of n nodes into appropriate number of m groups, where $1 < m < n/2$. In each group, one node is designated as a coordinator to deal with the scalability issue and ensures failure tolerance. To reduce the communication overhead, a highly loaded node first searches lightly loaded node in its group in decentralized fashion. However, in case if it does not

find a suitable node in the same group, its overload is transferred to another node in another group with the help of a central *MasterNode*. Utilizing this hybrid dynamic load balancing algorithm that can perform competitively for a heterogeneous distributed system, a highly loaded node is expected to find a lightly loaded node in lesser time. For the proposed algorithm, we have considered a new load measurement policy that takes into account the load of major influencing resources to obtain accuracy in load status and to support a variety of applications. Moreover, we believe the proposed algorithm will be able to deal with varying demands of different distributed applications such as CPU-bound, memory-bound, I/O-bound, etc.

The rest of the paper is organized as follows: In section 2, we briefly describe previous work related to dynamic load balancing algorithm. In section 3, we discuss major design issues taken into account while designing a new dynamic load balancing algorithm. The proposed hybrid DLB algorithm and its analysis are presented in section 4. Finally, conclusions and some future work are specified.

2 Related work

One can find numerous attempts in the literature to design the DLB algorithms that try to address one or the other design issue. However, as per our observations, none of them address the efficiency concerns in a typical heterogeneous system. We survey the existing attempts in the following.

A dynamic, decentralized load balancing scheme for heterogeneous grids is presented in [4] that utilizes an effective state information exchange to reduce the communication overhead. In [13], a semi-distributed scheduling technique that reduces the communication overhead by dividing the nodes of the system into mutually overlapping subsets is developed. However, this approach is targeted at systems that are composed of identical processors. An adaptive decentralized load balancing algorithm for heterogeneous grid environment is proposed in [2] that outperforms conventional approaches even in heterogeneous environment and even when communication overhead is significant. In [14][15], influence and quality of several load indices on the performance of a dynamic load balancing are empirically evaluated. The results therein show that the performance benefits of load balancing are indeed strongly dependent upon the load index. In [5][6][16][17], a DLB algorithm with new load estimation policy has been proposed. Based on exhaustive analysis of effects of system heterogeneity on load balancing algorithms' performance, new state measurement and information exchange techniques have been presented in [18].

However, none of these approaches considers all the critical issues while designing a DLB algorithm as we do in this attempt; at the same time overcoming the limitations of centralized and decentralized load balancing algorithms. Our proposal in this paper is aimed to generate a new hybrid effective dynamic load balancing scheme that

- overcomes the constraints of centralized and decentralized approaches
- utilizes efficient load index
- makes use of information strategy that minimizes the communication overhead
- considers heterogeneity in computing resources
- aids highly loaded node to obtain lightly loaded node in lesser time
- is applicable to wide range of applications viz. *CPU-bound, memory-bound, I/O-bound, and various combinations of these.*

3 Design issues

There are various factors pertaining to design of a load balancing algorithm. For the proposed hybrid approach, all these factors are discussed in the following.

3.1 Heterogeneity

The proposed algorithm is designed to consider system heterogeneity that is to be associated with varied hardware and software resources of the nodes. We address heterogeneity with respect to (1) diverse set of CPUs, memories, and disks (2) machine platform. Specifically, we characterize each node n_i by its CPU speed C_i, the storage capacity M_i and the disk speed D_i.

3.2 Target applications

A real distributed system can have workload consisting of applications that may be using significant amount of CPU, memory, and/or I/O. Therefore, to prevent performance fall under variety of applications, it is essential to make a decision upon intended applications while designing a DLB algorithm. The proposed algorithm is expected to perform efficiently for various types of applications that have different impacts on the underlying resource usage i.e. *CPU-bound, I/O-bound, memory-bound, and any combination of these*.

3.3 Metrics for load measurement

Our design also takes into account the following parameters to measure the load at each node in the system; based on which the decision for further distribution is done.

- average CPU queue length
- average CPU utilization
- average memory queue length
- average memory utilization
- average I/O queue length
- average I/O utilization
- average context switch rate

An appropriate combination of these parameters is used to constitute the load index. However, amidst the changing system load, using the instantaneous values of the resource utilization statistics may not give useful load information and may lead to bad job transfer decisions. Therefore, rather than using instant values of metrics, *time-averaged* metrics should be utilized to compute the load index. Obviously , averaging the load measurement metrics over a short interval (up to about 10 seconds) produces significant improvement over the instantaneous values, and using too long averaging interval reduces the responsiveness of the index to load changes [15]. Hence, we use time-averaged load of resources over an interval of 5 seconds.

3.4 Components of DLB algorithm

Although various kinds of components exist, they should be chosen according to the desired environment, such as application types or system environment. Following are the four components of the proposed algorithm.

The proposed *information policy* uses an effective mechanism to exchange state information that considerably reduces the communication overhead while quickly updating the state information. We use traditional *demand-driven* information policy. When any node becomes highly loaded, initially, load information of only local nodes that reside within the same group is collected. If a lightly loaded node is not found within the same group, other groups are searched to find a lightly loaded node. In former case, lesser number of messages will be used while in latter case, the number of messages will increase gradually.

We use a threshold policy as the *transfer policy* that is dynamic in nature. However, for simplicity single threshold policy is considered. Defining a suitable threshold value for a particular computing environment is a challenging task. Therefore, one node in each group is selected as a *SuperNode*. The SuperNode periodically collects the load information of other nodes in the group to compute the average load of the group. This average load is set as a new threshold value *T* and is broadcast to other nodes in the group.

Selection policy selects the best job to transfer. Here, newly arrived process is considered for transfer because it may be the process due to which a node has become highly loaded. Moreover, its execution has not started yet and therefore, it is cheap to transfer this process. Thus, nonpreemptive task transfers are supported by the proposed algorithm.

Location policy considered is polling (or probing). Under this policy, a highly loaded node polls all the nodes in the same group to find out a lightly loaded node. The node whose load is lowest taking into account all the parameters mentioned in section 3.3 is considered as the destination node. Thus, this approach makes an effort to choose the best node for the process transfer. The destination node executes the process regardless of its resource usage at the time of arrival of the transferred process.

4 Proposed algorithm and analysis

The proposed algorithm is designed for a system in which $N = \{ n_1, n_2, n_3, \ldots, n_n \}$ nodes are connected via a high speed communication network. Each node in the system is composed of a combination of various resources including processor, memory, disk, network connectivity and every node differs in its capacity of processor, memory and disk. A system is divided into m groups, where $1 < m < n/2$. For simplicity, static groups are created. Moreover, each group can have different size where a number of nodes in each group can be minimum 2 and maximum n/2. In each group one node is selected as *SuperNode* (SN) that periodically collects the load of the other nodes in the group to set new threshold value. Communication between groups is possible via a central node called *MasterNode* (MN). Figure 1 shows groups created for simplified distributed system. Here, system consists of n=16 nodes that are divided into m=4 groups, where group1 has 5 nodes, group2 and group3 have 4 nodes, and group4 has 3 nodes. Nodes 4, 6, 14, and 8 are SuperNodes. Each node has access to MasterNode. Each node itself is responsible for monitoring available resources of other nodes in the group.

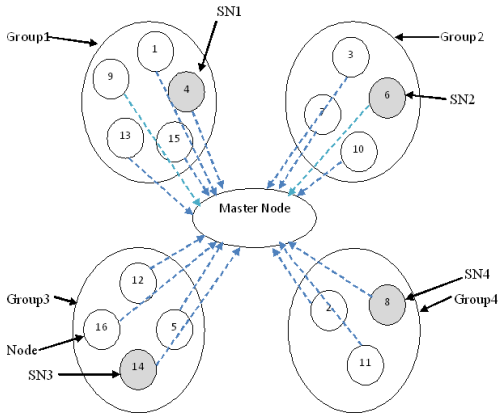


Figure 1. View of groups in simplified distributed system with Master Node

When a node becomes overloaded, it first queries the member nodes of its group and collects the load information in a decentralized fashion. There is a high probability that the overloaded node will find a lightly loaded node in the same group (unlikely if the majority of nodes in the group are highly loaded). However, when an overloaded node does not find a suitable destination node in the same group to transfer its overload, it requests a centrally located MN to search for the lightly loaded node in the other groups. The MN randomly selects one group and collects the load status of all nodes in this group to ensure the availability of destination node. If any lightly loaded node is found, the overload will be transferred

to this node, otherwise same process will be repeated until either a lightly loaded node is found or all groups are searched. In the latter case, the overload will be executed on the original node.

Many times when an overloaded node does not find a lightly loaded node in the same group, it needs to consult central MN to transfer its overload. When the system itself is highly loaded, many nodes are expected to seek advice from this MN and gradually MN becomes overloaded. In this case, performance cannot be improved much. Moreover, MN becomes a bottleneck and a single point of failure. To deal with this issue, one approach is to split MN into supporting nodes [19]. Figure 2 shows various groups in such a simplified distributed system where MN is replaced by multiple supporting nodes S_1, S_2, S_3, ……., S_p. Each node in the system is allowed to access any supporting node. Here, when a highly loaded node does not find a lightly loaded node in the same group, it randomly selects one supporting node and now supporting node follows the same procedure as pursued by MN.

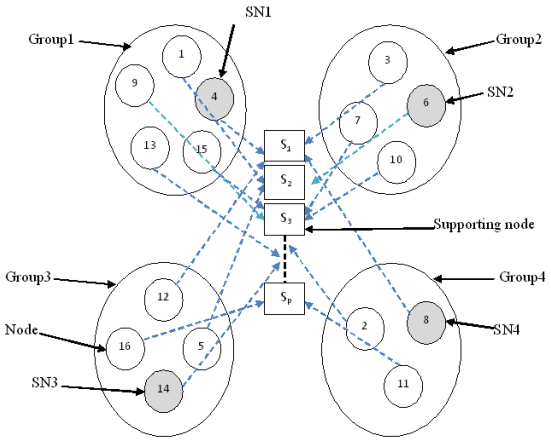


Figure 2. View of groups in simplified distributed system with supporting nodes

Table 1 shows various notations used in discussion.

Table 1. Definition of notations

Notations	Definition
N	Set of nodes in distributed system
N	Number of nodes in system
ni where 1 < i < n	Different nodes in system
M	Number of groups
MN	Master node
SN	SuperNode
S	Supporting node
Li	Load of the node ni
Pi	Overload of the node ni
T	Threshold value to decide status of the node

The pseudocode that is utilized by an overloaded node to make scheduling decisions is shown in figure 3.

```
Assumption: Each node is having some
load
Input: load li of the node and
threshold T
Output: destination node to transfer
overload

LoadBalancingProcedure (li, T)
1. IF li > T for any node ni
2. collect the load status of all
nodes in the same group
3. take out lightly loaded nodes
4. destination node =  most lightly
loaded node
5. IF (destination node found)
6.   dispatch ρi =  li – T load to
this node
7.   EXIT
8. ELSE
9. Consult MN (or randomly selected
supporting node) to find lightly
loaded node in other group
```

Figure 3. Algorithm utilized by a highly loaded node

The procedure for master node/supporting node is described in figure 4 that is operated when master node/ supporting node gets a request from an overloaded node to locate lightly loaded node, if any, in other groups of the system.

```
Assumption: Each node is having some
load
Input: no parameters
Output: destination node to transfer
overload

LoadBalancingProcedureMN ( )
1. WHILE (destination node is not
found OR all groups are not searched)
2. FOR each group in the system
3.   obtain value of threshold T for
the selected group from SN
4.   collect the load of all nodes in
the same group
5.   take out lightly loaded nodes
6.   destination node =  most lightly
loaded node
7. IF (destination node found)
8.   send ID of destination node to
overloaded node
9.   Transfer overload to destination
node
10.  EXIT
11.ELSE
12. Execute overload on the original
node
```

Figure 4. Algorithm used by Master Node or Supporting Node

4.1 Analysis

The proposed algorithm is evaluated using following two parameters: *communication overhead* and *response time*. The algorithm is expected to produce improved performance for both these parameters. Our theoretical analysis for the same is presented below.

Let's assume that node 9 in group-1 has become overloaded. As per the proposed approach, initial number of messages used to collect the load information from the same group is equal to 8 (4 request messages and 4 response messages). If lightly loaded node is found in this group, then 1 message will be used to transfer overload to destination node. Thus, the number of messages is limited to 9 if lightly loaded node is found in the same group. However, if a destination node is not found in the same group, the highly loaded node sends a request to MN to locate lightly loaded node from the other group. MN will search group2, group3, and group4 until lightly loaded node is found with 19, 27, and 33 messages respectively. Table 2 summarizes the number of messages utilized by decentralized and hybrid approaches in order to achieve load balancing for the given example.

Table 2. Comparison of decentralized approach and proposed hybrid approach for specified example

	Communication overhead	
	Decentralized approach	**Proposed hybrid approach**
Best case	31	9
Average case	31	19 or 27
Worst case	31	33

This analysis shows that proposed approach has higher probability of lesser communication overhead. For the above mentioned example, it takes only 9 messages to transfer overload in best case while the same communication in decentralized approach would have taken 31 messages. In average case, it takes 19 or 27 messages. However, in worst case, it may need 33 messages to transfer the overload, but the likelihood of worst case is less. Moreover, it has been reported in literature that lesser communication overhead minimizes response time. Thus, proposed approach is expected to improve performance of system in terms of communication overhead and response time.

5 Conclusions

We have designed an effective hybrid dynamic load balancing algorithm with three main objectives: efficient load balancing approach to overcome several limitations of centralized and decentralized approaches, load measurement policy that gives load status of major affecting parameters, and effective information policy. The proposed algorithm aims to balance the load in a decentralized fashion locally in each group. However, if suitable node in not found locally, load will be balanced utilizing a central node. This hybrid load balancing approach significantly minimizes the communication overhead along with demand-driven information policy. Our algorithm is expected to perform efficiently for CPU-intensive, memory-intensive, I/O intensive applications, and any combination of these because we have defined the load index utilizing load information of critical resources viz. CPU, I/O, memory to measure the load. At present, we are implementing the proposed algorithm using the GRIDSIM simulator.

6 References

[1] Belabbas Yagoubi, and Yahya Slimani. "Dynamic Load Balancing Strategy for Grid Computing"; Transactions on Engineering, Computing and Technology (World Academy of Science), Vol. No. 13, 260—265, May 2006.

[2] Ruchir Shah, Bhardwaj Veeravalli, and Manoj Misra. "On the Design of Adaptive and Decentralized Load Balancing Algorithms with Load Estimation for Computational Grid Environment"; IEEE Transactions on Parallel and Distributed Systems (IEEE Press), Vol. No. 18, Issue No. 12, 1675—1686, Dec 2007.

[3] Abbas Karimi, Faraneh Zarafshan, Adznan.b. Jantan, A.R Ramli, and M.Iqbal b.Saripan. "New Fuzzy Approach for Dynamic Load Balancing Algorithm"; International Journal of Computer Science and Information Security, Vol. No. 6, Issue No. 1, 1—5, Oct 2009.

[4] Issam Al-Azzoni, and Douglas G. Down. "Decentralized Load Balancing for Heterogeneous Grids"; Proceedings of the Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns, IEEE Computer Society, 545—550, 2009.

[5] Sandeep Sharma, Sarabjit Singh, and Meenakshi Sharma. "Performance Analysis of Load Balancing Algorithms"; World Academy of Science, Engineering and Technology, Vol. No. 28, 2008.

[6] M. Venu Gopalachari, P. Sammulal, and A. Vinaya Babu. "Correlating Scheduling and Load Balancing to Achieve Optimal Performance from a Cluster"; Proceedings of IEEE International Conference on Advance Computing, IEEE press, 320—325, Mar 2009.

[7] Pushpendra Kumar Chandra, and Bibhudatta Sahoo. "Prediction Based Dynamic Load Balancing Techniques in Heterogeneous Clusters"; Proceedings of the International Conference on Computer Science and Technology, National University, San Diego California, 189—192, Apr 2008.

[8] Janhavi A. Baikerikar, Sunil K. Surve, and Sapna U. Prabhu. "Comparison of Load Balancing Algorithms in a Grid"; Proceedings of the International Conference on Data Storage and Data Engineering (DSDE), IEEE Computer Society, 20—23, 2010.

[9] Youran Lan. "A Dynamic Load Balancing Mechanism for Distributed Systems"; Journal of Computer Science and Technology, Vol. No. 11, Issue No. 3, 192—207, 1996.

[10] Rupam Mukhopadhyay, Dibyajyoti Ghosh, and Nandini Mukherjee. "A Study on the Application of Existing Load Balancing Algorithms for Large, Dynamic, Heterogeneous Distributed Systems"; Proceedings of the 9th International Conference on Software Engineering, Parallel and Distributed Systems (SEPADS), World Scientific and Engineering Academy and Society, 238—243, 2010.

[11] Sagar Dhakal, Majeed M. Hayat, Jorge E. Pezoa, Cundong Yang, and David A. Bader. "Dynamic Load Balancing in Distributed Systems in the Presence of Delays: A Regeneration – Theory Approach"; IEEE Transactions on Parallel and Distributed Systems (IEEE Press), Vol No. 18, Issue No. 4, 485—497, Apr 2007.

[12] Mohammed Javeed Zaki, Wei Li, and Srinivasan Parthasarathy. "Customized Dynamic Load Balancing for a Network of Workstations"; Proceedings of the 5th IEEE International Symposium on High Performance Distributed Computing, IEEE Computer Society, 282—291, 1996.

[13] Md. Abdur Razzaque and Choong Seon Hong. "Dynamic Load Balancing in Distributed System: An Efficient Approach"; Online Available: http://networking.khu.ac.kr/publications/data/Dynamic%20Load%20Balancing%20in%20Distributed%20System%20An%20Efficient%20Approach.pdf

[14] Thomas Kunz. "The Influence of Different Workload Descriptions on a Heuristic Load Balancing Scheme"; IEEE Transactions on Software Engineering (IEEE Press), Vol. No. 17, Issue No. 7, 725—730, Jul 1991.

[15] Domenico Ferrari, and Songnian Zhou. "An Empirical Investigation of Load Indices for Load Balancing Applications"; Proceedings of the 12th IFIP WG 7.3 International Symposium on Computer Performance Modeling, Measurement and Evaluation, North-Holland Publishing, Netherlands, 515—528, 1988.

[16] Ali M. Alakeel. "A Guide to Dynamic Load Balancing in Distributed Computer Systems"; International Journal of Computer Science and Network Security, Vol. No.10, Issue No. 6, 153—160, Jun 2010.

[17] Paul Werstein, Hailing Situ, and Zhiyi Huang. "Load Balancing in a Cluster Computing"; Proceedings of the 7th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT), IEEE Computer Society, 569—577, 2006.

[18] Marta Beltran, Antonio Guzman, and Jose Luis Bosque. "Dealing with Heterogeneity in Load Balancing Algorithms"; Proceedings of the 5th International Symposium on Parallel and Distributed Computing, IEEE Computer Society, 123—132, 2006.

[19] Parveen Jain, and Daya Gupta. "An Algorithm for Dynamic Load Balancing in Distributed Systems with Multiple Supporting Nodes by Exploiting the Interrupt Service"; International Journal of Recent Trends in Engineering (Academy Publisher), Vol. No. 1, Issue No. 1, 232—236, 2009.

Time optimizing in Economical Grid Using Adaptive Stochastic Petri Net Based on Learning Automata

Intelligent approach in Economical Grid

Mohammad Shojafar
Msc. In Software Engineering, Computer & Elec. Dept.
Qazvin Islamic Azad University (Q.I.A.U)
Qazvin, Iran
Shojafar@qiau.ac.ir

Siamak Barzegar
Msc. In Software Engineering, Computer & Elec. Dept.
Qazvin Islamic Azad University (Q.I.A.U)
Qazvin, Iran
Barzegar@qiau.ac.ir

Mohammad Reza Meybodi
Computer Dept.
Amirkabir University Tech. (A.U.T)
Tehran, Iran
Mmeybodi@aut.ac.ir

***Abstract*—in this paper, intelligent algorithm of ALATO based on Learning Automata using Adaptive Stochastic Petri Net presented to optimize the time in Economical Grid. Adaptive Stochastic Petri nets based on learning automata predicts the following optimized state through getting information from former system states and dynamic environmental reactions, makes the system current states appeared based on that, changes the probability of events occurrences during the time and causes the events to be activated based on probability of occurrence. Comparing suggested algorithm and the existed algorithm has been shown in which the mentioned algorithm acts a significant decline in considered time in 200 considered independent tasks.**

Keywords-Economical Grid; Adaptive Stochastic Petri Net(ASPN); Learning Automata; Time Optimization

I. INTRODUCTION

Stochastic Petri Nets [1] are considered suitable tools for mathematical and graphical modeling. This tool can be used for modeling, description and analyzing of the systems which have concurrent, asynchronous, distributed, parallel, infinitive or accidental natures [2]. In fact, Petri nets are parts of those models which can show a system's manner and function.

One of the intelligent algorithms is Stochastic learning automata. Learning automata can be considered as a single object which has the limit numbers of actions [3, 4]. The function of this object includes this fact that every time one action is chosen from the actions collection and then will be evaluated in a stochastic environment. The taken respond from the environment will be used for the next action selection by automata and in this way, automata gradually recognize the optimum action. The method that is used to select another action by automata that is used to select another action by automata will be defined by the used learning algorithm. Stochastic Petri nets graphical features allow the simulated model to imagine the system complexity [5]. Stochastic Petri nets have a great application to analyze the Distributed systems.

Grid computing nets can be mentioned as one of these systems [6, 7]. In simply speaking, grid computing is distributed calculations which are got to the higher development level. The purpose is to make an image from a simple virtual computer and at the same time big and powerful which has its own management ability from a vast collection of computers. This collection includes services of heterogeneous related and complex systems which share various resources combinations. To manage these kinds of complex systems, the common approaches cannot be used to manage the resources which try to optimize the performance in the whole systems. In this approach, the sources owners follow the economical methods to manage their sources and to time the users' request to guarantee the considered service quality. Competitive economical models provide algorithms, policies and tools to share or allocated the sources in grid. The most common economical model is good market model. In this model, the sources have the prices which are defined based on supply and demand and value in economical system. Grid users request limited deal (the time of finishing program execution), Budget and the strategy of time optimizing as the demands of service quality. The economical scheduling , system in grid must allocate the grids source to the user's program (which consists of the great number of independent tasks) using an efficient algorithm as these limitations are observed and according to the requested time parameter becomes minimum paying attention to optimizing strategy.

In continue, an Adaptive stochastic Petri net based on learning automata is introduced and we study its application in economical grid. In this paper, stochastic learning automata used and tested types of LA in suggested idea are described in the second part. In the third section, the concept

of collation and Adaptive Stochastic Petri net(ASPN) is presented. In section four, there is an explanation about some performed algorithms to optimize the time in economical grids like BTO, AFBTO AND LATO algorithms and suggested algorithm of ALATO. The fifth section reviews the suggested algorithm with the previous methods in different situations and the results are figured out in a diagram with mentioning the details. In the last section, the achieved conclusion and suggested works are performed.

II. LEARNING AUTOMATA

Learning Automata is a machine with limited positions which can perform limited actions. Every selected action will be evaluated by a probable environment and a respond will be returned to the learning automata. Learning automata uses this response and choose its next action as "Fig. 1".

Learning automata can have fixed or variable structure. Learning automata can be shown by four $\{\alpha, \beta, p, T\}$ that α is the automata collection actions, β is an environment response set, p is the vector of selection probability of each action contain s probabilities and The function of T is the reinforcement algorithm, which modifies the action probability vector p with respect to the performed action and received response.. Reinforcement learning algorithm of learning automata makes the automata actions probability vector up to date in the model of S-L_{RP} [8]. Learning Automata algorithm is classified as S and P models. P-model or standard model is a position that the considered reward amount (getting a favorable response) and considered penalty amount (getting an unfavorable response)is a fixed and equal amount for every action, the general structure of learning automata in standard model is come in (1) and (2).

$$\begin{aligned} & p_i(n+1) = p_i(n) + a[1 - p_i(n)] \\ & p_j(n+1) = p(1-a)p_j(n); \\ & \forall j; j \neq i \end{aligned} \tag{1}$$

$$\begin{aligned} & p_i(n+1) = (1-b)p_i(n) \\ & p_j(n+1) = \frac{b}{1-r}(1-b)p_j(n); \\ & \forall j; j \neq i \end{aligned} \tag{2}$$

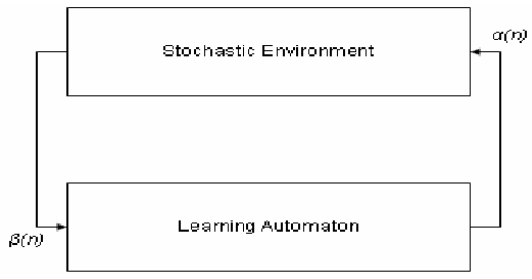


Figure 1. Learning Automata and Stochastic Environment

S-model of environment respond which is considered as a described function to improve automata has been added to the original learning automata. S-model is divided in to three classifications. a and b are reward and penalty parameters. When a and b are equal, the algorithm is called S-L_{RP}, When a>> b the algorithm is called S-L_{REP} and whenever b is zero the algorithm is called S-L_{RI} [9]. Learning Automata of S-L_{REP} is appeared with r actions and reward parameter of a and penalty parameter of b appear their vectors as following. If α_i would be chosen in nth repetition and the environment respond would be $\beta_i(n)$ to it, automata probabilities vector is appeared in (3).

$$\begin{aligned} & p_i(n+1) = p_i(n) + a(1-\beta_i(n)) \times \\ & (1 - p_i(n)) - b\beta_i(n)p_i(n) \\ & p_j(n+1) = p_j(n) - a(1-\beta_i(n))p_j(n) + \\ & b\beta_i(n)[\frac{1}{r-1} - p_j(n)] \quad \forall j \quad j \neq i \end{aligned} \tag{3}$$

III. ADAPTIVE STOCHASTIC PETRI NET(ASPN)

Although, there is investigation doing to explain the quality of forcing these nets into different system , a day by day interest becomes existed in the recent years to progress a new style and method which is prepared the field in adaptive features plan in Petri net huge models [10,11].

There are various approaches presented to make the stochastic Petri net adaptive. These kinds of approaches use a general development in the field of intelligent techniques like Artificial Neural Network (ANN), Fuzzy logic (FL), Knowledge Base system (KBS) and Stochastic Learning Automata (SLA). All the attempts have been done in order to improving and developing of the samples which can transfer the adjustable feature to the Petri nets. The performed works are classified into two general classes in this field .In the first class, which is called *Hybrid Fusion*, the investigations have been done in the recent years with the purpose of defining the basic conditions under which Petri net is able to be made to describe the adaptive actions similar to existed real system. In fact, there is a real system of performance which we want to make model artificially. This artificial model which is an illustration of the real world can be easily analyzed and revised and it is possible to perform the results in the real world. The reason of being adaptive in this class is related to basic changes of Petri net structuring and its getting homogenous with the real system frame. Some changes in Petri net basic structure have caused the ability and power of learning and teaching in Petri net using intelligent techniques. This class has some disadvantages too. Learning in Petri nets of this class causes increase in operation complexity and computing loads because it takes a lot of time to model all real system features in addition using the intelligent techniques in Petri net. In the second class, which is called *Combination Hybrid*, a quiet small researcher's group activate in the field of adaptive intelligent techniques usage and performance in addition to Petri net methodology in real world matters. This class models are

simpler than the first group models. However, sometimes their performing time is more. Also, the complexity measurement in designing is less than the first method. In this method, different technologies are put next to each other and the presented model in order to perform in the environment will be exerted with more profitability and accuracy by proportional changes in each one with regard to normal condition, this method uses the basic features of Petri Net in with intelligent techniques which is exerted to internal actions and requests of every part of a real system .Here, there is a suggestion of a veal and clear combination of intelligent features with system present condition in order to improvement of current system position [12,13].

In fact, adaptive stochastic Petri net is a kind of a stochastic Petri net which can predict the later system position with the help of present condition with regard to environmental information in other designing steps and gradually make it up to date with environmental condition Paying attention to the definition presented in this section about Adaptive Stochastic Petri Net (ASPN), we conclude that a stochastic Petri Net can be used in intelligent systems and the systems which are dynamic and have a lot of changes. Adaptive stochastic Petri Net performed in this paper is a part of the first algorithms that is the algorithms of Fusion Hybrid. We will explain the suggested algorithms i.e. ALATO based on learning automata.

IV. PROPOSED IDEA(ALATO)

In task scheduling in Economical grid, there are two features of resource *Time-Shared* or *Space-Shared* which have a lot of influences.

A. Time-Shared Resources

Everything which is left to a Time-shared resource immediately transfers to one of the processors for performance because time-shared processors can perform several works synchronously and as a matter of fact tasks use processors commonly.

This means that every task settles on a processor in a define time distance and then they will be stopped and the next task transfers to the processor to be performed. Every task, whose performance ends, will be exit from queue and the processor time will be equally divided between the remained tasks. One of problems that time-shared resources have is that the tasks performance finishing time cannot be unpredictable because there is always the probability of new task transferring and breaking of predicting done, one of the other problems of these resources is that while scheduling if the great number of tasks is left to them, they will be faced with overload position. In this case, because of concurrent performance of a lot of tasks in a Time-Shared processor, a small percentage of time processor allocates to every task and the stopping operation of one task and performance continuing of another task must be done a lot of times that causes to make overload in a processor and reduces the efficiency.

B. Space-Shared Resources

In a Space-Shared Resource, there is a queue of unprocessed tasks that as soon as a processor doesn't have anything to do (Idle), the first task of the queue transfers to it to be performed. So, each space-shared processor can perform one task in a time. This means first task of the queue transfers it to be performed. This means when a task transfers to processor for performance, it stands there continually up to the end and its performance will never be stopped. Here, the reviewed tasks is intelligent algorithm based on space-shared are considered independently and commonly. The tasks allocation way divides into two cases of all *at-once allocation* and *phased allocation*.

C. Phased Allocation

In phased Scheduling, scheduling operation becomes a phased operation which will continue up to the end of all tasks transferring. In this solution, task is left to every resource only equal to the number of the existed processors in it to avoid the concurrent performance of some tasks on one processor. Also, there is a queue for each user and the not transferred tasks of user program are kept in this queue. The task method is in this case that after entering a collection of independent tasks which make a a practical program belonged to the user, Scheduling system mapped all the tasks to the resources which is defined by scheduling algorithm; but the tasks allocating, tasks are left only in the number of not allocated processors. Then, all the extra tasks will be collected and returned to their own users. On the next stage, scheduling algorithm will be implemented again and the resources will be chosen to be mapped to which tasks are left in not allocated processors numbers. This operation will be continued until all tasks are transferred to the resources.

D. At-once Allocation

In At-once scheduling, the scheduler transfers the tasks to the resources which are defined by scheduling algorithm At-once and every resource takes the control of the number of performing tasks on its every process or (no processor doesn't perform more than one task at the same time) and regarding to its belonged tasks queue, it transfers them regularly to the processors which will be not allocated.

In fact, the difference between all at-once scheduling and phased scheduling is that in scheduling system which uses phased scheduling. Phased scheduling is that in scheduling system which uses phased scheduling method, the tasks are kept in laws related to their own users and are left to the resources gradually and through repeated stages. However, in at-once scheduling method, all the tasks are left to the resources at-once and tasks gradual leaving to the processor is done by the resources, themselves (every resource makes a queue for allocated tasks to itself). It means all resource Behaviors are like a space-shared resource. There are some algorithms presented in scheduling optimizing field to allocate the resource according to demands in Economical Grid. All algorithms purpose is to be able

to use the defined budget by the user in maximum and decrease the program performing time as possible. BTO algorithm is suggested to time optimizing by Buyya in [14, 15] and is evaluated in [16]. BTO is a heuristic algorithm presence that shows acceptable responds to the user for independent homogenous or little heterogeneous tasks. Also, Buyya considers the resources time-shared and causes some Complexity and defects in his algorithm. In the continue, Mr. Mahdavifar has suggested some algorithms for this after studying and simulation of various ways to optimize the time. He has suggested AEBTO and LATO algorithms [17, 18]. In continue a kind of algorithm named ALATO is presented to make load balance in tasks allocation into grid computing resources. BTO and AEBTO algorithms are heuristic algorithms and LATO and ALATO algorithms are intelligent algorithms and based on Learning Automata. AEBTO algorithm makes some changes in BTO algorithm which making better scheduling. LATO algorithm uses with Learning Automata try to minimize time for desired budget. ALATO algorithm changes in getting off method and paying reward and penalty in LATO algorithm and get the best result among the mentioned algorithms.

Here, we consider *Minimum* algorithm which is calculated with the least time. To get the minimum of calculation time, the presented algorithm can be used to optimize the time that we use scheduling operation for instructions as the smallest executive unit instead of scheduling the tasks. It means, we allocate some budget to every million instructions and leave them to the suitable resource to be presented. Another way to get the minimum of calculation time is to use the performed method to get the minimum of cost. To do this, we carry out the minimum cost method considering the performance respite and study the getting expresses.

Then, we lose some of the respite and perform the cost minimum method again. We repeat this operation as many times to get the time in which tasks doing cost minimum become almost equal to the defined budget. In this case, the used time in cost minimum method is calculation time minimum. We use this method to get the minimum of tasks performing time spending the defined budget by the user.

Suggested algorithms (ALATO) in this section is in the case of *Space-Shared* and uses all *At-once Scheduling* method and regulate the tasks queue regarding to length in a descending position to decrease the not allocated time of more expensive resources. At user's time entrance to the system, if his practical program doing is possible in the defined time and budget area, it will accept the user and guarantee the program ending obeying these areas. To do this, at the end of suggested algorithm and after distinguishing the tasks doing time total and program performing expenses which are approximately gained in scheduling.

In the case of user's acceptance, the distribution stage will be done in which the tasks are transferred to the defined resources.

Therefore, ALATO algorithm tries to fulfill the complicated operation of time optimizing with better efficiency than the heuristic algorithms. This algorithm uses a collection of learning automates that are attributed to the tasks. These learning automates have variable structures and their actions are in agreement with grid resources. In fact, learning automata belonged to a task selects the recourse to which the task must be allocate. When all the words choose their resources regularly, they will get reward or penalty from the environment. Then again, all the woks fulfill the resource selection to get the environment response. This operation repeats in definite numbers up to the time that all tasks find their suitable resources and be allocating them.

ALATO algorithm cannot catch the final scheduling with studying the tasks and scheduling for only once, but it does this operation by a lot of repetition and makes a new scheduling in every repetition. Environment responds the tasks (reward or penalty) regarding the resources they have chosen. Tasks use this environmental response in the following repetition in order to make the new existed scheduling closer to optimum scheduling. These repetitions occur in a great number till all the tasks find their suitable resources and don't have any other change in their decisions. Actually, repetitions continue up to the time that all the learning automatas become convergent and a harmonic scheduling is made in repetitions. This homogenous scheduling is final scheduling in whose base the tasks are transferred to the resources.

Also in ALATO algorithm, the made scheduling must be temporary done to the resources in every repetition to evaluate it. To do this, every resource include a temporary queue called *relation queue* which is emptied in the beginning of every repetition and then stands immediately in the relation queue when the tasks are selecting regularly.

The environment response to the learning automata belonged to tasks occurs regarding to ideal conditions in a time optimizing. To do this, environment uses the following points to give reward or penalty to the tasks:

1) Tasks executive expense total must not be more than budget defined by the user.

2) The execution time for left tasks to the busier resource must not be allotted different from performance time in other resources.

3) Every tasks mast is given to a resource which can finish its performance sooner than the other resources.

In this case, an environment punishes the tasks using two first points and rewards to them regarding the third point. According to the first point, it tasks performance total costs on their selected resources is more than the define budget, the environment will punish the tasks which need a lot of expenses to perform. In fact, punishing operation starts from tasks queue end in the most expensive used resource (the most expensive resource to whose one tasks is related at least once).

To punish every task and omit it from the selected resource related queue, the execution difference on this resource with its performance expense on the cheapest existed resource deducts from the total costs. The punishing operation continues till the total cost is more than the defined budget.

As a matter of fact, tasks are gaited toward using the cheaper recourses with this penalty. The purpose in this stage is decreasing the cost fast and takes it to a less amount measurement than the considered one to transfer the tasks from the most expensive resource in to the cheapest resource. To regard the second point, environment considers the tasks punishment in related queue of the most activated resource.

Pay attention to this point that the most activated resource is not necessarily the most expensive resource. We start from the last activated queue end.

We chose Y number tasks from the end queue of the most activated resources. If one of the used resources in scheduling (the resource which has at least one related task) except the last one (the most expensive) can process this task sooner than the most activated resource, environment will punish this task then it will choose more suitable resources for itself in the following repetitions unless this task won't be punished. This operation will be done for all Y tasks from the most activated resources. This operation causes that woks distribute with more balance and equilibrium among the recourses. The amount of Y is calculated according to "TABLE I" from (4) and (5).

$$X = (Task\_No - A)/(\mathrm{Res}\_No - 1) \quad (4)$$

$$Y = (B - X) * \frac{\mathrm{Res}\_No - 2}{\mathrm{Res}\_No - 1} \quad (5)$$

Finally environment rewards tasks according to the third point. The tasks which can get less finishing time or equal in compare with the previous repetition will get rewards. Of course, if they haven't punish in this repetition. So. Every task encourages choosing a source for its transferring which ends its performance in the least time.

As it is figured out in "Fig 2", ALATO algorithm firstly arranges the tasks based on their length in a descending way. Then in a lot of repetitions, finds the suitable resource to perform each task. This algorithm finally transfers tasks to those resources and uses all at once scheduling method.

All models of learning automata (Standard Model and S-Model Methods) are studied to achieve the best answer. In standard automata environment uses two penalty response and one reward response during algorithm learning. Since a task isn't punished in any of two cases and also gets less or equal ending in compare with the previous repetition, probably it had found a suitable source for itself. In this case, the reward amount is (0.1) more than the amount of penalties in the

TABLE I. THE ABBREVIATION SIGNS FOR ALATO

Acronym	Details Types
Tasks_No	Task Number
Res_No	Resource Number
A	Task Numbers Allocated To The Most Expensive Resource
B	Task Numbers Allocated To busiest Resource

***1. Arrangement**: Arrange the tasks according to their length in a descending way.*
***2. Learning**: Repeat 10000 times:*
a. Make the resource relation queue empty
*b. **Selection**: Do this for each task in order:*
i : choose resource to transfer using related learning automata
ii: Put the task in selected in source relation queue.
*c. **Penalty (1)**: Repeat since the total cost of tasks performance on selected resources are more than the defined budget:*
i: Divide one task at the relation queue ending in the most expensive used resource.
ii: Decrease the difference of the task performance expenses on this resource with the task executive cost on the cheapest existed resource from the total cost of tasks performance.
iii : Fine the task (Penalty amount : 0.01)
*d. **Penalty (2)**: Repeat Y times:*
i: Consider the last related task to the most activated resource.
ii: Fine the task in one of the used resources lout the last one (the most expensive used resource) in able to finish this task performance sooner (Penalty amount : 0.05)
*e. **Reward**: Do for every task:*
If it isn't fined: if the finishing time of its performance on selecting resource is less than or equal to the finishing time in previous repetition, reward it. (Reward amount: 0.1)
***3. Scheduling**: choose a resource for each task in order using the learning automata be longed to it and record the task to that resource.*

Figure 2. ALATO Algorithm

First case, it's (0.01) more and in the second case, it is (0.05) more. The cause for being more in penalty amount in the second case is because of putting the smaller tasks in more activated resources and increasing in that recourse accumulation. In fact, the tasks go towards the putting in less activated resources and increasing in load balance in every step by increasing the tasks penalty and automata convergent speed is increased significantly. We have tested the learning automata S-model with various amount for reward and penalty rate which is come in simulation section.

In the next part, Minimum, LATO, AEBTO, BTO and ALATO suggested algorithm are studied in different cases of Learning Automata.

V. SIMULATION AND PERFORMANCE EVALUATION

We describe a task model in CSPL language of SPNP software in this performance. A Sample Formatting is used in tests for grid resources named GRI that of course is not existed in the outside world.

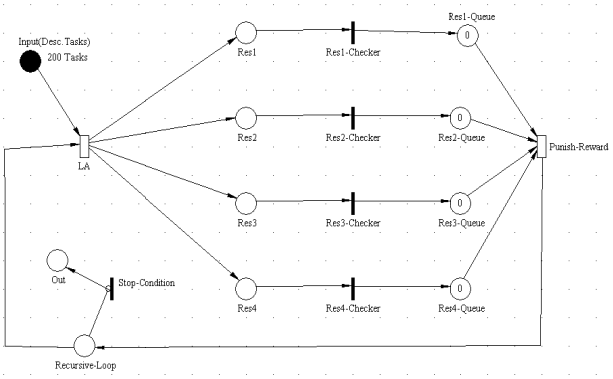


Figure 3. To allocate the tasks to the resources and applying the reward, Penalty and performance 10000 times

A sample of a real formatting named WWG is come in [19] which are also used in experiments. In this formatting four resources with different performance rate and useful prices are used that are arranged ascend through cost. In this simulation. 200 tasks amount with the various lengths is studied in a defined time by the user in various non-homogeneous budgets for four resources.

In presented adaptive Stochastic Petri Net (ASPN), every token shows one task of user's tasks. Every token include the information like task length, priority number based on its length and learning automata.

In "Fig. 3", every token choose an action (one of the exited resource in grid net) based on its probability vector. Then all the allocated tasks in each of the resources based on the capacity of that resource, the time user defined to do that task and the algorithm suggested in previous section will get reward or penalty. Our four resources are *Res1, Res 2, Res3,* and *Res4* for allocating action that each resource has distinct queue that located in order *Res1_Queue, Res2_Queue, Res3_Queue* and *Res4_Queue*. In every tasks allocation to every resource the tasks number located in its resource allocated queue and transition Punish_Reward is for penalty/reward action for each allocated resources. This may is done up to 10000 times on the resources in a returning manner and the gained results of every performance go to the next executive stage after storing in *Recursive-Loop* state.

In Transition LA, S-model Learning Automata is considered for all four resources. We have tested the S model LA with various amounts for reward and penalty rate. In S model methods, if action α_i is chosen in the nth repetition the environment response (undesired response) is $\beta_i(n)=1$ to it and environment response (desired response) to it illustrated in (6).

$$B_i(n)\frac{1}{1+\frac{\Pr evous\,time}{New\,time}} \qquad (6)$$

It means it gains better response in Compare with the cheaper resource selecting.

According to the results we will observe that S-L $_{Rep}$ method will have the best result with the reward rate equals to 0.1 and penalty rate equals to 0.05.

Every task has its own specific length which is stated according to million instructions (MI). We always consider the tasks length valid, in this case that on area (Maximum ... Minimum) will be chosen for tasks length. The for every task length, an amount is chosen from this area as the tasks length distribution is UNIFORM. For the most homogeneous case, we consider the tasks length distribution area of (100.000 ... 110000) and for the most heterogeneous case the area of (10000...200000). The conditions of doing simulation experiment are figured out in "TABLE II". The tasks length area parameter is various and the experiments are done in different areas (it means different Heterogeneous). The parameter of the number of experiments distinguishes that 20 tests are done to gain the time amount of algorithms performance in a definite Heterogeneous and them the average amount is selected.

In "Fig. 4", there is a comparison between the mentioned automates in LATO and ALATO algorithms in addition to Minimum algorithm. Here, the best time to perform each automaton is considered 150000 for the budget. ALATO algorithm is improved a lot according the time and is very close to the optimum case comparing with LATO algorithm. S-L$_{Rep}$ model spends less time in compare with the other automates.

TABLE II. TIME OPTIMIZING TEST CONDITIONS IN DIFFERENT HETEROGENEOUS AND 150000 BUDGET

Parameter	Value	Parameter	Value
Algorithm Type	Alternative	Time	20000
Res. Config.	GR1	Budget	150000
User No.	1	Task Length	Alternative
Exp. No	20	Task No.	200

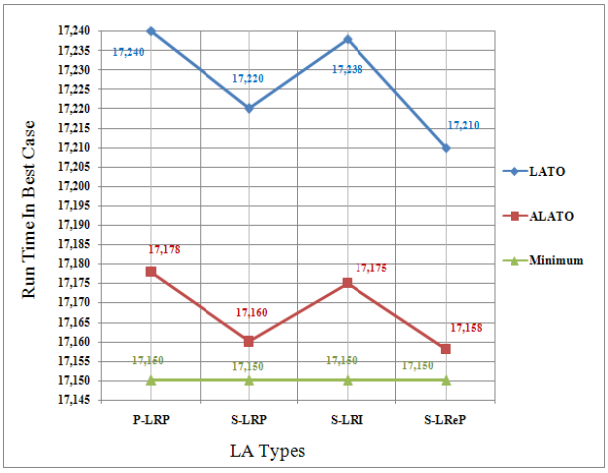


Figure 4. Comparing theRun Time of LA's with the Fixed Budget of 150000 in the Best Situation of Every LA

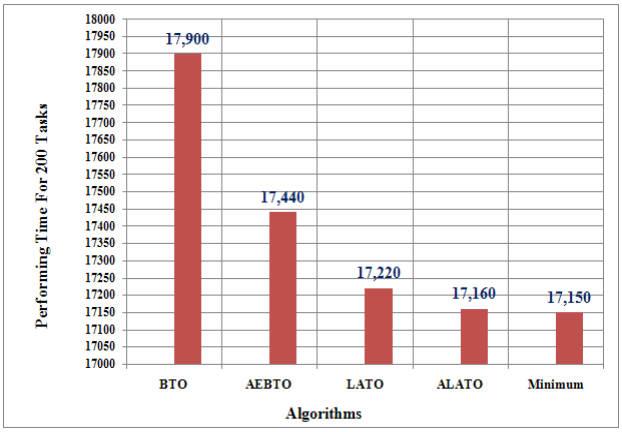


Figure 5. The Performing Time in an Acceptable Situation of S-L_{RP} Automata with te Fixed Budget of 150000

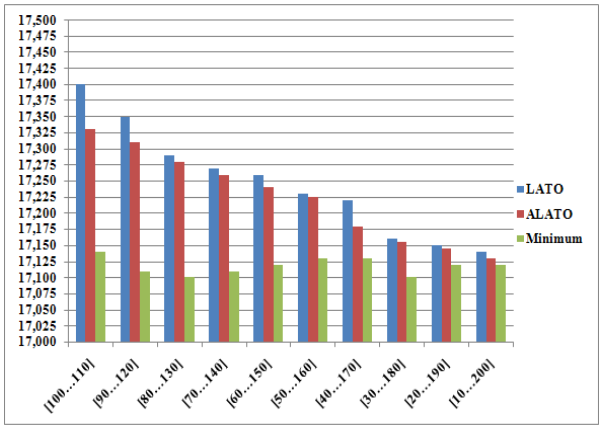


Figure 6. Comparing LATO, ALATO and Minimum Algorithms in Different Non-homogeneity

This is because of this fact that whatever the reward amount is more than penalty, the time length is more possible to make update and choose resource with the less repetition. S model shows the better result comparing with P-Model. Because, Considering the smaller changes amount in β_i makes the automata more accurate and proving time of resources selection probabilities will be possible with less repetition and as a result the time for automata to be constant is more that the P case.

"Fig. 5" illustrates the performing time duration in an acceptable situation of S-L_{Rp} LA and the case which penalty is 0.05 and the reward is 0.1. As you see,

ALATO algorithm acts better than LATO algorithm about 60 units and it has just 10 units difference from Minimum or Optimum case.

Because S-Model automata present a better time comparing with Standard LA, this automata is chosen to study. In S-Model types, S-L_{Rep} automata have presented a better result, so, we evaluate the results in these automata. We have done 20 tests for every heterogeneous and considered the average amount implemented for each algorithm as the enrage time to perform the tasks.

LATO and ALATO algorithms that use the learning automata and ALATO as the best suggested algorithm wants to improve LATO algorithm. Here, we review the tasks heterogeneous effect on improvement measure resulted from LATO and ALATO algorithm.

"Fig. 6" shows the results of these two algorithms in addition to minimum algorithm next to each other and different homogeneities. It is observed that the improvement resulted from ALATO and LATO algorithms scheduling becomes more with increasing the tasks Non-homogeneity. The time decreasing trend in ALATO algorithm is more than LATO algorithm.

In fact, the speed of getting Minimum in ALATO is more than LATO.

As you see, ALATO algorithm is very close to optimized algorithm (Minimum) Comparing with LATO heterogeneous.

VI. Conclusion and Future Work

In this paper an Adaptive Stochastic Petri Net is presented to improve the time in economical grid helping learning automata. This presented net is a part of fusion Hybrid class. As it is state in previous sections, there are some disadvantages in this class. Most of the time learning in Petri nets from this class cause increasing in operation complexity and rise in overhead calculation. As it is also reviewed in section two of this paper, Adaptive Stochastic Petri Net use intelligent techniques like artificial neural nets, Fuzzy logic and knowledge based system, but the Adaptive Stochastic Petri Net that we are presented is based on learning automata. Adaptive Stochastic Petri Net based on learning automata doesn't need any training with some pre-defined models in contrast with Adaptive Petri Net that use artificial neural nets. In the systems in which the environment is unknown and changing every time, there can't be any model with which the artificial neural net can be trained lout suggested adaptive model in these environments are completely flexible because learning automata doesn't need any training with the previous data, so, it doesn't have the fusion Hybrid class problem which is increasing in calculating overhead. Besides, in Adaptive Stochastic Petri Net based on learning automata, every token has its specific automata and in contrast with previous adaptive models that need to make extra place and passes to make artificial neural net, there is no need to any newer place and pass and it causes non-complexity in planning and raise in calculating overhead.

Also, using time optimizing algorithms simulation, it is distinguished that ALATO learning algorithm in a lot of tasks heterogeneous gets better results comparing

with the heuristic algorithms and it can perform the practical plan be longed to user on the grid in less time comparing with the other algorithms spending the defined budget. Also, regarding to the small amount of gained time from this algorithm with the minimum of calculating time, there is no significant possibility to make the ALATO algorithm improved.

The future tasks which will be stated in this field we can mention using presented Adaptive Stochastic Petri Net in demands scheduling and the quality of their allocation in economical grid base on techniques like bargaining, auctions, calling for tenders and barter trade.

REFERENCES

[1] J. Peterson, Petri Net Theory and the Modeling of Systems, Perantic Hall, Englewood Cliffs, NJ, Vol. 1, 1981.

[2] W. Reisig, Petri Nets: An Introduction, EATCS Monographs on Theoretical Computer Science, Springer Verlag, Vol. 1, 1985.

[3] R. Boppana and M. M. Halldorsson, "Approximating maximum independent sets by excluding subgraphs," BIT Magazine, Published By BIT Computer Science and Numerical Mathematics, ISSN:0006-3835, Vol. 32, Issue 2, 1992, pp.180-196. DOI: http://dx.doi.org/10.1007/BF01994876

[4] T. N. Bui and P. H. Eppley, "A hybrid genetic algorithm for the maximum clique problem," Proceedings of 6th International Conference on Genetic Algorithms, Published By Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, ISBN: 1-55860-370-0, 1995, pp. 478-484.

[5] C. Hirel, S. Wells, R. Fricksy and K. S. Trivedi, "ISPN: An Integrated Environment for Modeling Using Stochastic Petri Nets,", Center for Advanced Computing and Communication Department of Electrical and Computer Engineering Duke University, Durham, NC 27708-0291, 2000.

[6] R. D. Venkataramana and N. Ranganathan, "Multiple Cost Optimization for Task Assignment in Heterogeneous Computing Systems Using Learning Automata," Heterogeneous Computing Workshop (HCW'99), 1999, pp. 137-145.

[7] J. Moore and L. Hahn, "Petri Net Modeling of High-Order Genetic Systems Using Grammatical Evolution," Bio Systems 72, 2003, pp. 177–186.

[8] K. S. Narendra and M. A. L. Thathachar, Learning automata: An introduction, Prentice Hall, 1989.

[9] A.S. Poznyak and K. Najim, Learning Automata and Stochastic Optimization, ISBN 3-540-76154-3 Springer-Verlag Berlin Heidelberg New York, 1997.

[10] T. Murata, "Some recent applications of high-level Petri nets," IEEE International Symposium on Circuit and System, ISBN: 0-7803-0050-5, Vol. 2, Old Version (1991), 2002, pp. 818-821. DOI: http://dx.doi.org/10.1109/ISCAS.1991.176488

[11] M. C. Zhou and M. D. Jeng, "Modeling, analysis, simulation, scheduling, and control of semiconductor manufacturing systems: A Petri net approach," IEEE Transaction on Semiconductor Manufacturing, ISSN: 0894-6507, Vol. 11, Issue 3, Old Version (1998), 2002, pp. 333-357. DOI: http://dx.doi.org/10.1109/66.705370

[12] R. Khosla and T. Dillon, "Intelligent hybrid multi-agent architecture for engineering complex systems," International Conference on Neural Networks, Houston, TX, and ISBN: 0-7803-4122-8, Vol. 4, Old Version (1997), 2002, pp. 2449-2454. DOI: http://dx.doi.org/10.1109/ICNN.1997.614540

[13] L. C. Jain and N. M. Martin, Fusion of Neural Networks, Fuzzy Sets, and Genetic Algorithms: Industrial Applications, ISBN: 0849398045, First edition, CRC Press, FL, USA, 1998. DOI: http://www.amazon.com/exec/obidos/ASIN/0849398045/acmorg-20

[14] R. Al-Ali, O. Rana, D. Walker, S. Jha and S. Sohail, "G-QOSM: Grid Service Discovery Using QOS Properties," Computing and Informatics Journal, Special Issue on Grid Computing, Vol. 21, No. 4, 2002, pp. 363–382.

[15] R. Buyya, Economic-Based Distributed Resource Management and Scheduling for Grid Computing, Ph.D. Thesis, School of Computer Science and Software Engineering, Monash University, Melbourne, Australia, 2002.

[16] K. Czajkowski, I. Foster, C. Kesselman, V. Sander and S. Tuecke, "SNAP: A Protocol for Negotiating Service Level Agreements and Coordinating Resource Management in Distributed Systems," 8Th Workshop on Job Scheduling Strategies for Parallel Processing, 2002.

[17] Y. Mahdavifar and M. R. Meybodi,"Time Optimization in Economic Computational Grids Using Learning Automata," Proceedings of the First Iranian Data Mining Conference, Amirkabir University of Technology, Tehran, Iran, 2007.

[18] Y. Mahdavifar and M. R. Meybodi, "Cost-Time Optimization in Economic Computational Grids," Proceedings of the Third Information and Knowledge Technology, Ferdowsi University of Mashad, Mashad, Iran, 2007.

[19] R. Buyya, "The World-Wide Grid (WWG)," 2002. http://www.buyya.com/ecogrid/wwg/

A New Approach for Solving Nonlinear Fredholm-Volterra-Hammerestein Integral Equations Based on Cubic B-Spline Wavelets

K. Maleknejad[1], **and M. Nosrati Sahlan**[2]
[1,2]School of Mathematics, Iran University of Science and Technology, Tehran, Iran.

Abstract— *In this work, a computational method for solving nonlinear Volterra-Fredholm-Hammerestein integral equations is proposed. Compactly supported semiorthogonal cubic B-spline wavelets are employed as basis functions then collocation method is utilized to reduce the computation of integral equations to some algebraic system. The method is computationally attractive, and applications are demonstrated through illustrative example.*

Keywords: Fredholm-Volterra-Hammerstein integral equations, collocation method , cubic B-spline wavelets, error considereation, function approximation

1. Introduction

The past two decades have been witnessing a strong interest among physicists, engineers and mathematicians for the theory and numerical modeling of integral equations. These equations are solved analytically; for example in [1-2] and the references therein. Although for proving existence and uniqueness of solution of integral equation has been done lots of researches [3-6], but analytical solutions for those often are not available. Nonlinear integral equations have been studied in relation to physics, vehicular traffic, biology, the theory of optimal control, economics, etc.
Several numerical methods for approximating the solution of this class of equations are known. For Fredholm-Hammerstein integral equations, the classical method of successive approximations was introduced in [2]. First kind Fredholm integral equations are solved in [7] by Tikhonov regularization. In [8] Brunner applied a collection type method for nonlinear Volterra-Hammerstein integral equations and integro-differential equations and discussed its connection with the iterated and collection methods. Guoqiang [9] introduced and discussed the asymptotic error expansion of a collection type method for Volterra-Hammerstein integral equations. The methods in [10-15] transform a given integral equation into a system of nonlinear equations, which has to be solved with some kind of an iterative method.
Consider the second kind nonlinear Fredholm-Volterra-Hammerstein integral equation of the form

$$\begin{aligned} y(x) =& f(x) + \int_0^1 K_1(x,t) g_1(t, y(t)) dt \\ &+ \int_0^x K_2(x,t) g_2(t, y(t)) dt, \quad 0 \le x, t \le 1, \end{aligned} \tag{1}$$

where f, K_1 and K_2 are known L^2 functions, with $g_1(t, y(t))$ and $g_2(t, y(t))$ nonlinear in y, the unknown function that to be determined. Among conventional numerical methods for solving integral equations, the collocation method receives more favorable attention from engineering applications due to lower computational cost in generating the coefficient matrix of the corresponding discrete equations.

2. Cubic B-spline Scaling and Wavelet Functions

The general theory and basic concepts of the wavelet theory and MRA is given in [16-21].Wavelets and scaling functions are defined on the entire real line so that they could be outside of the integration domain. This behavior may be require an explicit enforcement of the boundary conditions. In order to avid this occurrence, semiorthogonal compactly supported spline wavelets, constructed for the bounded interval $[0, 1]$, have been taken into account in this paper. These wavelets satisfy all the properties verified by the usual wavelets on the real line.
definition Let m and n be two positive integers and

$$\begin{aligned} a = x_{-m+1} =& \ldots = x_0 < x_1 < \ldots < x_n = x_{n+1} \\ &= \ldots = x_{n+m-1} = b, \end{aligned} \tag{2}$$

be an equally spaced knots sequence. The functions

$$\begin{aligned} B_{m,j,X}(x) = & \quad \tfrac{x - x_j}{x_{j+m-1} - x_j} B_{m-1,j,X}(x) \\ & + \quad \tfrac{x_{j+m} - x}{x_{j+m} - x_{j+1}} B_{m-1,j,X}(x), \\ & j = -m+1, \ldots, n-1, \end{aligned} \tag{3}$$

and

$$B_{1,j,X}(x) = \begin{cases} 1 & x \in [x_j, x_{j+1}), \\ 0 & otherwise, \end{cases} \tag{4}$$

are called cardinal B-spline functions of order $m \ge 2$ for the knot sequence $X = \{x_i\}_{i=-m+1}^{n+m-1}$, and

$Supp\ [B_{m,j,X}(x)] = [x_j, x_{j+m}] \bigcap [a,b]$.
For the sake of simplicity , suppose $[a,b] = [0,n]$ and $x_k = k, k = 0, ..., n$. The $B_{m,j,X} = B_m(x-j), j = 0, ..., n-m$, are interior B-spline functions, while the remaining $B_{m,j,X}, j = -m+1, ..., -1$ and $j = n-m+1, ..., n-1$ are boundary B-spline functions, for the bounded interval $[0,n]$. Since the boundary B-spline functions at 0 are symmetric reflections of those at n, it is sufficient to construct only the first half functions by simply replacing x with $n-x$.
By considering the interval $[a,b] = [0,1]$, at any level $j \in Z^+$, the discretization step is 2^{-j}, and this generates $n = 2^j$ number of segments in $[0,1]$ with knot sequence

$$X^{(j)} = \begin{cases} x^{(j)}_{-m+1} = ... = x^{(j)}_0 = 0, & \\ x^{(j)}_k = \frac{k}{2^j} & k = 1, ..., n-1, \\ x^{(j)}_n = ... = x^{(j)}_{n+m-1} = 1. & \end{cases} \tag{5}$$

Let j_0 be the level for which $2^{j_0} \geq 2m-1$; for each level $j \geq j_0$ the scaling functions of order m can be defined as follows:

$$\varphi^{(j)}_{m,k}(x) = \begin{cases} B_{m,j_0,k}(2^{j-j_0}x) & k = -m+1, ..., -1 \\ B_{m,j_0,2^j-m-k}(1-2^{j-j_0}x) & k = 2^j-m+1, ..., 2^j-1 \\ B_{m,j_0,0}(2^{j-j_0}x - 2^{-j_0}k) & k = 0, ..., 2^j-m. \end{cases} \tag{6}$$

And the two-scale relation for the m-order semi orthogonal compactly supported B-wavelet functions are defined as follows:

$$\psi_{m,j,i-m} = \sum_{k=i}^{2i+2m-2} q_{i,k} B_{m,j,k-m}\ ,\ i = 1, ..., m-1, \tag{7}$$

$$\psi_{m,j,i-m} = \sum_{k=2i-m}^{2i+2m-2} q_{i,k} B_{m,j,k-m}\ ,\ i = m, ..., n-m+1, \tag{8}$$

$$\psi_{m,j,i-m} = \sum_{k=2i-m}^{n+i+m-1} q_{i,k} B_{m,j,k-m}\ ,\ i = n-m+2, ..., n, \tag{9}$$

where $q_{i,k} = q_{k-2i}$.
Hence, there are $2(m-1)$ boundary wavelets and $(n-2m+2)$ inner wavelets in the boundary interval $[a,b]$. Finally by considering the level j with $j \geq j_0$, the B-wavelet functions in $[0,1]$ can be expressed as follows:

$$\psi_{m,j,i}(x) = \begin{cases} \psi_{m,j_0,i}(2^{j-j_0}x) & i = -m+1, ..., -1 \\ \psi_{m,2^j-2m+1-i,i}(1-2^{j-j_0}x) & i = 2^j-2m+2, ..., 2^j-m \\ \psi_{m,j_0,0}(2^{j-j_0}x - 2^{-j_0}i) & i = 0, ..., 2^j-2m+1. \end{cases} \tag{10}$$

The scaling functions $\varphi^{(j)}_{m,k}(x)$, occupy m segments and the wavelet functions $\psi^{(j)}_{m,i}(x)$ occupy $2m-1$ segments.

Therefore the condition $2^j \geq 2m-1$, must be satisfied in order to have at least one inner wavelet.
Cubic B-spline scaling function $B_4(x)$ is given by [11]:

$$B_4(x) = \varphi_4(x) = \begin{cases} \frac{1}{6}x^3 & x \in [0,1) \\ \frac{1}{6}(-3x^3+12x^2-12x+24) & x \in [1,2) \\ \frac{1}{6}(3x^3-24x^2+60x-4) & x \in [2,3) \\ \frac{1}{6}(4-x)^3 & x \in [3,4) \\ 0 & otherwise, \end{cases} \tag{11}$$

and its *two-scale dilation equation* defined as follows:

$$\varphi_4(x) = \sum_{k=0}^{4} \frac{1}{8} \begin{pmatrix} 4 \\ k \end{pmatrix} \varphi_4(2x-k).$$

In this section, the scaling functions used in this work, for $j_0 = j = 3$ and $m = 4$, are reported :

Boundary scalings

$$\varphi^{(3)}_{4,-3}(x) = \begin{cases} (1-8x)^3 & x \in [0,1/8) \\ 0 & otherwise, \end{cases} \tag{12}$$

$$\varphi^{(3)}_{4,-2}(x) = \begin{cases} 896x^3 - 288x^2 + 24x & x \in [0,1/8) \\ 2(1-4x)^3 & x \in [1/8,2/8) \\ 0 & otherwise, \end{cases} \tag{13}$$

$$\varphi^{(3)}_{4,-1}(x) = \begin{cases} -\frac{1408}{3}x^3 + 96x^2 & x \in [0,1/8) \\ \frac{896}{3}x^3 - \frac{576}{3}x^2 + 36x - \frac{3}{2} & x \in [1/8,2/8) \\ -\frac{1}{6}(8x-3)^3 & x \in [2/8,3/8) \\ 0 & otherwise, \end{cases} \tag{14}$$

$$\varphi^{(j)}_{4,k}(x) = \varphi^{(3)}_{4,k}(2^{j-3}x)\ ,\quad k = -3, -2, -1\ ,\quad j = 3, 4, \ldots. \tag{15}$$

$$\varphi^{(3)}_{4,5}(x) = \varphi^{(3)}_{4,-1}(1-x), \tag{16}$$

$$\varphi^{(3)}_{4,6}(x) = \varphi^{(3)}_{4,-2}(1-x), \tag{17}$$

$$\varphi^{(3)}_{4,7} = \varphi^{(3)}_{4,-3}(1-x), \tag{18}$$

$$\varphi^{(j)}_{4,2^j-k-3}(x) = \varphi^{(3)}_{4,k}(2^{j-3}x)\ ,\quad k = -3, -2, -1\ ,\quad j = 3, 4, \ldots. \tag{19}$$

Inner scalings

$$\varphi^{(3)}_{4,0} = \begin{cases} \frac{256}{3}x^3 & x \in [0,1/8) \\ -256x^3 + 128x^2 - 16x + \frac{2}{3} & x \in [1/8,2/8) \\ 256x^3 - 256x^2 + 80x - \frac{22}{3} & x \in [2/8,3/8) \\ \frac{32}{3}(1-2x)^3 & x \in [3/8,1/2) \\ 0 & otherwise, \end{cases} \tag{20}$$

$$\varphi_{4,1}^{(3)} = \begin{cases} \frac{1}{6}(8x-1)^3 & x \in [1/8, 2/8) \\ -256x^3 + 224x^2 - 60x + \frac{31}{6} & x \in [2/8, 3/8) \\ 256x^3 - 352x^2 + 156x - \frac{131}{6} & x \in [3/8, 1/2) \\ \frac{1}{6}(5-8x)^3 & x \in [1/2, 5/8) \\ 0 & otherwise, \end{cases} \tag{21}$$

$$\varphi_{4,2}^{(3)} = \begin{cases} \frac{4}{3}(4x-1)^3 & x \in [2/8, 3/8) \\ -256x^3 + 320x^2 - 128x + \frac{50}{3} & x \in [3/8, 1/2) \\ 256x^3 - 448x^2 + 256x - \frac{142}{3} & x \in [1/2, 5/8) \\ \frac{1}{6}(6-8x)^3 & x \in [5/8, 3/4) \\ 0 & otherwise, \end{cases} \tag{22}$$

$$\varphi_{4,3}^{(3)} = \begin{cases} \frac{1}{6}(8x-3)^3 & x \in [3/8, 1/2) \\ -256x^3 + 416x^2 - 220x + \frac{229}{6} & x \in [1/2, 5/8) \\ 256x^3 - 544x^2 + 380x - \frac{220}{3} & x \in [5/8, 3/4) \\ \frac{1}{6}(7-8x)^3 & x \in [3/4, 7/8) \\ 0 & otherwise, \end{cases} \tag{23}$$

$$\varphi_{4,4}^{(3)} = \begin{cases} \frac{1}{6}(8x-4)^3 & x \in [1/2, 5/8) \\ -256x^3 + 512x^2 - 336x + \frac{109}{3} & x \in [5/8, 3/4) \\ 256x^3 - 640x^2 + 528x - \frac{430}{3} & x \in [3/4, 7/8) \\ \frac{1}{6}(8-8x)^3 & x \in [7/8, 1) \\ 0 & otherwise, \end{cases} \tag{24}$$

$$\varphi_{4,k}^{(j)}(x) = \varphi_{4,k}^{(3)}(2^{j-3}x - k), \quad k = 0, 1, ..., 2^j - 4, \quad j = 3, 4, \ldots. \tag{25}$$

Two scale delation equation for cubic B-spline wavelet is given by:

$$\psi_4(x) = \sum_{k=0}^{10} \frac{(-1)^k}{8} \sum_{l=0}^{4} \begin{pmatrix} 4 \\ l \end{pmatrix} \varphi_8(k-l+1)\varphi_4(2x-k). \tag{26}$$

Other inner and boundary wavelets are made by Eqs. (7)-(11).

3. Function Approximation

A function $f(x)$ defined over $[0, 1]$ may be approximated by cubic B-spline wavelets as:

$$f(x) = \sum_{i=-3}^{2^{j_0}-1} c_{j_0,i}\varphi_{j_0,i}(x) + \sum_{k=j_0}^{\infty} \sum_{j=-3}^{2^k-4} d_{k,j}\psi_{k,j}(x), \tag{27}$$

where $\phi_{j_0,i}$ and $\psi_{k,j}$ are scaling and wavelets functions, respectively. If the infinite series in Eq. (27) is truncated, then it can be written as:

$$f(x) \simeq \sum_{i=-3}^{i=2^{j_0}-1} c_{j_0,i}\varphi_{j_0,i}(x) + \sum_{k=j_0}^{j_u} \sum_{j=-3}^{2^k-4} d_{k,j}\psi_{k,j}(x),$$

or in abstract form

$$f(x) \simeq C^T \Upsilon(x) \tag{28}$$

where C and Ψ are $(2^{(j_u+1)} + 3) \times 1$ vectors given by

$$C = [c_{-3}, ..., c_7, d_{3,-3}, ..., d_{3,4}, ..., d_{j_u,-3}, ..., d_{j_u,2^{j_u}-4}]^T, \tag{29}$$

$$\Psi = [\varphi_{4,-3}^{(3)}, ..., \varphi_{4,7}^{(3)}, \psi_{4,-3}^{(3)}, ..., \psi_{j_u,-3}^{(3)}, ..., \psi_{j_u,2^{j_u}-4}^{(3)}]^T, \tag{30}$$

with

$$c_i = \int_0^1 f(x)\tilde{\varphi}_{4,i}^{(3)}(x)dx \quad , \quad i = -3, ..., 7, \tag{31}$$

$$d_{j,k} = \int_0^1 f(x)\tilde{\psi}_{4,k}^{(j)}(x)dx \ , \ j = 3, ..., j_u \ , \ k = -3, ..., 2^{j_u} - 4, \tag{32}$$

where $\tilde{\varphi}_{4,i}^{(3)}$ and $\tilde{\psi}_{4,k}^{(j)}$ are dual functions of $\varphi_{4,i}^{(3)}, i = -3, ..., 7$ and $\psi_{4,k}^{(j)}, j = 3, ..., j_u$, respectively. These can be obtained by linear combinations of $\varphi_{4,i}^{(3)}$ and $\psi_{4,k}^{(j)}$.

4. Nonlinear Mixed Fredholm-Volterra-Hammerstein Integral Equations

In this section, we solve the integral equation of the form (1) by using collocation method based on cubic B-spline wavelets. The unknown functions in Eq. (1) can be expanded in term of the selected scaling and wavelet functions as follows:

$$y(x) = C_y^T \Upsilon(x),$$

$$g_1(x, y(x)) = Z_1(x) = C_{Z_1}^T \Upsilon(x),$$

$$g_2(x, y(x)) = Z_2(x) = C_{Z_2}^T \Upsilon(x),$$

by substituting this expressions in Eq. (1) and computing the residual function, we get

$$\begin{aligned} R(x) = C_y^T \Upsilon(x) - f(x) \\ - \int_0^1 K_1(x,t) C_{Z_1}^T \Upsilon(t) dt \\ - \int_0^1 K_2(x,t) C_{Z_2}^T \Upsilon(t) dt. \end{aligned}$$

To find the solution $y(x)$ in (31), we collocate the following equations

$$\begin{aligned} &(i) \quad R(x) = 0, \\ &(ii) \quad g_1(x, C_y^T \Upsilon(x)) = C_{Z_1}^T \Upsilon(x), \\ &(iii) \quad g_2(x, C_y^T \Upsilon(x)) = C_{Z_2}^T \Upsilon(x), \end{aligned}$$

in $x_m = \frac{m}{2^{(j_u+1)}+3}, m = 1, 2, ..., 2^{(j_u+1)} + 3$. The current equations generate a set of $3 \times (2^{(j_u+1)} + 3)$ algebraic equations that could be easily solved by some iterative methods.

5. Convergence and error estimate

In this section, we found an error bound for the presented method.

Theorem 1([14]) : We assume that $f \in C^4[0,1]$ is represented by cubic B-spline wavelets as Eq. (28), where ψ has 4 vanishing moments, then

$$|d_{j,k}| \leq \alpha\beta\frac{2^{-5j}}{4!}, \tag{33}$$

where $\alpha = \max|f^{(4)}(t)|_{t\in[0,1]}$ and $\beta = \int_0^1 |x^4\tilde{\psi}_4(x)|dx$.□

Theorem 2([14]): Consider the previous theorem assume that $e_j(x)$ be error of approximation in V_j, then

$$|e_j(x)| = O(2^{-4j}). \quad \square$$

Thus, order of error depend on the level j. Obviously, for larger level of j, the error of approximation will be smaller.

Theorem 3 ([22,23]): For the $m-$th order B-spline wavelet the approximation error decreases with the $m-$th power of the scale 2^j,

$$\|f - P_j f\| \leq C2^{-jm}\|f^{(m)}\|.$$

Specifically we can derive the following asymptotic relation [15],

$$\lim_{j\to\infty}\|f - P_j f\| = C_m 2^{-jm}\|f^{(m)}\|,$$

where the constant C_m is the same for all spline wavelet transforms of a given order m, and is given by [24]

$$C_m = \sqrt{\frac{B_{2m}}{(2m)!}}.$$

where B_{2m} is Bernouilli's number of order $2m$. □ In the above theorem, $\|f\|$ defined as follows:

$$\|f\| = \left(\int_0^1 f^2(x)dx\right)^{\frac{1}{2}}.$$

Theorem 4 Assume $K_1, K_2 \in L^2$ in two dimensional rectangle $[0,1]\times[0,1]$ and $g_1, g_2 \in C([0,1]\times[0,1])$. If y and y_j are the exact and approximate solution (obtained by $m-$order B-spline wavelet) of Eq (1), respectively, then

$$\|y(x) - y_j(x)\| \leq B2^{-jm}\|y^{(m)}\|.$$

Proof **:** From Eq 1

$$\begin{aligned}\|y(x)-y_j(x)\| \leq \\ \|\int_0^1 K_1(x,t)\left(g_1(t,y(t)) - g_1(t,P_j y(t))\right)dt\| \\ +\|\int_0^x K_2(x,t)\left(g_2(t,y(t)) - g_2(t,P_j y(t))\right)dt\|,\end{aligned} \tag{34}$$

by Cauchy Schwartz inequality the first part of right hand integral can be written as:

$$\begin{aligned}\|\int_0^1 K_1(x,t)\left(g_1(t,y(t)) - g_1(t,P_j y(t))\right)dt\| \leq \\ \|K_1(x,t)\|\left(\int_0^1 \left(g_1(t,y(t)) - g_1(t,P_j y(t))\right)^2 dt\right)^{\frac{1}{2}},\end{aligned}$$

suppose

$$\left(\int_0^1 \left(K_1(x,t)\right)^2 dt\right)^{\frac{1}{2}} \leq M_1, \tag{35}$$

on the other hand by the mean value theorem we can write:

$$g_1(t,y) - g_1(t,P_j y) \leq A_1|y(t) - P_j y(t)|, \tag{36}$$

where

$$A_1 = sup\{|g_{1_2}(t,s(t))|, 0 \leq t \leq 1\}$$

(g_{1_2} is the derivative of g_1 respect to the second variable) and

$$s(t) = max\{g_1(t,y(t)), g_1(t,P_j y(t)), 0 \leq t \leq 1\}.$$

Also,

$$\begin{aligned}&\int_0^x K_2(x,t)\left(g_2(t,y(t)) - g_2(t,P_j y(t))\right)dt \\ \leq &\int_0^x |K_2(x,t)|.|g_2(t,y(t)) - g_2(t,P_j y(t))|dt \\ \leq &M_2\int_0^x |g_2(t,y(t)) - g_2(t,P_j y(t))|dt,\end{aligned} \tag{37}$$

where $M_2 = sup\{|K_2(x,t)|, 0 \leq x,t \leq 1\}$.

$$|g_2(t,y(t)) - g_2(t,P_j y(t))| \leq A_2|y(t) - P_j y(t)|, \tag{38}$$

with

$$A_2 = sup\{|g_{2_2}(t,s(t))|, 0 \leq t \leq 1\}$$

(g_{2_2} is the derivative of g_2 respect to the second variable) and

$$s(t) = max\{g_2(t,y(t)), g_2(t,P_j y(t)), 0 \leq t \leq 1\}.$$

Substituting Eqs. 33-36 and pervious theorem in Eq. 32 we get:

$$\|y(x) - y_j(x)\| \leq (M_1A_1 + M_2A_2)C2^{-jm}\|y^{(m)}\|.$$

Putting $B = (M_1A_1 + M_2A_2)C$, proof is completed. □

5.1 Illustrative example

In this section, for showing the accuracy and efficiency of the described method we present some examples.

Example 1: Consider the nonlinear two-point boundary value problem

$$u''(t) - e^{u(t)} = 0, \quad 0 \leq t \leq 1, \quad u(0) = u(1) = 0. \tag{39}$$

which evidently is of some interest in magnetohydrodynamics [16, p. 41]. This equation can be reformulated as

$$u(t) = \int_0^1 k(t,s)e^{u(s)}ds,$$

where

$$k(t,s) = \begin{cases} -s(1-t) & s \le t \\ -t(1-s) & s \le t. \end{cases}$$

The exact solution of this integral equation is

$$u(t) = \ln(\frac{c^2}{2}) - \ln(\cos(\frac{c}{2}(t-\frac{1}{2})))^2.$$

where c is the only solution of $\frac{c^2}{2} = \cos(\frac{c}{4})$. Table 1 represents the error estimates using the method of [3] together with the results obtained for maximum errors by the present method.

Table 1: Error vs j_u for example 1.

Methods	$\|y - y_*\|$
Method of [3]	
$N = 17$	5.61×10^{-3}
$N = 33$	1.44×10^{-4}
$N = 65$	3.66×10^{-5}
Present method for $j_0 = 3$	
$j_u = 3$	$< 10^{-5}$
$j_u = 4$	$< 10^{-7}$
$j_u = 5$	$< 10^{-9}$

Example 2: [15] Consider the equation

$$u(x) = \frac{-1}{30}x^6 + \frac{1}{3}x^4 - x^2 + \frac{5}{3}x - \frac{5}{4} + \int_0^x (x-t)(u(t)^2)dt + \int_0^1 (x+t)u(t)dt,$$

with the exact solution $u(x) = x^2 - 2$. Table 2 present exact and approximation solution for $u(x)$, obtained by the method in section 4 at the octave level $j_0 = 3$ and at the levels $j_u = 4, 5$.
RHF: approximation solution by rationalized Haar functions.

Table 2: Exact and approximate solutions of example 2.

	Approximate		RHF	
x	$j_u = 4$	$j_u = 5$	$k = 16$	Exact
0	-2.0000	-2.0004	-1.9999	-2
0.2	-1.9607	-1.9600	-1.9599	-1.96
0.4	-1.8403	-1.8400	-1.8399	-1.84
0.6	-1.6405	-1.6400	-1.6400	-1.64
0.8	-1.3601	-1.3600	-1.3600	-1.36
1	-1.0004	-1.0000	-1.0000	-1

Examples of reference items of different categories shown in the References section include:

6. Conclusions

In this paper, we proposed an advanced numerical model in solving nonlinear Fredholm-Volterra-Hammerstein integral equation of the second kind by means of semi orthogonal compactly supported spline wavelets via collocation method. The approach can be extended to nonlinear integro-differential equation with little additional work. Further research along these lines is under progress and will be reported in due time.

References

[1] N.I. Mushkelishvili, Singular Integral Equations, Noordhoff, Groningen, 1953.
[2] F.G. Tricomi, Integral Equations, Dover, 1982.
[3] R.P. Agarwal, D. O'Regan, P.J.Y. Wong, Positive Solutions of Differential, Difference and Integral Equations. Kluwer Academic: Dordrecht; 1999.
[4] N.H. Ibragimov, R.N. Ibragimov, Invariant solutions as internal singularities of nonlinear differential equations and their use for qualitative analysis of implicit and numerical solutions. Communications in Nonlinear Science and Numerical Simulation. 2009;14:3537-3547.
[5] D. O'Regan, M. Meehan, Existence Theory for Nonlinear Integral and Integrodifferential Equations. Kluwer Academic: Dordrecht; 1998.
[6] J. Banat's, V. Rzepka, On existence and asymptotic stability of solutions of a nonlinear integral equation. J. Math. Anal. Appl. 2003;284:165-173.
[7] J. Weese, A reliable and fast method for the solution of Fredholm integral equations of the first kind based on Tikhonov regularization, Computer Physics Communication, 1992: 69, 99-111.
[8] H. Brunner, Implicitly linear collocation method for nonlinear Volterra equations, J. Appl. Num. Math. 9 (1982) 235-247.
[9] H. Guoqiang, Asymptotic error expansion variation of a collocation method for VolterraŰHammerstein equations, J. Appl. Num. Math. 13 (1993) 357-369.
[10] K. Maleknejad and M. Karami, Using the WPG method for solving integral equations of the second kind, Appl. Math. Comput. 166 (2005), pp. 123-130.
[11] K. Maleknejad and M. Yousefi, Numerical solution of the integral equation of the second kind by using wavelet bases of Hermit cubic splines, Appl. Math. Comput. 183 (2006), pp. 134-141.
[12] K. Maleknejad, T. Lotfi and Y. Rostami, Numerical computational method in solving Fredholm integral equations of the second kind by using Coifman wavelet, Appl. Math. Comput. 186 (2007), pp. 212-218.
[13] K. Maleknejad, K. Nouri, M. Nosrati Sahlan, Convergence of approximate solution of nonlinear Fredholm-Hammerstein integral equations, Commun Nonlinear Sci Num Simul, Vol. 15, Issue 6 (2010) 1432-1443.
[14] K. Maleknejad, M. Nosrati Sahlan, The method of moments for solution of second kind Fredholm integral equations based on B-spline wavelets, International Journal of Computer Mathematics, International Journal of Computer Mathematics, 2010; Vol 87: 1602-1616.
[15] Y. Ordokhani, M. Razzaghi, Solution of nonlinear VolterraŰFredholm-Hammerstein integral equations via a collocation method and rationalized Haar functions, Applied Mathematics Letters 21 (2008) 4-9.
[16] C. Chui, An introduction to wavelets. New york: Academic press, 1992.
[17] I. Daubechies, Ten lectures on wavelets. Philadelpia, PA; SIAM, 1992.
[18] G. Strang, T. Nguyen, Wavelets and filter banks. Cambridge, MA: Wellesley-Cambridge, 1997.
[19] L. Telesca, G. Hloupis, I. Nikolintaga, F. Vallianatos, Temporal patterns in southern Aegean seismicity revealed by the multiresolution wavelet analysis. Commun Nonlinear Sci Num Simul, 12 (2007) 1418-1426.
[20] CK. Chui, Wavelets: a mathematical tool for signal analysis. Philadelpia, PA: SIAM, 1997.
[21] S.G. Mallat, A theory for multiresolution signal decomposition: The wavelet representation , IEEE Trans, Pattern Anal. Mach. Intell. 11 (1989) 674-693.

[22] G. Strang, Wavelets and dilation equations: a brief introduction, SIAM Review, Vol. 31 (1989) 614-627.
[23] G. Strang and G. Fix, A Fourier analysis of the finite element variational method, in: Constructive Aspect of Functional Analysis, Edizioni Cremonese, Rome, 1971, 796-830.
[24] M. Unser, Approximation power of biorthogonal wavelet expansions, IEEE Trans. Signal Processing, Vol. 44, No. 3 (1996) 519-527.

SESSION

CLOUD AND GRID COMPUTING + WORKFLOW + LOAD BALANCING AND SCHEDULING

Chair(s)

Prof. Hamid R. Arabnia

Towards simplifying workflow execution on the Grid

T. Mehlinger, Dan Andresen[1], A. Tygart
Computing & Information Sciences, Kansas State University
234 Nichols Hall, Manhattan, KS 66506-2302
{tcm, dan, kuffs}@k-state.edu

Abstract - *Researchers often spend valuable time performing mundane tasks to prepare for processing data on HPC. This is a function of two problems: first, researchers are forced to use unfamiliar tools to perform tasks on a compute cluster; and second, typical data management practices are not conducive to quickly moving data to a compute cluster for processing. In this paper, we propose a system that solves both these problems. Using VisTrails in conjunction with the Globus Toolkit, researchers can use simple visual tools to submit jobs to a variety of clusters, effectively making scheduling tasks transparent (i.e., the researcher is not required to interact with any particular schedule). We also propose a scheme for storing scientific data that integrates with VisTrails to reduce the data-management effort required of researchers.*

Keywords: VisTrails HPC Python MongoDB Web Services

1 Introduction

The software stack described in this paper aids researchers by providing simple tools to fetch data for processing on HPC clusters. Using VisTrails, a workflow system developed by the University of Utah, and modules we have developed to integrate VisTrails with XML-RPC services and the Globus Toolkit, researchers can migrate their existing processes into a workflow system with simple drag-and-drop semantics. Our tools remove much of the "bookkeeping" associated with managing scientific data, and also alleviate much of the difficult work associated with communicating between the desktop and an HPC installation.

2 Related Work

There exist several other competing approaches for allowing one to do remote execution triggered from a workflow.

CrowdLabs [1] is an extension to VisTrails itself. It offers a web portal for submitting and running jobs via VisTrails running in a so-called server mode. A user would configure a VisTrails client to log in to CrowdLabs and modify workflows and data synced with CrowdLabs. Then, when a user executes a workflow, the entire workflow is pushed to CrowdLabs and executed remotely, after which the results are synced back to the client. This is approach is advantageous in that it offers a cohesive VisTrails experience. Users never have to work outside of the workflow application. However, CrowdLabs limits users to 10MB of data storage and has no options for parallelization. Furthermore, some scientists may be wary of executing bleeding-edge research on hardware they do not control.

The Opal Toolkit [2] looks to solve the more general problem of making cluster applications accessible to everyday users. It offers a web portal for creating and executing jobs derived from a set of scientific applications. It has tight integration with Globus and its related authentication frameworks. It also can generate VisTrails modules for use in users' workflows. The disadvantage is that the web application is deployed on a per-cluster-application or per-set-of-cluster-applications basis. Furthermore, Opal requires the presence of a service dedicated to handling cluster requests, making it a non-ideal ideal solution for a research team without a system administrator.

Taverna [3] is a competing workflow system written in Java. It offers many of the same features as VisTrails but with stronger first-party support for SOAP and REST services. In fact, Taverna's primary use case is consumption and processing of web-hosted data. It offers remote execution through its own server instance. Users can submit and execute workflows on the server instance just as though it were hosted locally, with all the same features. Taverna is more robust than the CrowdLabs add-on for VisTrails in that it does not place restrictions on data file sizes while maintaining the cohesive workflow experience. However, it still requires a separate server installation.

3 Implementation

Our work was driven by a desire to provide an easy-to-use interface to high-performance computing for researchers, and a need for efficient, high-capacity storage for carbon flux data collected from the Konza Prairie Biological Station. We chose the Globus Toolkit [4] for our cluster interface layer. This toolkit gives a full stack of interface libraries and communication protocols for interacting with any member cluster of the Globus Alliance. Features such as authentication, job submission, status checking, and file transfer are all supported. The toolkit also has the advantage of being the de facto standard for inter-cluster communication. For storage, we initially used PostgreSQL to store carbon flux data, ultimately settling on MongoDB for its improved performance and scalability.

Our system provides simpler tools for researchers than they would typically use to submit jobs to HPC clusters and manage their data. Scheduling tasks to run on HPC is a rather specialized task--there are a variety of schedulers available

for HPC facilities to use and they all vary somewhat in implementation and interface. The Globus Toolkit provides a single unified interface to a multitude of schedulers, making it possible for us to develop packages for VisTrails to harness this flexibility and pass the gains in usability on to researchers. Researchers are also often faced with a myriad of ways in which to store data, including (but certainly not limited to) databases, raw text, established scientific data formats, proprietary formats, spreadsheets, etc. Using research on carbon flux data at Kansas State University as a test case, we have developed means by which to flexibly store and retrieve data using MongoDB and web services. All these tools combined form a powerful tool--researchers need only write the code to conduct their research, and our tools are responsible for data management and job submission.

The entire stack is composed of a number of distinct subsystems. Our first task was to develop a way for researchers to easily integrate our tools into their existing processes. We reasoned that a workflow system like VisTrails was the best candidate to that end; researchers could easily take their existing processes and develop VisTrails workflows to accomplish the same tasks. Having selected VisTrails as our workflow tool, we needed to integrate support for Globus-enabled HPC clusters. In order to integrate Globus into a user's workflow, we needed to write Python modules to expose the low-level C libraries. Lawrence Berkeley National Laboratory has written pyGlobus [5] for this purpose; however, though pyGlobus is referenced in the Globus documentation, it is poorly supported and was impossible to configure and build on any of our development platforms. Thus we wrote our own ctypes interface to the Globus toolkit that we call cglobus [6]. It is meant to be a lightly pythonic library, reproducing the C API as closely as possible while exploiting useful features from Python. Our API is similar to such a degree that the Globus C API should suffice as reference documentation.

In Figure B, we have an example of the cglobus library in use. This is a re-implementation of the example job_submit.c code provided with the Globus C API documentation. Our Python version is significantly easier to read and requires fewer lines of code (21 lines in our implementation as opposed to 66 in the reference implementation). We've made the effort to preserve the namespaces used in the C API in order to achieve maximum compatibility with the reference documentation. As such, it should be easy for a developer to translate the C API examples into Python. For convenience, we have used the `require_module` decorator to load Globus modules into memory in this example, but developers could still revert back to the classic `module_load()` and `module_unload()` functions if necessary.

```
#!/usr/bin/env python
import sys
from ctypes import c_char_p, byref

from cglobus.common import requires_module
from cglobus.libc import free
from cglobus import gram_client

@requires_module(gram_client.module)
def main(argv):
    if len(argv) != 3:
        print >> sys.stderr, "Usage: %s RESOURCE-MANAGER-CONTACT RSL" % argv[0]
        sys.exit(1)

    job_contact = c_char_p()
    gram_client.job_request(argv[1], argv[2], 0, None, byref(job_contact))
    print job_contact.value
    free(job_contact)

if __name__ == '__main__':
    main(sys.argv)
```

Fig. A: job_submit.py, an example of the ease of use of cglobus.

After completing this API bridge, we produced a VisTrails package for interacting with Globus and an ancillary XML-RPC service to track state in our interaction with Globus. The package consists of a number of modules for submitting jobs, querying job status, and exchanging data; the first of these modules is called JobSubmit. This module takes various parameters like the cluster URL, program name, invocation arguments, etc. and then builds and sends a job script to the cluster defined in the URL. It returns a job contact URL (a unique job id) that can be used to check the status.

This JobSubmit module relies on a mediating XML-RPC service in order to track job status updates from the cluster. This service functions as a job callback contact that caches job status information from Globus. Globus does not retain job status after a job has completed, so in order to stay informed about the state of a given job, Globus expects clients to implement an HTTP server to serve as a callback listener. This works well in public networks but it is common for clients to be NAT-ed behind a firewall, making it impossible for callback requests to reach the client without configuring the firewall to forward the traffic (a task well beyond the responsibility of most developers or researchers). Our XML-RPC server acts as a proxy for these callbacks and will store the result of the job. We can then simply query the proxy for job status updates.

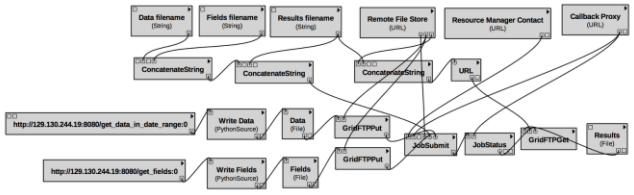


Fig. B: VisTrails workflow connecting data providers with Globus.

The second module in our VisTrails package, JobStatus, queries this XML-RPC service and throws an exception (which VisTrails will catch and handle) if a job is incomplete or returned an error code, preventing further execution in the workflow. This, combined with the caching features of VisTrails allows us to "pause" the execution of the workflow until the job is finished running on the cluster. When the job is completed, JobStatus will return the status codes for the job instead of throwing an exception, allowing execution to

proceed. The remaining modules in our VisTrails package are intended to sync files to and from a cluster, named GridFTPPut and GridFTPGet. These modules are still under development, providing minimal functionality for testing.

In a typical use case (Figure C), a user would fetch data from web-enabled data providers for preprocessing and aggregation. The user would then write this data to file and upload it to the cluster for more involved computation. Next, the user would submit a job to operate on that data using JobSubmit. The user would then use the JobStatus module to "pause" the workflow. Finally, the user would copy the result data from the cluster, optionally using VisTrails to do facilitate further computation. Using our tools, a typical desktop user is able to interact with a cluster without being forced to write shell scripts or use unfamiliar tools, and with minimal disruption of their existing workflows. Note that for working with very large datasets, the data would ideally be stored in the same facility with the cluster and the user would not use VisTrails as a conduit for uploading data. Rather, the job submitted by the user would simply operate on data already existing at the target cluster (illustrated by the dashed line between "Compute Cluster" and "Datastore" in Fig. C).

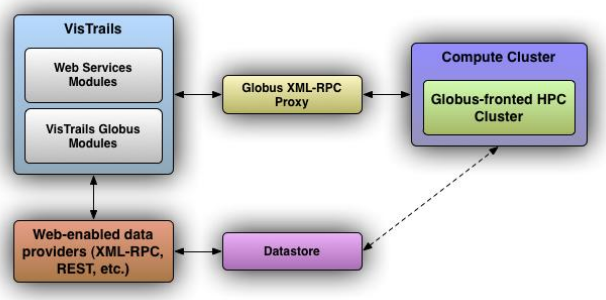


Fig. C: relationships among data provider, VisTrails, Globus Proxy, and Globus-enabled cluster.

Having developed a workflow system and cluster interface, we also required means to efficiently store the carbon flux data for later manipulation. The carbon flux data are tabular, high-frequency time-series data, streamed in raw format to storage at Kansas State University for preparation. The raw, unprepared data are stored in text files; the fully-curated data are stored in a proprietary binary format, also in files. Both sets of data are hosted on a Windows file share behind a NAT. This storage scheme has a number of primary disadvantages:

- Researchers cannot easily select arbitrary ranges of data to investigate. For a small range of data, a researcher must select a small portion from one particular file. For a large range of data, the researcher must aggregate data from many files.
- Researchers cannot easily share data. Anyone who desires on-demand access to the data must also be behind the NAT. Anyone outside the NAT relies on someone inside the NAT to provide the data. This disadvantage compounds the difficulties already described, and in particular, sharing large volumes of data is very cumbersome.
- Searching the data is potentially intensely time-consuming and computationally wasteful; e.g., searching for outliers in a particular range of values could require a full scan of the data.

To resolve these problems, we reasoned that the best solution was comprised of using database-backed storage for persistence and web services for data access. This approach conferred many advantages over shared files:

- Selecting an arbitrary range of data is easy.
- Sharing the data, safely, outside a NAT is much more feasible, particularly for large data sets.
- Searching the data is more efficient.
- Sharing data with web services makes the data available to more researchers and more tools.

We designed a storage scheme composed of two components: the data store itself (initially a single PostgreSQL instance, then a single MongoDB instance, ultimately a MongoDB cluster), and a data provider (an application server exposing an XML-RPC to fetch data).

We first settled on using PostgreSQL for housing the data, extracting the raw data from files using tools written in Python. The data would then be served from the database using an XML-RPC service, also written in Python. We initially considered other communication protocols (such as SOAP), settling on XML-RPC for its simplicity, allowing us to rapidly develop a proof-of-concept. We then developed two packages for VisTrails: one that would communicate with an XML-RPC service, discover the methods it exposed, and populate VisTrails with modules to handle fetching data; and a second that handled communication and data transfers with Globus.

We had mixed success with the initial configuration. Although the relational features of PostgreSQL were unnecessary for our use case, it seemed at first that the tabular nature of the data was well suited to storage in fixed columns (this later developed into a disadvantage). However, insert performance in PostgreSQL led us to seek an alternate solution. While the insert rate for new records was quite reliable (it remained constant, independent of the growth of the stored data), it was not high enough to satisfy our needs.

We abandoned PostgreSQL in favor of MongoDB. MongoDB confers a number of advantages over PostgreSQL in our particular use case:

- MongoDB is not relational. The absence of constraints inherent in a relational database confers an immediate performance benefit.
- MongoDB is schema-free, i.e., collections (analogous to tables in relational databases) are

merely a table of records with arbitrary number of named field/value pairs, and values are not strictly typed. This allows us much more flexibility to correct flawed data because we can store it in its uncorrected state alongside valid data.
- MongoDB has built-in map/reduce functionality, allowing us to offload many processing tasks into the database (e.g., computing averages or finding outliers).
- MongoDB scales more effectively. PostgreSQL scales to balance load but individual query performance does not benefit (and can in fact suffer in some cases).
- Configuring for scalability and redundancy in MongoDB is much simpler than doing so in PostgreSQL.

Given all the advantages, MongoDB was an obvious choice to succeed PostgreSQL as our storage tool. In addition to the advantages listed above, MongoDB has one feature in particular that would make later development much more convenient: configuring off-site replication is simple, allowing us to mirror data remotely. By encouraging collaborating teams at other universities to host MongoDB, we not only have data security, we also have a means to easily share large volumes of data for collaborative purposes.

To make actual data available for scientific research, we have developed simple XML-RPC services to expose the data to VisTrails, which communicates via our custom XML-RPC module. A user wishing to retrieve data from our service simply configures the module with the URL of the service, which then exposes all the available methods for fetching data. The VisTrails XML-RPC module inspects the signature of all available methods exposed by the XML-RPC service and constructs additional modules for each of those methods. The user can then simply drag a module that returns the data they desire into a workflow; the XML-RPC calls will run and fetch data when the workflow is executed. Notably, users need not commit to using VisTrails just to fetch data using our services--any tool capable of communicating via XML-RPC can pull data from our services.

4 Evaluation

Integrating the Globus Toolkit within VisTrails enables easy access to the member HPC facilities of the Globus Alliance. This enables us to submit jobs to any of a number of clusters without regard for the scheduler used at any particular facility but obviously adds some overhead to the job submission process; i.e., instead of submitting jobs directly to a job queue, Globus acts as a broker between the client submitting a job and the scheduler, inherently adding some time to the job submission process. This adds some overhead in transit and processing time for Globus operations:

- Submitting a job: 3.834s
- Checking job status: 0.005s

Submitting a job is indeed quite slow, while checking the job status is rather quick. However, note that both operations involve sending commands to our callback proxy rather than interacting directly with Globus. The proxy takes some time to process a submit request and a because we have no way to send a notification back to VisTrails that a job is finished, a user could accumulate a vast number of job status requests that total some appreciable amount of time. Were we able to interact directly with Globus, submission would certainly be quicker, although we would still encounter the problem with accepting a callback for a finished job. However, in practice, the overhead is negligible; given the scale at which it becomes practical to run code on HPC, turnaround time from job submission to finished results is dominated by queue wait and computation time. Were Globus-attributed latency to actually become a significant portion of the total turnaround time for a particular task, the task would simply be considered unsuitable for running on HPC--even a massively parallel or memory-intensive task run in seconds on HPC would conceivably only take a matter of minutes on a modest workstation. Tasks operating on this time scale are simply inappropriate for systems typically used for hours-, days-, or weeks-long computation.

Where we do find significant time consumed is during data transfer. The dataset we used to test our VisTrails modules, XML-RPC services, and grid proxy consisted of 60,000 rows (approximately 5.5MB) of carbon flux data. Using our test workflow shown in Fig. B, we transferred the data from the data provider to the VisTrails host, then stored the data in plain text and pushed it to the cluster datastore using GridFTP. Fetching the data into VisTrails and pushing to the cluster took 34.67s and 38.46s, respectively, giving us nominal data transfer rates of 162kB/s and 146kB/s, respectively. However the actual transfer rate over XML-RPC must have been much faster owing to XML-RPC overhead, while the figure for transfer over GridFTP is a realistic reflection of the real transfer rate (GridFTP is essentially FTP, which has low overhead). We found these figures to be somewhat disappointing, although isolating the bottleneck was both infeasible (we tested on a cluster over which we had no direct control) and irrelevant; transferring large volumes of data through VisTrails to a cluster at the transfer rates we experience would be quite impractical. In practice, a job submitted using our tools would expect to find data already hosted at the HPC facility (one way to do this would be by using MongoDB's off-site replication feature described previously); VisTrails would then be used to pull metadata or smaller sets of more volatile supporting data from XML-RPC data providers to then push to the cluster. We are currently collaborating with the Universities of Kansas and Oklahoma on a system that supports these semantics inherently.

The systems powering both the PostgreSQL database and the single-instance MongoDB database and the MongoDB cluster are configured as follows:

- PostgreSQL and single-instance MongoDB:
  - 2x dual-core AMD Opteron 270
  - 8 GB DDR2 memory
  - 1.5 TB from 2x 750G SAS, hardware RAID 0, 256 KB stripe size
- MongoDB cluster (four machines, per-node configuration shown)
  - 2x quad-core AMD Opteron 2376
  - 8 GB DDR2 memory
  - 60 GB SAS (single disk)

As stated earlier, we had only moderate success with PostgreSQL, owing the drawbacks to poor insert performance. To evaluate PostgreSQL, we recorded time-to-insert for varying-size data sets as reported by the time command. We have also calculated and included the time the insert process spent blocked. Because performance scaled linearly with increasing record set size, we have shown figures for 1 million and 8 million records.

Records	Real	User + Sys	Blocked	% Blocked
1M	26m15.749s	20m40.980s	5m34.769s	21.2%
8M	3h29m15.344s	2h44m8.080s	45m7.264s	21.6%

Fig. D: Time to execute insertions on PostgreSQL.

Although the insert rate is significantly faster than the real-time data collection (a 2 gigabyte record set, approximately 18 million records, is 11 days of data), time-to-insert is unrealistically long for the volume of data still residing in text files. Insert time for six months of data would be approximately 3.5 days. Of particular note is the time the spent blocked; for each set of data, time blocked was approximately 21% of the total execution time. For a six-month data set, this would result in 17 hours of compute time wasted.

Because we were dissatisfied with PostgreSQL's performance, we opted to use MongoDB. We first tested it in a single-instance configuration, noting significant performance gains. We also tested in a four-way sharded configuration. Shown are figures for inserting 8 million records:

	Real	User + Sys	Blocked	% Blocked
Single	1h3m8.520s	1h2m53.640s	14.881s	0.4%
4x Sharded	43m32.400s	42m2.399s	1m30.000s	3.4%

Fig. E: Single-instance and four-way sharded MongoDB performance.

There is a clear performance advantage over PostgreSQL, with MongoDB achieving 3.3 times the insert performance in single-instance configuration, and 4.8 times the insert performance in sharded configuration. For a full six-months of carbon flux data, this translates to a total insert time of just 17.5 hours on clustered MongoDB.

MongoDB also performs much better on select (properly, "find" in MongoDB jargon) than PostgreSQL. To test select performance, we issued a query that would return 1 million records, stored the returned records in memory, and recorded the time to perform the entire operation. Performance was as follows:

- PostgreSQL: 2m24.53s
- MongoDB (single): 55.92s
- MongoDB (sharded): 58.62s

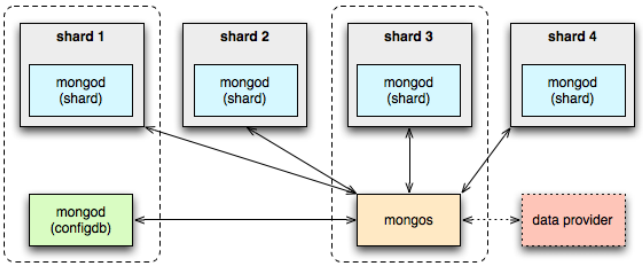


Fig. F: Communication among MongoDB services. Dashed boxes indicate services sharing a single host.

At first, we were surprised that sharded read performance lagged behind single-instance read performance. However, this was easily explained by MongoDB's documentation; chunks of data are read in parallel from their respective shards and must be put assembled via merge sort by the mongos service. Although individual reads are fast, the merge sort has some overhead. Using replication in addition to sharding would further improve read performance (in addition to adding data security), but for testing purposes, we were satisfied with these results.

5 Conclusions

In this paper, we have presented a system for simplifying interaction with HPC and managing scientific data. We have described a number of ways in which we have integrated disparate tools in order to bring greater power and flexibility to researchers by relieving them of many of the tasks typically associated with HPC. We have also evaluate the performance of MongoDB as a scientific datastore, demonstrating its excellent performance characteristics and suitability for hosting data at a large scale. Finally, we have described web services to make data seamlessly available not only in VisTrails, but to any tools that can communicate over XML-RPC.

Our effort is not the first of its kind. Of note, DataOne [7, 8] is a collaborative effort of universities, government organizations, and research institutions whose goal is to facilitate the sharing of scientific data and tools at a global scale. DataOne members provide infrastructure for storage and data sharing, and development effort to provide tools to member institutions and the scientific community at-large. Unfortunately, DataOne is not currently in a state that provides useful functionality for our use cases.

Acknowledgements

This work has been supported in part by NSF grants 1006860 and 0919443.

6 References

[1] E. Santos, P. Mates, J. Freire and C. Silva, "CrowdLabs: Social Analysis and Visualization for the Sciences," To appear in SSDBM 2011

[2] S. Krishnan, B. Stearn, K. Bhatia, K. Baldridge, W. Li, P. Arzberger, "Opal: SimpleWeb Services Wrappers for Scientific Applications," pp.823-832, IEEE International Conference on Web Services (ICWS'06), 2006

[3] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Greenwood, T. Carver, M. Pocock,, A. Wipat and P. Li, "Taverna: A tool for the composition and enactment of bioinformatics workflows," pp.3045, Computer Applications in the Biosciences, 2004

[4] I. Foster, "Globus Toolkit Version 4: Software for Service-Oriented," pp.2-13, Lecture Notes in Computer Science, 2005

[5] K. R. Jackson, "pyGlobus: a Python interface to the Globus Toolkit". Concurrency and Computation: Practice and Experience, 14: 1075–1083. doi: 10.1002/cpe.683, 2002

[6] CyberCommons cglobus project: https://bitbucket.org/cybercommons/cglobus

[7] W. Michener, D. Vieglais, T. Vision, J. Kunze, P. Cruse, G. Janeé, "DataONE: Data Observation Network for Earth — Preserving Data and Enabling Innovation in the Biological and Environmental Sciences," D-Lib Magazine, 2011

The Improvement of Dynamic Load Balancing Strategy in Grid Computing

Amr Rekaby[1], **Prof. Mohamed Abu Rizkaa**[1]
[1]Arab Academy for Science, Technology and Maritime Transport
College of Computing & Information Technology, Cairo - Egypt
rekaby0@hotmail.com, m_rizka@aast.edu

Abstract - *Load balancing is one of the major search areas in grid computing. The strategy of the grid computing is another important topic. Job scheduling is the way of scheduling jobs in a grid environment that contains heterogeneous resources.*

The strategy of the grid computing describes the overall model of the grid hierarchy. Some papers show a tree model for a grid computing. This paper enhances a model for grid environment structure and provides a new algorithm towards improving the dynamic job scheduling activity in grid environment. First, introduction to job scheduling models and taxonomies will be discussed. Then a survey of tree grid model is described, after that the enhanced model will be shown, proposed strategy and algorithm steps will be presented, then an analytical comparison is presented to show what this new strategy could improve.

Keywords: Grid computing, dynamic load balancing strategy, dynamic jobs scheduling.

1 Introduction

By definition, a grid computing is an environment that contains a lot of different resources capabilities and specification connected to each other via internet and/or local area network. This kind of heterogeneous of resources adds a big challenge on the job scheduling. The main target of job scheduling is firstly assign a task to a proper resource according to the tasks needs and resource qualifications. Secondly balance the workload between the grid resources, to avoid having an over utilized resource while have others resources under-utilized and could be used to help the over utilized resource. This work load and job scheduling algorithm is highly dependent on the grid structure and the overall strategy.

Theoretically there are two kind of job scheduling algorithms:

- Static scheduling, selection of suitable resource is done in compilation time, so there is logic needed in run time to assign the task to the proper resource. This means that any run time conditions will not be considered in the job scheduling algorithm.
- Dynamic scheduling, selection of the suitable resource is done in run time, per actual task request, so the updated status will be considered in this kind of scheduling.

The strategy of the grid is the context that the job scheduling algorithm will be applied. Some papers discussed a monitoring techniques for job status monitoring, others talks in details in the tasks workflow structure. The new strategy in this paper will touch such staff, but the main target is describing the overall grid strategy.

This research is discussing already exist grid strategy (tree structure), then apply some modification on it. Proposing a multi-level cluster grid environment structure (unlimited clusters levels), describe the grid elements roles and responsibilities, after that presents a new algorithm for dynamic job scheduling. At the end of the paper, an analysis of the new strategy will be presented against already existing strategies.

2 Load Balancing Models Survey

Researches in the Grid strategy discuss some models like "Star" and "Tree" model. The "star" model (like in Fig 1) is connecting all the grid resources together under supervision by grid manager, which store the information of the resources, manages the tasks propagation from the task sender to the task receiver and so on.

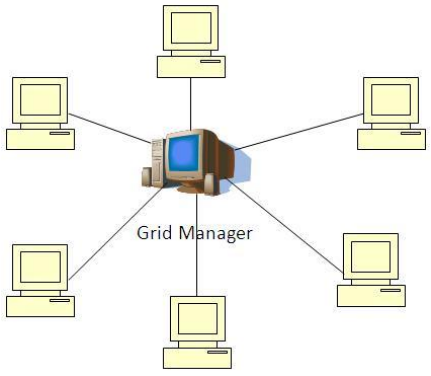


Fig 1: Star Grid Model

A 4-layer tree model [3] is grouping the grid resources into "sites", then encapsulates many sites under "cluster", then manages all clusters via grid manager. The 4-levels here are:

- Grid manager level.
- Clusters level.
- Sites level.
- Actual resources level (attached to a site).

The scheduling algorithm inside the sites is called "intra-site" algorithm. Others inside the clusters called "intra-cluster" and the global grid algorithms are called "intra-grid" algorithms.

These 3 levels of algorithms are focusing on the load balancing from job migration perspective. When a site, cluster or a grid reach an unbalanced state which means, some resources in this scope are over utilized and others are under-utilized, a migration of some tasks should be done towards balancing this site, cluster or grid. But the criteria that are used for resource selection in intra site scope are not considered in this research.

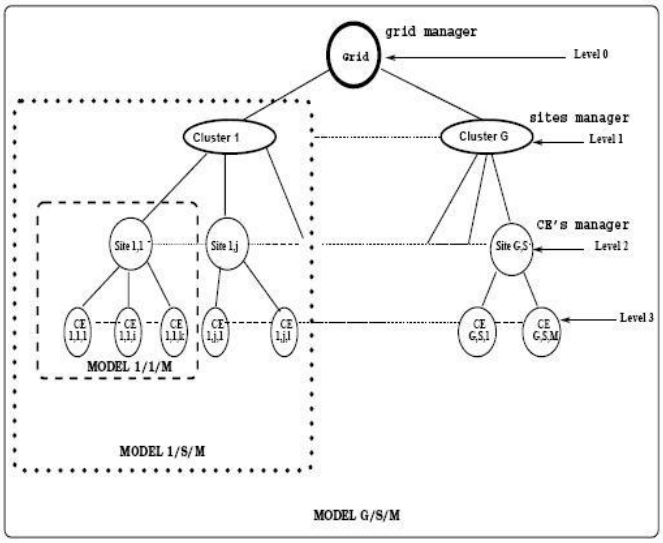


Fig 2: Star Grid Model [3]

Almost all researches which care about this point, study the intra-site level only, how the resource in the site will be selected for a specific task. But the migration between site and another or between cluster and another is done without criteria. If a task needs mandated specifications and there is no resource in its sender site that have these specifications, so the task should be migrated into another site/cluster. Or if the site is overloaded and a new task is submitted, this task also should be migrated. The decision of which target site/cluster should be selected is almost a random decision. The task is migrated from its owner site to the nearest one, then start search for a suitable resource in this new site, if could find, so it is matched with a resource that will run it, if could not find, a migration to a new site/cluster should be done.

This try and error way consume time till find a suitable resource. This paper presents a new algorithm in this point to safe the needed time.

The migration from site to another site, or another cluster also depends on the grid model, if it is tree model like described above, "1 cluster/M sites" model or "M clusters/ M sites" model. So for increasing the usefulness of the new algorithm, it is coupled with a new grid topology (N-Levels Tree Model).

3 Proposed grid model and algorithm

This paper proposes two ways of enhancements. The first way is development of 4-levels tree grid model. The second one is presenting a new algorithm for job scheduling in intra-cluster scope.

Regarding the grid model (topology), this paper works on changing in the tree model described earlier as described in Fig 3. The new model is N-Levels tree model which means that, the grid will be constructed from many computation elements (CE) grouped in a site, others CE are grouped in another site (like described tree model). One or more sites are grouped in a cluster, after that the levels of cluster hierarchy is not fixed to specific count. One or more clusters/sites could be grouped in higher level cluster which could also exist under a higher cluster and so on. So there is no level for "grid management", because the highest level in the grid is also a normal cluster level and works with same algorithm.

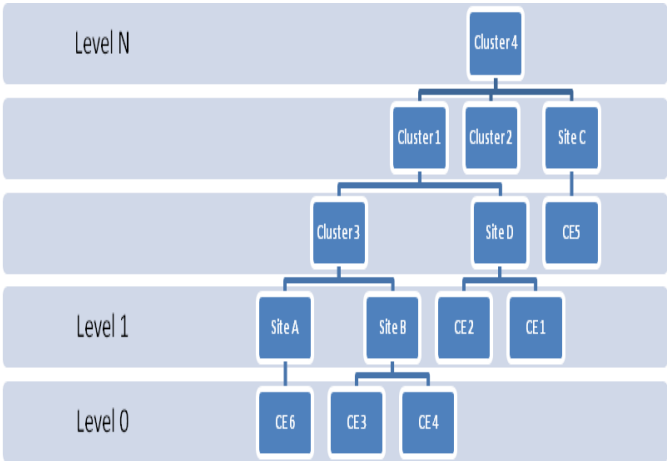


Fig 3: N-Levels grid tree model

These modifications in the tree model provide:

- More hierarchal structure is provided, which will decrease the communication time that needed when migrating tasks from site to another.

- More logical grouping, a site is a directed connected resources via local area network, but a cluster is a logical group. For example shown in fig 3, a task submitted from "CE 3" will run in "Site B" if this is applicable, else it will try to run under "Cluster 3" umbrella, so "Site A" has a priority here than "Site D" or "Site C". This point will add more control to grid administrator in structuring the grid clusters as administrator prefer. These administrations' preferences could consider the resources specifications in its grouping.

These modifications means that, there is not "Grid Manager" algorithm needed, the "Cluster Manager" algorithm would fulfill the cluster and grid management needs.

This paper is interested in intra-cluster algorithm, so no new enhancements will be presented in intra-site job scheduling algorithm. In the simulation section, a linear search algorithm is used for the intra-site search scope as a default solution.

In intra-cluster scope, the algorithm proposes metadata for each site/cluster to be tracked. These metadata fields are:

- Platform (all distinct operating systems values exist in this site/cluster).
- CPU utilization (average of all site/cluster resources utilization).
- Memory utilization (average of all site/cluster resources utilization).
- Fault history (the average fault history of resources).

Tasks are sent to the grid environment in a directed acyclic graph (DAG). Each node in this DAG is presenting a task with its dependencies. Each task has its needs from specification point of view. Some of these needs are mandatory to be considered to be able to find a suitable resource such as operating system needed for this task.
Others depend on the grid environment setup, which could be considered or not according to the grid algorithm and intelligent, such as fault history.
CPU utilization and memory utilization are two factors that are considered in our proposed algorithm. The submitted task could request a specific maximum utilization of the hosting resource.
Regarding the fault history, it is an optional task need, some tasks request a minimum level of mature resources and others doesn't care about it.
Each site saves a failure history for each resource (how many failed tasks against the total executed tasks). The site/cluster fault history is an average of this site/cluster resources history.

Cluster manager job scheduling algorithm will be executed when a task migrated from its own site into higher cluster manager. Job scheduling algorithm is shown in Fig 4.

```
Begin
    task received from lower level
    get cluster metadata information
    for each site/cluster under this cluster
        for each platform in the metadata platform list
            if match task needs
                platform_match=true
            end if
        end loop
        if platform_match=false
            go to next site/cluster loop
        end if
        check others metadata (CPU, memory and fault)
        if match with task needs
            Target_site= this loop site
            Break loop over site/cluster
        End if
        If not match
            Continue with next site/cluster loop
        End if
    End loop
    If Target_site <> null
        Migrate task to Target_site
    Else
        Search is failed under this cluster
        If there is higher cluster level exist
            Propagate task to higher level
        Else
            Scheduling in the grid is failed
        End if
    End if
End
```

Fig 4: cluster job scheduling search algorithm

This algorithm will avoid the time consuming in sending the task to another site/cluster, then in intra-site scope find that, this site doesn't have resources with such needs like the needed platform. So holding distinct values of site platform will be maintained in the upper cluster to shortcut the search algorithm. The average CPU, memory utilization and fault history of the site/cluster is not 100% accurately reflect the resources information, because these are related to all the site/cluster resources. So for example, it may present that this site is utilized 70% while the actual resource that the task match is only utilized 20%. But these criteria give the cluster manager a general feedback on lower site/cluster which could help in neglecting some over utilized or bad failure history site/cluster.
Each resource has its own tracking monitor module, which track the status of this resource such as CPU, memory utilization and fault history periodically, then send these updates to the site/cluster manager. The site/cluster manager will have the responsibilities of collecting the resources information and construct the overall site/cluster status depends on the average resources updates.
Information such as the distinct platforms exist in the site/cluster is maintained during the functionality of joining a

new resource to the grid, or disconnecting a resource out of the grid.

4 Experiments and Results

A simulation experiments were done to find the improvement effect of this new model and algorithm. The communication time is neglected in these experiments, the results are mentioning the consumed time of job scheduling search algorithm, starting from search inside the site, then search inside the cluster if needed, till find the suitable resource in the grid.

As shown in Fig 5 the experiments of new algorithm are executed on environment of 20 sites, in each site there are 200 CE. The specifications of these CEs are randomly distributed. These sites are grouped into 6 clusters over 5 levels tree. Against experiments on 4-levels tree model with the same 20 sites, and the same 200 CEs per each site then grouped into 6 clusters over 4 levels tree (4-levels tree described above). Different tasks are submitted in each setup parallel in each site. 25% of the tasks submitted could not be run in the owner site, so they have to be migrated into another site/cluster.

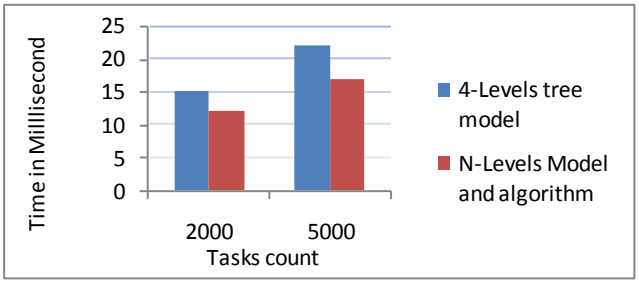


Fig 5: performance analysis with different tasks count

The experiments show that, the search algorithm is much faster in the N-Levels grid tree topology than 4-Levels tree model. If the ratio of migrated tasks (25% in the above results) is increase, this will show more different between N-Levels and 4-Levels tree model, and also between the proposed search algorithm and the randomly propagating technique which always used on the intra-cluster scope.

5 Conclusion

Job scheduling and Load balancing problem in grid environment depends on the overall grid model. In intra-cluster scope, it is hard to have detailed resources information, but saving information as much as possible and use it in the navigation of migrated tasks could help in the performance of the job scheduling search algorithm.

The proposed N-Level tree model is generalizing the concept of cluster from just a group of sites to a group of sites/clusters and then a whole grid environment. This would help in a logical grouping of resources against network bandwidth, resources capabilities and so on, which make the grid environment more manageable.

6 References

[1] R. Al-Khannak, B. Bitzer, "Load Balancing for Distributed and Integrated Power Systems using Grid Computing",

[2] Cui Zhendong, Wang Xicheng,"A Grid Scheduling Algorithm Based on Resources Monitoring and Load Adjusting"

[3] Belabbas Yagoubi, Yahya Slimani, "Dynamic Load Balancing Strategy for Grid Computing", PROCEEDINGS OF WORLD ACADEMY OF SCIENCE, ENGINEERING AND TECHNOLOGY VOLUME 13 MAY 2006 ISSN 1307-6884

[4] Babar Nazir, Taimoor Khan,"Fault Tolerant Job Scheduling in Computational Grid", IEEE--ICET 2006 2nd International Conference on Emerging Technologies Peshawar, Pakistan, 13-14 November 2006

[5] Igor Sfiligoi, "Making Science in the Grid World:Using Glideins to Maximize Scientific Output", 2007 IEEE Nuclear Science Symposium Conference Record

[6] Yu Liang, Zhou Jiliu, "The Improvement of A Task Scheduling Algorithm in Grid Computing", First International Symposium on Data, Privacy and E-Commerce

[7] Ozan, Nezih, Saeid, Alexandru, Dick "Performance Analysis of Dynamic Workflow scheduling in Multicluster Grids"

[8] Making Science in the Grid World: Using Glideins to Maximize Scientific Output

Profile for Effective Service Management on Mobile Cloud Computing

Changbok Jang, Hyokyung Chang, Hyosik Ahn, Yongho Kang, Euiin Choi*

Dept. Of Computer Engineering, Hannam University, Daejeon, Korea
{chbjang, hkjang, hsahn}@dblab.hannam.ac.kr, kang@r2soft.co.kr, eichoi@hnu.kr

Abstract. Mobile Cloud Computing has become as a new IT paradigm because of the growth of mobile device like smartphone and appearance of Cloud Computing environment. This mobile cloud environment provides various services and IT resources according to users' requests, so an effective management of service and IT resources is required. Hence, this paper designs a profile on mobile cloud service platform in order to provide distributed IT resources and services to users based on context-awareness information. As the profile proposed in this paper uses context-aware information, it enables to provide more accurate personalized services and manage distributed IT resources.

Keywords: Cloud computing, Context-aware, Profile, Intelligence Service, Mobile cloud computing

1 Introduction

The market of mobile recently has been evolving rapidly and cloud computing is spreading into mobile as well. That is why mobile cloud computing is becoming a new issue today. Cloud computing is the computing that provides virtualized IT resources as a service by using Internet technology. In cloud computing, a user lends IT resources (software, storage, server, network) as needed, uses them, get a support of real-time scalability according to service load, and pays as he/she goes. Especially the cloud computing environment distributes IT resources and allocates according to user's request, so there should be a study on technology that manages these resources and deals with effectively[1].

Mobile cloud computing creates a new chance for IT industry because it allows the superiority and economic of cloud computing to meet the mobility and convenience of mobile and draws a synergy effect for both. Also mobile cloud computing refers to an infrastructure that data storage and data processing is done outside mobile device by using cloud computing in the regardless of kinds of mobile devices. Mobile devices used in the mobile environment include personal information and enable to provide the environment that collects a variety of context-aware information. Users' demand on service types suitable for the individual situation has been increasing.

* Corresponding Author

So, context-aware reasoning technique has been studied to provide a suitable service for user by using user' context and personal profile information in mobile environment[2-9]. In this context-aware system, a formal context model has to be provided to offer information needed by application as well as store context and manage. However, there are some technical constraints for this context-aware model to overcome because it itself cannot be applied to mobile platform due to limited device resources, so the study on intelligent mobile service in mobile platform is still insufficient. Recent interest related to mobile cloud is personal smartphone. The study on physical support like connecting smartphone to personal virtual system on cloud and using computing resources unlimitedly is quite active, but the study on how to manage distributed IT resources effectively and provide intelligent mobile service through reasoning based on collected information and role as a medium of collecting context of mobile device is ignored.

Therefore, this paper designs a profile based context-aware knowledge on mobile cloud service platform and develops in order to the optimized mobile cloud service through recognizing the conditions of user and cloud server and reasoning on the basis of external context achieved from mobile or internal user's personal information and information of resources from cloud server and service use information.

2 Related Works

Mobile platform is mainly referred to mobile middleware what lets users operate the optimized contents or service on mobile and it provided a formed interface to UI and service by using RTOS (Realtime OS) and hardware function. There are Windows Mobile, iPhone, Android, Symbian, etc. as these mobile platform.

The object of awareness in this mobile cloud computing which differ from previous computing environment is not human, but device[10]. This means that information of awareness is to be personalized. Because inference power of computer was not enough to understand human's action and thought. But because they actively have to provide various service and information to aware human's behavior and thought, there is need of definition, how expressed human's behavior and thought to context information. Also, because human's behavior and thought was individually differ, how understand personal inclination is important problem. For this problem solving on mobile cloud computing, it was inferred context which was able to understand human's behavior based situation information which collected from various sensors. And they are provided service and information to user through the inferred context. Generally, situation which arose around human was able to collect from sensor, but personal inclination and thought was not. Therefore, they used the method of personal information storage for analyzing inclination, such as personal profile, history, diary[11,12]. As mentioned above, user of mobile cloud computing was provided various services without human's recognization by mobile devices in anywhere and anytime. So, we have to infer context for provide the services to users correctly. Therefore, there are studying about technique of context inference to use personal inclination and information. As it demanded personal inclination and information in

context inference, using the user's profile for storing it, research about technique of profile was as follow:

UbiData project was suggested data process and synchronization and addresses these three challenges using an architecture and sophisticated hoarding, synchronization, and transcoding algorithms to enable continuous availability of data regardless of user mobility and disconnection, and regardless of the mobile device and its data viewing/processing applications[13,14]. Annika Hinze describe TIP (Tourism Information Provider) system, which delivers various types of information to mobile devices based on location, time, profile of end users, and their "history", i.e., their accumulated knowledge. The system hinges on a hierarchical semantic geospatial model as well as on an Event Notification System (ENS)[15]. Annie Chen proposed a context-aware collaborative filtering system that can predict user preferences in different context situations, based on past user-experiences. The system uses what other like-minded users have done in similar context, to predict a user's preference towards an item in the current context[16]. Manuele Kirsch-Pinheiro proposed a context-based filtering process, aimed at adapting awareness information delivered to mobile users by collaborative web systems. This filtering process relied on a model of context which integrates both physical and organizational dimensions, and allows representation of the user's current context as well as general profiles. These profiles are descriptions of potential user contexts and express awareness information filtering rules to apply when the user's current context matches one of these rules. Given a context, these rules reflect user preferences. They describe how the filtering process performs in two steps, the first for identifying the general profiles that apply, and the second for selecting awareness information[17].

However, these profile techniques not sufficient on mobile cloud computing. Therefore, this paper proposes profile in order to manage resources more effectively by using personal context information and do modeling context-aware information in mobile platform and reason.

3 System Architecture

In this paper, we suggested context-aware-based intelligence mobile cloud service platform for efficiently managing resource to use context-aware information.

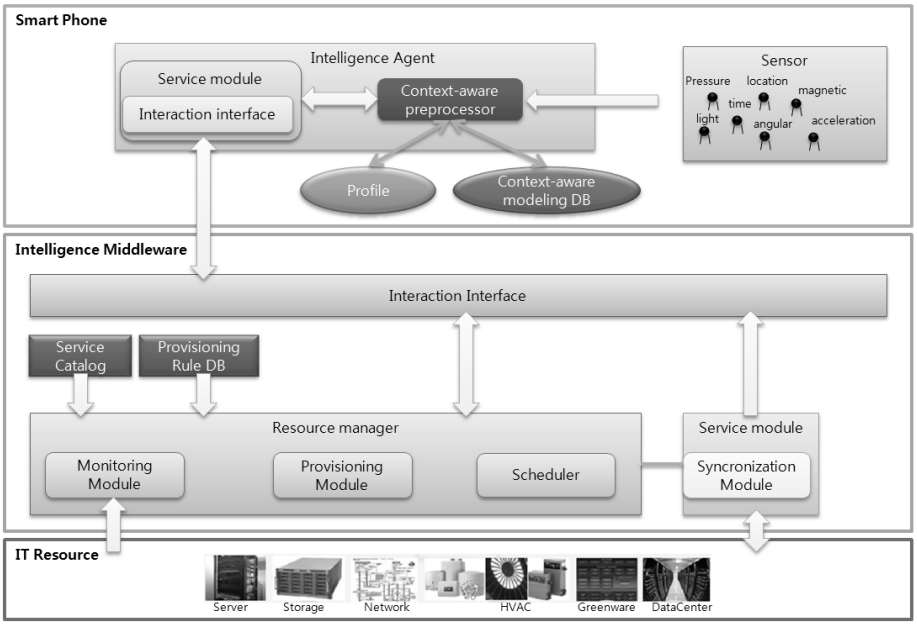


Fig. 1. Suggested platform architecture

As shown figure 1, suggested system consisted of intelligence agent and intelligence middleware. Intelligence agent was responsible for understanding a variety of context-aware information and inferring it. And it consisted of sub-modules such as service module, context-aware preprocessor, personal profile, context-aware information modeling database. Intelligence middleware was responsible for providing services and efficiently managing IT resources by user's request on mobile cloud computing. Context-aware preprocessor on intelligence agent included process for collecting context-aware information and modeling it, inferring context-aware information, and responsible for understanding what user's situation was. Service module was responsible for sending context-aware information to intelligence middleware, providing services that suitable to user. Personal profile was repository which was stored personal information, such as service information by using user, user's ID, password. Context-aware modeling database was stored to information which was modeled by using ontology. Intelligence middleware consisted of interaction interface for communicating to agent, resource manager, service manager, Service catalog, Provisioning Rule Database. Resource manager responsible for effectively allocating and managing service information was required for processing user's request service, and consisted of monitoring module, provisioning module, and scheduler. Monitoring module crawled information of IT resource utilization. Provisioning module set up plan for providing best service to analyze context-aware information which was transferred by user and utilization information of IT resource. Scheduler was scheduled to utilization of service and resource by plan which was established to provisioning module. Service Catalog was stored service information for which user used, provisioning rule database was stored rule for providing best provisioning process to use context-aware information and utilization of resource.

Also, service module was responsible for executing service and using distributed IT resource to providing service to user, and consisted of sub-module, such that synchronization module. Synchronization module responsible for synchronizing resource which user was using on cloud computing.

4 Profile Structure

5.2 Definition of User Profiles

A user's profile specifies information of interest for an end user. So in this paper, the profile was consisted of user information part and service information part. User information part stored user's information such as user's name, inclination, hobby and Service information part stored services that they were used such as service name, service provider etc.

Structure of user profile was follow:

- User Information: User name, User ID, Personal inclination, hobby, etc
- Service Information: Service Name, Service Provider, Service context, Service frequency value, etc

Because profile stored how much the service information used, stored not only used service, but also information when, how, where used. Also, there are stored the information about what context used.

DTD of profile which we were suggested follow.

```
<?xml version='1.0' encoding='UTF-8' ?>
<!ELEMENT ups_profiles (upsp)>
<!ELEMENT upsp(user_info, service_list?, device_info)>
<!ELEMENT user_info (username , password, hobby)>
<!ATTLIST user_info userID ID #REQUIRED >
<!ELEMENT username (firstname , lastname)>
<!ELEMENT firstname (#PCDATA)>
<!ELEMENT lastname (#PCDATA)>
<!ELEMENT password (#PCDATA)>
<!ELEMENT hobby (#PCDATA)>
<!ELEMENT service_list (service*)>
<!ELEMENT service (service_name, service_provider, service_time, service_frequency*) >
<!ELEMENT service_name(#PCDATA)>
<!ELEMENT service_provider(#PCDATA)>
<!ELEMENT service_time(#PCDATA)>
<!ELEMENT service_frequency(week_info, access_time, location)>
```

<!ATTLIST service_frequency value CDATA #REQUIRED >
<!ELEMENT week_info (#PCDATA)>
<!ELEMENT access_time (#PCDATA)>
<!ELEMENT location (#PCDATA)>

5.2 Profile Manipulation

We assumed that the services will use this place and time next time, if service was demanded in specific location and time. So, we used the information of time, location, frequency to provide services to user more correctly and suggested profile which using recently access time, access time, frequency of access, location value, weekend value. And the values stored in profile.

- recently access time(t): This value stored time when service used recently, and use for finding service which not used for a long time.
- access time(a): This value have to 24 from 0, and if service was used on 1 P.M, it's value has 13.
- frequency of access (f): This value stored frequency of service how many user used the service.
- location value(l): This value have unique number of place where service was used. For example, if user used A service in house and office, location value of A service which used in house is 1, other is 10.
- weekend value(e): This value have to 7 from 1, if service used on Monday, weekend value is 1. Generally, people's life pattern was repeated per week. So we use the value for analyzing service frequency of user per week.

In case of using the service, we need location information where service used for inferring user's inclination and context efficiently. So, we simply classified location information which has a unique value, such as the following:

- Home : Bathroom(1), bedroom(2)
- Office : Lobby(3), elevator(4), floor(5), office room(6), conference room(7)
- Other : Street(8), Car(9), etc

And we represented frequency of access to 3-Dimension graph which have three-coordinate values(access time, location value, weekend value). For example, if user demand A service at 7 A.M, Monday on bathroom, an then weekend value(monday is mean 1 in the location model) has 1, access time has 9, location value has 1. So frequency of A service is represent at coordinate (7, 1, 1) and has 1 value. If user will demand A service at same time and place, frequency of A service which has coordinate (7, 1, 1) will become 1 by increasing. Also, we find location information of service which has most high value of frequency which place on responding coordinate among it, and put it in service storage which user will use the service.

5 Conclusion and Future Work

In this paper, to provide users with suitable services and manage resources effectively by using context information in the mobile cloud environment, we were proposed to profile which stored location and time, frequency information of often used service among various services, and put the service which will expect to use in any location. The system used a profile made in the form of an XML document, and classified information, which was used by users when context arose with elements such as location, time, date(week), and frequency. If the user is located at a specific place, our system provide service to the user through location, time, date(week), and frequency information, which is stored in the user's profile.

As a further research, we are needed algorithm of similarity assessment between current and past context, and technique that extracts context information and does modeling, the resources management technique that manages distributed IT resources effectively by using context information, and the part that examines the performance and tests after embodying the actual platform proposed.

Acknowledgments. This research was financially supported by the Ministry of Education, Science Technology (MEST) and National Research Foundation of Korea(NRF) through the Human Resource Training Project for Regional Innovation

References

1. AMIT GOYAL and SARA DADIZADEH : A Survey on Cloud Computing, In: University of British Columbia Technical Report for CS 508(2009).
2. Hess, CK., Campbell, RH. : An application of a context-aware file system, In: Pervasive Ubiquitous Computing, Vol. 7, No. 6(2003).
3. Khungar, Sh., Riekki, J. : A Context Based Storage for Ubiquitous Computing Applications, In: Proceedings of the 2nd European Union symposium on Ambient intelligence, pp. 55-58(2004).
4. Mayrhofer, R. : An Architecture for Context Prediction, In: Trauner Verlag, Schriften der Johannes-Kepler-Universität Linz, Vol. C45(2005).
5. Byun H.E., K. Cheverst : Exploiting User Models and Context-Awareness to Support Personal Daily Activities, In: Workshop in UM2001 on User Modelling for Context- Aware Applications(2001).
6. Byun, H.E., Cheverst, K. : Utilising context history to support proactive adaptation, In: Journal of Applied Artificial Intelligence, Vol. 18, No 6, pp. 513-532(2004).
7. G. M. Sur, J. Hammer : Management of user profile information in UbiData, In: University of Florida Technical Report TR03-001(2003).
8. Biegel, G., Vahill, V.: A Framework for Developing Mobile, Context-aware Applications. In: IEEE International Conference on Pervasive Computing and Communications (PerCom)(2004).
9. Gu, T., Pung, H.K., Zhang, D.Q.: A Middleware for Building Context-Aware Mobile Services. In: Proceedings of IEEE Vehicular Technology Conference (VTC) (2004).
10. M. Weiser. : Hot topics-ubiquitous computing, In: IEEE Computer, Vol. 26, No. 10(1993).
11. Barkhuus, L., Dey, A. :Is Context-Aware Computing Taking Control away from the User Three Levels of Interactivity Examined. In: Proceedings of the 5th International Conference on Ubiquitous Computing (UbiComp'03)(2003).

12. Elfeky, M.G., Aref, W.G., Elmagarmid, A.K. : Using Convolution to Mine Obscure Periodic Patterns in One Pass, In: Proceedings of the 9th International Conference on Extending Database Technology (EDBT)(2004).
13. G. M. Sur, J. Hammer, : Management of user profile information in UbiData, In: Technical Report TR03-001, Dept. of CISE, University of Florida, Gainesville(2003).
14. J. Zhang, A. S. Helal, J. Hammer. : Ubidata: Ubiquitous mobile file service, In: Eighteenth ACM Symposium on Applied Computing(2003)
15. A. Hinze, A. Voisard. : Location- and time-based information delivery in tourism, In: Advances in Spatial and Temporal Database(2003).
16. Annie Chen : Context-Aware Collaborative Filtering System: Predicting the User's Preference in the Ubiquitous Computing Environment, In : LoCA 2005, Volume 3479(2005).
17. Manuele.Kirsch-Pinheiro, Marlene Villanova- Oliver, Jerome Gensel, Herve Martin, : Context-Aware Filtering for Collaborative Web Systems: Adapting the Awareness Information to the User's Context, In : ACM Symposium on Applied Computing(2005).

Context Model Based on Ontology in Mobile Cloud Computing

Changbok Jang, Euiin Choi*

Dept. Of Computer Engineering, Hannam University, Daejeon, Korea
chbjang@dblab.hannam.ac.kr, eichoi@hnu.kr

Abstract. Mobile Cloud Computing has become as a new IT paradigm because of the growth of mobile device like smartphone and appearance of Cloud Computing environment. This mobile cloud environment provides various services and IT resources according to users' requests, so an effective management of service and IT resources is required. Hence, this paper designs a context model based on ontology in mobile cloud computing in order to provide distributed IT resources and services to users based on context-awareness information. As the context model proposed in this paper uses context-aware information, it enables to provide more accurate personalized services and manage distributed IT resources.

Keywords: Cloud computing, Context-aware, Context model, Ontology, Intelligence Service, Mobile cloud computing

1 Introduction

The market of mobile recently has been evolving rapidly and cloud computing is spreading into mobile as well. That is why mobile cloud computing is becoming a new issue today. Cloud computing is the computing that provides virtualized IT resources as a service by using Internet technology. In cloud computing, a user lends IT resources (software, storage, server, network) as needed, uses them, get a support of real-time scalability according to service load, and pays as he/she goes. Especially the cloud computing environment distributes IT resources and allocates according to user's request, so there should be a study on technology that manages these resources and deals with effectively[1].

Mobile cloud computing creates a new chance for IT industry because it allows the superiority and economic of cloud computing to meet the mobility and convenience of mobile and draws a synergy effect for both. Also mobile cloud computing refers to an infrastructure that data storage and data processing is done outside mobile device by using cloud computing in the regardless of kinds of mobile devices. Mobile devices used in the mobile environment include personal information and enable to provide the environment that collects a variety of context-aware information. Users' demand on service types suitable for the individual situation has been increasing.

* Corresponding Author

Therefore, context-aware reasoning technique has been studied to provide a suitable service for user by using user' context and personal profile information in mobile environment[2-9]. In this context-aware system, a formal context model has to be provided to offer information needed by application as well as store context and manage. However, there are some technical constraints for this context-aware model to overcome because it itself cannot be applied to mobile platform due to limited device resources, so the study on intelligent mobile service in mobile platform is still insufficient. Recent interest related to mobile cloud is personal smartphone. The study on physical support like connecting smartphone to personal virtual system on cloud and using computing resources unlimitedly is quite active, but the study on how to manage distributed IT resources effectively and provide intelligent mobile service through reasoning based on collected information and role as a medium of collecting context of mobile device is ignored.

Therefore, this paper designs a context model based on ontology in mobile cloud computing and develops in order to the optimized mobile cloud service through recognizing the conditions of user and cloud server and reasoning on the basis of external context achieved from mobile or internal user's personal information and information of resources from cloud server and service use information.

2 Related Works

Mobile platform is mainly referred to mobile middleware what lets users operate the optimized contents or service on mobile and it provided a formed interface to UI and service by using RTOS (Realtime OS) and hardware function. There are Windows Mobile, iPhone, Android, Symbian, etc. as these mobile platform.

There are Context-aware information modeling techniques such as Key-value model, Markup scheme model, Graphical model, Object oriented model, and ontology based model which are used in the existing ubiquitous environment and Web environment. Ontology model, a Context-aware model which has been studied mostly recently, enables to express concepts and interactions easily. Recently ontology model has been studied lively related to Semantic Web study based on OWL(Web Ontology Language) and there is a movement to adapt ontology-based model in a variety of context-aware framework. One of the early methods of context modeling using ontology was proposed by Otzturk and Aamodt. Van Heijst divided ontologies into Structure Type and Concept Issues in the study for ontology. Structure Type is classified as Knowledge Modeling Ontology, Information Ontology and Terminological Ontology. Concept Issues is divided as Domain Ontology, Application Ontology, Representation Ontology and Generic Ontology[10]. Guarino classified ontologies according to general level to represent context of different kinds [11]. Top-level Ontologies describe general concepts like space, time, matter, object, event and action. Domain Ontologies and Task Ontologies describe the vocabulary related to a generic domain or a generic task or activity by specializing the terms introduced in the Top-level Ontology. Application Ontologies describe concepts depending both on a particular domain and task, which are often specializations of both the related ontology. These concepts correspond to roles played by domain

entities while performing a certain activity. Context modeling in context-awareness needs to acquire context initially. Then it is necessary to process modeling to enable acquired context to use. Many projects have used context model with their certain type. Context Toolkit[12] suggested middleware layers that serve to convey to application after acquiring original information and transforming it into any type that application can be understandable. Hydrogen was developed by Hofer[13]. This system is based on hierarchical architecture. This model's representation ability is admirable because it represented model with object-oriented method. But the representation formality is incomplete. Karen's context information model is based on object-oriented method. This modeling concept provides a formal basis for representing and reasoning about some of the properties of context information such as its persistence and other temporal characteristics, its quality and its interdependencies[14]. He attempted to model using both the Entity-Relationship model and the class diagrams of UML. CASS(Context-Awareness Sub-Structure)[15] is a framework for context-aware mobile application designed with middleware approach. By separating into application and context inference, this middleware can be able to infer context without recompiling. CONON(the Context Ontology)[16] is divided as Upper Domain and specific Sub Domain. The context model is structured around a set of abstract entities, each describing a physical or conceptual object including Person, Activity, Computational Entity and Location, as well as a set of abstract sub-classes. This model supports extensibility to add specific concepts in different application domain. It also supports the use of logic reasoning to check the consistency of context information, and to reason over low-level, but it's difficult to represent diverse context with upper context restricted selectively.

However, these context models not sufficient on mobile cloud computing. Therefore, this paper proposes context model in order to manage resources more effectively by using personal context information and do modeling context-aware information in mobile platform and reason.

3 System Architecture

In this paper, we suggested context-aware-based intelligence mobile cloud service platform for efficiently managing resource to use context-aware information.

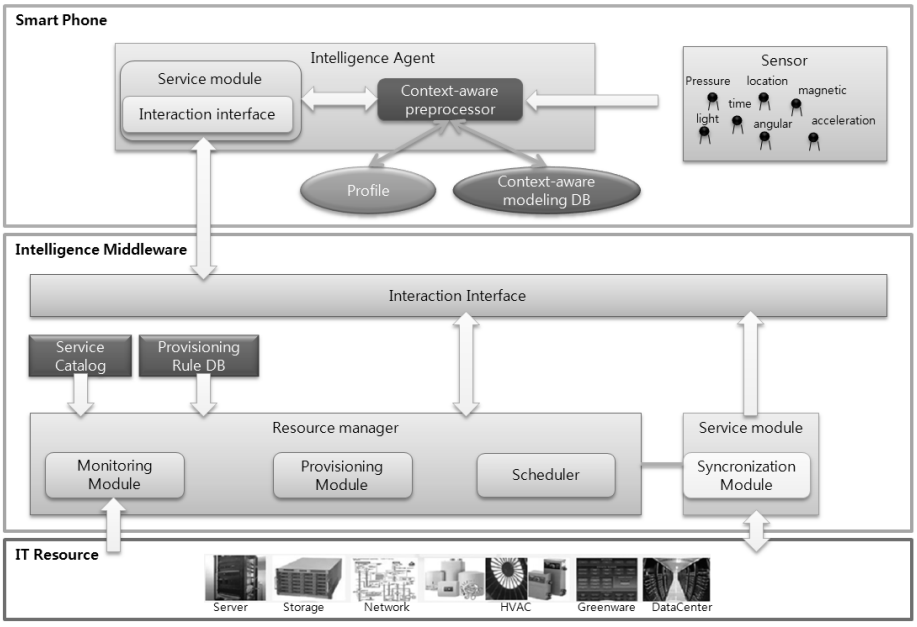


Fig. 1. Suggested platform architecture

As shown figure 1, suggested system consisted of intelligence agent and intelligence middleware. Intelligence agent was responsible for understanding a variety of context-aware information and inferring it. And it consisted of sub-modules such as service module, context-aware preprocessor, personal profile, context-aware information modeling database. Intelligence middleware was responsible for providing services and efficiently managing IT resources by user's request on mobile cloud computing. Context-aware preprocessor on intelligence agent included process for collecting context-aware information and modeling it, inferring context-aware information, and responsible for understanding what user's situation was. Service module was responsible for sending context-aware information to intelligence middleware, providing services that suitable to user. Personal profile was repository which was stored personal information, such as service information by using user, user's ID, password. Context-aware modeling database was stored to information which was modeled by using ontology.

Intelligence middleware consisted of interaction interface for communicating to agent, resource manager, service manager, Service catalog, Provisioning Rule Database. Resource manager responsible for effectively allocating and managing service information was required for processing user's request service, and consisted of monitoring module, provisioning module, and scheduler. Monitoring module crawled information of IT resource utilization.

Provisioning module set up plan for providing best service to analyze context-aware information which was transferred by user and utilization information of IT resource. Scheduler was scheduled to utilization of service and resource by plan which was established to provisioning module.

Service Catalog was stored service information for which user used, provisioning rule database was stored rule for providing best provisioning process to use context-aware information and utilization of resource. Also, service module was responsible for executing service and using distributed IT resource to providing service to user, and consisted of sub-module, such that synchronization module. Synchronization module responsible for synchronizing resource which user was using on cloud computing.

4 Classification of Context Model

In mobile cloud computing, context-aware information which can be used is user's profile, services that user was used, resources for providing services. And we need to provisioning techniques in order to manage resources more effectively on mobile cloud computing, multimodal techniques for supporting convenient user's interface, inferring user's intention more accurately. So, we include entity such as provision, activity. Figure 2 shows each entity and relational property.

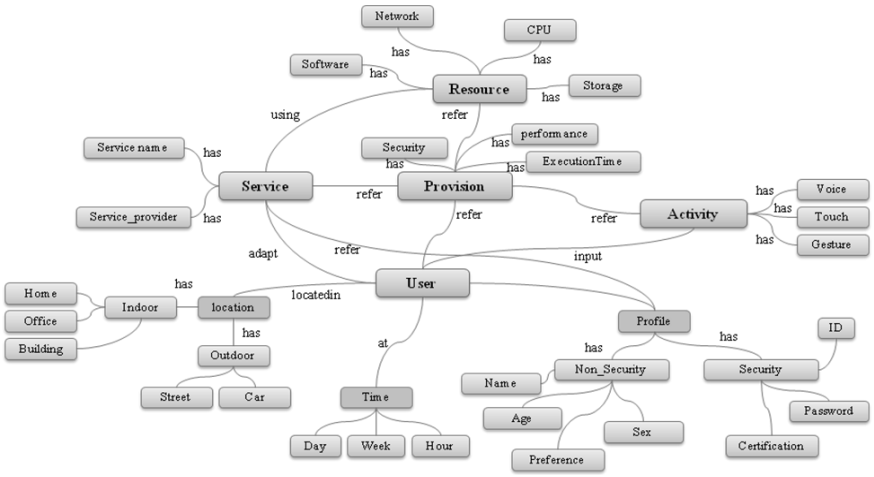


Fig. 2. Context model on mobile cloud computing

In this paper, generic ontology is user, service, resource, provision, activity. And they are connected with each other through relational property(eg. Locatedin between User and location). Individual generic ontology includes domain ontology as a detailed material and immaterial entity(eg, User and location). Consequently, it provides extensibility and formal representation ability by hierarchical ontology classification.

5 Conclusion and Future Work

Context modeling in context-awareness needs to acquire context initially. And then it is necessary to process modeling to enable acquired context to use. In this paper, we have proposed context model to provide users with suitable services and manage resources effectively by using context information in the Mobile Cloud environment. We have also defined context for modeling through diverse context definitions. We have classified ontology and represent hierarchically. The proposed context model by the paper is expected to help have the optimized personalized service and effective IT resources management in the Mobile Cloud environment.

As a further research, we will include additional function for inference. Also we will try to progress in a study that interpret and inference the high level context, and study the resources management technique that manages distributed IT resources effectively by using context information, and the part that examines the performance and tests after embodying the actual platform proposed.

Acknowledgments. This research was financially supported by the Ministry of Education, Science Technology (MEST) and National Research Foundation of Korea(NRF) through the Human Resource Training Project for Regional Innovation

References

1. AMIT GOYAL and SARA DADIZADEH : A Survey on Cloud Computing, In: University of British Columbia Technical Report for CS 508(2009).
2. Hess, CK., Campbell, RH. : An application of a context-aware file system, In: Pervasive Ubiquitous Computing, Vol. 7, No. 6(2003).
3. Khungar, Sh., Riekki, J. : A Context Based Storage for Ubiquitous Computing Applications, In: Proceedings of the 2nd European Union symposium on Ambient intelligence, pp. 55-58(2004).
4. Mayrhofer, R. : An Architecture for Context Prediction, In: Trauner Verlag, Schriften der Johannes-Kepler-Universität Linz, Vol. C45(2005).
5. Byun H.E., K. Cheverst : Exploiting User Models and Context-Awareness to Support Personal Daily Activities, In: Workshop in UM2001 on User Modelling for Context- Aware Applications(2001).
6. Byun, H.E., Cheverst, K. : Utilising context history to support proactive adaptation, In: Journal of Applied Artificial Intelligence, Vol. 18, No 6, pp. 513-532(2004).
7. G. M. Sur, J. Hammer : Management of user profile information in UbiData, In: University of Florida Technical Report TR03-001(2003).
8. Biegel, G., Vahill, V.: A Framework for Developing Mobile, Context-aware Applications. In: IEEE International Conference on Pervasive Computing and Communications (PerCom)(2004)
9. Gu, T., Pung, H.K., Zhang, D.Q.: A Middleware for Building Context-Aware Mobile Services. In: Proceedings of IEEE Vehicular Technology Conference (VTC) (2004).
10. N.Guarino. : Formal Ontology in Information Systems. In : Proceedings of FOIS'98, Trento, Italy, 6-8 June(1998).
11. B.N. Schilit, N. Adams and R. Want. : Context-Aware Computing Applications. In : IEEE Workshop on Mobile Computing Systems and Applications, 8-9 December(1994).

12. H. Wu, M. Siegel and S. Ablay. : Sensor Fusion for Context Understanding. In : IEEE Instrumentation and Measurement Technology Conference, AK, USA, 21-23 May(2002).
13. T. Hofer, W. Schwingwe, W. Pichler, G. Leonhartsberger and J. Altmann. : Context-awareness on mobile devices - the hydrogen approach. In : Proceedings of the 36th Annual Hawaii International Conference on System Sciences, pp.292-301(2002).
14. S. Fahy and S. Clarke. : A middleware for mobile context-aware applications. In : Workshop on Context Awareness, MobiSys(2004).
15. B. Schilit, M. Theimer. : Disseminating Active Map Information to Mobile Hosts. In : IEEE Network,8(5), pp.22-32(1994)
16. Xiaohang Wang, Daqing Zhang, et al. : Ontology-Based Context Modeling and Reasoning using OWL. In : Workshop on Context Modeling and Reasoning at IEEE International Conference on Pervasive Computing and Communication (PerCom'04), Orlando, Florida, March 14(2004).

Budget Constrained Scheduling of Grid Workflows Using Partial Critical Paths

Saeid Abrishami, and Mahmoud Naghibzadeh
Computer Engineering Department, Ferdowsi University of Mashhad, Mashhad, Iran

Abstract— *Workflow scheduling on utility Grids is a complex multi-objective optimization problem which tries to optimize several scheduling criteria such as execution time and cost. One common approach to the problem is to optimize only one criterion, while other criteria are constrained within fixed thresholds. In this paper, we propose a new scheduling algorithm called Budget Partial Critical Path (Budget-PCP) which aims to create a schedule that minimizes the total execution time of a workflow, such that the total execution cost does not exceed the user specified budget. The Budget-PCP algorithm has two phases: in the budget distribution phase, it recursively assigns sub-budgets to the tasks on the partial critical paths ending at previously assigned tasks, and in the planning phase it assigns the fastest service to each task while meeting its sub-budget. The simulation results show that the performance of our algorithm is very promising.*

Keywords: Grid computing, workflow scheduling, utility Grids, economic Grids, QoS-based scheduling

1. Introduction

Workflows are one of the most popular application models on heterogeneous distributed systems like Grids [1]. The problem of scheduling workflows on the Grid environment is a complex optimization problem which deals with the allocation of workflow tasks to the appropriate resources, in order to minimize the total execution time. The scheduling problem becomes more challenging in *utility Grids* which is an open market of distributed services, sold at different prices, with different performance and Quality of Service (QoS) [2]. In this environment, consumers negotiate with service providers on the required QoS and the price to reach a Service Level Agreement (SLA) [3]. There are many potential QoS attributes such as execution time, reliability, security, availability, and so on. Obviously, higher QoS attributes mean higher prices for the services. Due to the presence of multiple QoS attributes, the scheduling problem in utility Grids becomes a multi-objective problem which tries to simultaneously optimize more than one scheduling criterion, e.g., execution time and cost.

There are several methods to deal with the multi-objective scheduling problem [4]. One popular method is to assign a weight to each scheduling criterion and produce a schedule which optimizes the weighted sum of them [5][6]. This approach is used in the *Amadeus* workflow management system [7]. Another common method is to optimize only one criterion, while other criteria are constrained within fixed thresholds [8]. In our previous work [9], we proposed a new QoS-based workflow scheduling algorithm, called the *Partial Critical Paths* (PCP), which tries to minimize the execution cost of the workflow, while meeting the user defined deadline. In this paper, we propose Budget-PCP, which aims to create a schedule that minimizes the total execution time of a workflow, such that the total execution cost does not exceed the user specified budget. First, the Budget-PCP algorithm tries to estimate the minimum makespan which can be achieved using the user specified budget, then it tries to assign a sub-budget to each task and edge of the workflow according to the minimum makespan using a recursive procedure, and finally it schedules the workflow tasks based on their assigned sub-budgets.

The remainder of the paper is organized as follows. Section 2 describes our system model, including the application model, the utility grid model, and the objective function. The Budget-PCP scheduling algorithm is explained in Section 3. A performance evaluation is presented in Section 4, and Section 6 concludes.

2. Scheduling System Model

The proposed scheduling system model consists of an application model, a utility Grid model, and a performance criterion for scheduling. An application is modeled by a directed acyclic graph $G(T, E)$, where T is a set of n tasks $\{t_1, t_2, ..., t_n\}$, and E is a set of arcs. Each arc $e_{i,j} = (t_i, t_j)$ represents a precedence constraint which indicates that task t_i should complete executing before task t_j can start. In a given task graph, a task without any parent is called an *entry task*, and a task without any child is called an *exit task*. As our algorithm requires a single entry and a single exit task, we always add two dummy tasks t_{entry} and t_{exit} to the beginning and the end of the workflow, respectively. These dummy tasks have zero execution time and they are connected with zero-weight arcs to the actual entry and exit tasks, respectively.

A utility Grid model consists of several Grid Service Providers (GSPs), each of which provides some services to the users. Each workflow task t_i can be processed by m_i services $S_i = \{s_{i,1}, s_{i,2}, ..., s_{i,m_i}\}$ from different service

providers with different execution times and different costs. The cost of a service depends on its execution time, i.e., shorter execution times are more expensive. We assume $ET(t_i, s)$ and $EC(t_i, s)$ to be the estimated execution time and execution cost for processing task t_i on service s, respectively. Besides, $TT(e_{i,j}, r, s)$ and $TC(e_{i,j}, r, s)$ are defined as the estimated transfer time and transfer cost of sending the required data along $e_{i,j}$ from service r (processing task t_i) to service s (processing task t_j), respectively. To obtain the available services and their information, the scheduler should query a Grid Information Service (GIS). In a utility Grid, the GIS is used to provide information such as the type, the provider, and the QoS parameters (including price) for all services. Whenever a scheduler accepts a workflow, it contacts the GIS to query about available services for each task and their QoS attributes. Then the broker directly contacts the service's GSP to gather detailed information about the dynamic status of the service, especially the available time slots for processing tasks. Using this information, the scheduler can execute a scheduling algorithm to map each task of a workflow to one of the available services. According to the generated schedule, the broker contacts GSPs to make advanced reservations of the selected services, which results in an SLA between the broker and the GSP.

The last element in our model is the performance criterion, which is to minimize the execution time (makespan) of the workflow subject to the condition that the total execution cost of the workflow is less than or equal to the user specified budget.

3. Budget Partial Critical Paths Algorithm

The Budget Partial Critical Path (Budget-PCP) algorithm has two main phases: Budget Distribution and Planning. In the first phase, the overall budget of the workflow is distributed over the workflow tasks and edges. In the second phase, the planner selects the fastest service for each task which completes that task and all of its incoming edges within their total sub-budget.

Our main contribution in this paper is the budget distribution algorithm which is based on a Critical Path (CP) heuristic. Critical path heuristics are widely used in workflow scheduling. The *critical path* of a workflow is the longest execution path between the entry and the exit tasks of the workflow. Our budget distribution algorithm uses the average execution time and the average execution cost of the critical path to estimate the minimum makespan of the workflow which can be achieved using the user specified budget. Then, the algorithm assigns a sub-budget to each task and edge of the critical path according to the estimated makespan. Now, the algorithm can carry out the same procedure by considering each critical node in turn as an exit node, and creating a *partial critical path* that ends in the critical node and that leads back to an already assigned node, i.e., a node which has already been given a sub-budget. In the Budget-PCP algorithm, this procedure continues recursively until all tasks are successfully assigned a sub-budget. Finally, the planning algorithm schedules the tasks according to their sub-budgets. In the following sections, we elaborate on the details of the Budget-PCP algorithm.

3.1 Basic Definitions

In the Budget-PCP scheduling algorithm, we have two notions of the start times of tasks, the earliest start time computed before scheduling the workflow, and the actual start time computed by the planning algorithm. For each unscheduled task t_i we define its Earliest Start Time, $EST(t_i)$, as the earliest time at which t_i can start its computation, regardless of the actual service that will process the task (which will be determined during planning). Clearly, it is not possible to compute the exact $EST(t_i)$, because a Grid is a heterogeneous environment and the computation time of tasks varies from service to service. Furthermore, the data transmission time is also dependent on the selected services and the bandwidth between their providers. Thus, the minimum execution and data transmission times are used as an approximation for each unscheduled task. The Minimum Execution Time, $MET(t_i)$, and the Minimum Transmission Time, $MTT(e_{i,j})$, are defined as follows:

$$MET(t_i) = \min_{s \in S_i} ET(t_i, s) \tag{1}$$

$$MTT(e_{i,j}) = \min_{r \in S_i, s \in S_j} TT(e_{i,j}, r, s) \tag{2}$$

Having these definitions, we can compute $EST(t_i)$ as follows:

$$\begin{aligned} EST(t_{entry}) &= 0 \\ EST(t_i) &= \max_{t_p \in t'_i s\, parents} EST(t_p) + MET(t_p) + MTT(e_{p,i}) \end{aligned} \tag{3}$$

For each scheduled task we define the Selected Service $SS(t_i)$ as the service selected for processing t_i during scheduling, and the Actual Start Time $AST(t_i)$ as the actual start time of t_i on that service. These attributes will be determined during planning.

3.2 Workflow Scheduling Algorithm

Algorithm 1 shows the pseudo-code of the overall Budget-PCP algorithm for scheduling a workflow. This algorithm receives a workflow graph, $G(T, E)$, and a budget, B, and tries to schedule the workflow within the budget. First, two dummy nodes t_{entry} and t_{exit} are added to the task graph, even if the task graph already has only one entry or exit node. After that, the METs and the MTTs are computed and using them, the EST of each task is computed in lines 4 - 6. Then, the algorithm tries to estimate the minimum makespan of the workflow which can be achieved using the user's budget. For this reason, first the algorithm finds the overall critical

Algorithm 1 The Budget-PCP Scheduling Algorithm

```
1: procedure SCHEDULEWORKFLOW(G(T, E), B)
2:     request available services for each task in G from GIS
3:     add t_entry, t_exit and their corresponding edges to G
4:     compute MET(t_i) for each task according to Eq. 1
5:     compute MTT(e_{i,j}) for each edge according to Eq. 2
6:     compute EST(t_i) for each task in G according to Eq. 3
7:     compute EM according to Eq. 5
8:     EST(t_exit) ← EM
9:     mark t_entry and t_exit as assigned
10:    call AssignParents(t_exit)
11:    call AdjustBudgets(G(T, E), B)
12:    call Planning(G(T, E))
13: end procedure
```

Algorithm 2 Assigning Deadline to the Parents Algorithm

```
1: procedure ASSIGNPARENTS(t)
2:     while (t has an unassigned parent) do
3:         PCP ← null, t_i ← t
4:         while (there exists an unassigned parent of t_i) do
5:             add CriticalParent(t_i) to the beginning of PCP
6:             t_i ← CriticalParent(t_i)
7:         end while
8:         call AssignPath(PCP)
9:         for all (t_i ∈ PCP) do
10:            update EST for all successors of t_i
11:            call AssignParents(t_i)
12:        end for
13:    end while
14: end procedure
```

path (CP) of the algorithm using the previously computed ESTs. Then, it computes the critical path sub-budget, CPB, as follows:

$$CPB = \frac{\sum_{t_i \in CP} AEC(t_i) + \sum_{e_{i,j} \in CP} ATC(e_{i,j})}{\sum_{t_i \in T} AEC(t_i) + \sum_{e_{i,j} \in E} ATC(e_{i,j})} \times B \quad (4)$$

where the Average Execution Cost of a task t_i, $AEC\,(t_i)$, and the Average Transfer Cost of an edge $e_{i,j}$, $ATC(e_{i,j})$, are defined as follows:

$$AEC(t_i) = \frac{\sum_{s \in S_i} EC(t_i, s)}{m_i}$$

$$ATC(e_{i,j}) = \frac{\sum_{r \in S_i,\, s \in S_j} TC(e_{i,j}, r, s)}{m_i \times m_j}$$

Eq. 4 computes the critical path sub-budget according to the average computation and transmission cost of the critical path compared to the computation and transmission cost of the whole workflow, and the user's budget. Having the critical path sub-budget, we can compute the estimated makespan of the workflow, EM, as follows:

$$EM = k \times \frac{\sum_{t_i \in CP} AEC(t_i) + \sum_{e_{i,j} \in CP} ATC(e_{i,j})}{CPB} \times \left(\sum_{t_i \in CP} AET(t_i) + \sum_{e_{i,j} \in CP} ATT(e_{i,j}) \right) \quad (5)$$

where the Average Execution Time of a task t_i, $AET\,(t_i)$, and the Average Transfer Time of an edge $e_{i,j}$, $ATT(e_{i,j})$, are defined in a similar way to $AEC\,(t_i)$ and $ATC(e_{i,j})$, respectively. The coefficient $0 < k \leqslant 1$ determines the factor by which we want to decrease the makespan, by assigning more budget to the critical tasks and edges on the critical path. This coefficient should be big enough such that the estimated makespan can be achieved by the user specified budget. We set k=0.8 in our experiments.

Then, the EST of t_{exit} has been set to EM (line 8), which enforces the parents of t_{exit}, i.e., the actual exit nodes of the workflow, to be finished before the estimated makespan. After that, the dummy nodes are marked as assigned (line 9). An *assigned* node/edge is defined as a node/edge which has already a sub-budget, and clearly a node/edge without a sub-budget is called *unassigned*. The most important part of the algorithm is the last three lines. In line 10 the procedure *AssignParents* is called for t_{exit}. This procedure assigns sub-budgets to all unassigned parents and their corresponding edges of its input node. As it has been called for t_{exit}, i.e., the last node of the workflow, it will assign sub-budgets to all workflow tasks and edges. Therefore the *AssignParents* algorithm is responsible for distributing the overall budget among the workflow tasks and edges. Then, in line 11 the procedure *AdjustBudgets* is called which adjusts (increases or decreases) the assigned sub-budgets to each task and edge according to the user's budget. Finally, in line 12 the procedure *Planning* is called to select a service for each task according to its sub-deadline.

3.3 Parents Assigning Algorithm

The pseudo-code for *AssignParents* is shown in Algorithm 2. This algorithm receives an assigned node as input and tries to assign a sub-budget to each of its parents and their corresponding edges (the while loop from line 2 to 13). First, *AssignParents* tries to find the *Partial Critical Path* of unassigned nodes ending at its input node and starting at one of its predecessors that has no unassigned parent. For this reason, it uses the concept of *Critical Parent*.

Definition 1: The Critical Parent of a node t_i is the unassigned parent of t_i that has the latest data arrival time at t_i, that is, it is the parent t_p of t_i for which $EST\,(t_p) + MET\,(t_p) + MTT\,(e_{p,i})$ is maximal.

We will now define the fundamental concept of the PCP algorithm.

Definition 2: The Partial Critical Path of node t_i is:

i empty if t_i does not have any unassigned parent.

ii consists of the Critical Parent t_p of t_i and the Partial Critical Path of t_p if has any unassigned parents.

Algorithm 2 begins with the input node and follows the critical parents until it reaches a node that has no unassigned parent, to form a partial critical path (lines 3-7). Note that

in the first call of this algorithm, it begins with t_{exit} and follows back the critical parents until it reaches t_{entry}, and so it finds the overall critical path of the complete workflow graph.

Then the algorithm calls procedure *AssignPath* (line 8), which receives a path (an ordered list of nodes) as input, and assigns sub-budget to each node and edge on the path. We elaborate on this procedure in the next sub-section. Note that when a sub-budget is assigned to a task, the EST of its successors may change (according to the Eq. 3). For this reason, the algorithm updates these values for all tasks of the path in the next loop. Finally, the algorithm starts to assign sub-deadlines to the parents of each node on the partial critical path, from the beginning to the end, by calling *AssignParents* recursively (lines 9-12).

3.4 Path Assigning Algorithm

The *AssignPath* algorithm receives a path as input and assigns sub-budgets to each of its nodes and edges, including all incoming/outcoming edges from/to the previously assigned nodes of the workflow. The proposed algorithm, first tries to find the cheapest schedule which can finish the whole path before the earliest start time of all assigned children of the last task on the path. Then it uses this primary schedule to assign sub-budgets to the nodes and the edges of the path. As this is just an estimation and not a real schedule, it does not consider the free time slots of the services.

The Algorithm is shown in Algorithm 3, and it is based on a *Backtracking* strategy. It starts from the first task on the path and moves forward to the last task, at each step selecting the next slower available service for the current task (line 6). If there is no available untried service for a task left then the algorithm backtracks to the previous task on the path and selects another service for it (lines 7-9). Then, if the current task is the last task on the path, the algorithm checks if the current schedule is *admissible* and has a lower cost than the current best schedule, it is recorded as the best schedule. We call an schedule *admissible* if it finishes before the EST of all assigned children of the last task on the path. At the end of the while loop (line 19), the algorithm checks if an admissible schedule has been found, then the sub-budgets is set according to the best schedule. Otherwise, the sub-budgets is set according to the *minimum schedule* which is the schedule that has the minimum finish time.

3.5 Budget Adjustment Algorithm

In the budget distribution phase, the algorithm tries to assign sub-budgets to all tasks and edges of the workflow, such that it can finish before the *estimated makespan*. However, since the estimated makespan is just an approximation of the minimum makespan of the workflow which can be achieved using the user specified budget, there may exists a budget difference, *BD*, which can be computed as follows:

Algorithm 3 Optimized Path Assigning Algorithm

```
1: procedure AssignPath(path)
2:     compute path sub-budget according to Eq. 4
3:     best ← null
4:     t ← first task on the path
5:     while (t is not null) do
6:         s ← next slower service ∈ S_t
7:         if (s = ï£¡) then
8:             t ← previous task on the path and continue while loop
9:         end if
10:        if (t is the last task on the path) then
11:            if (current schedule is admissible and has a lower cost than
    best) then
12:                set this schedule as best
13:            end if
14:            t ← previous task on the path
15:        else
16:            t ← next task on the path
17:        end if
18:    end while
19:    if (best is not null) then
20:        set ESTs and sub-budgets according to best
21:    else
22:        set ESTs and sub-budgets according to the minimum schedule
23:    end if
24:    mark all tasks of the path as assigned
25: end procedure
```

Algorithm 4 Planning Algorithm

```
1: procedure Planning(G(T, E))
2:     Queue ← t_entry
3:     while (Queue is not empty) do
4:         t ← delete first task from Queue
5:         query available time slots for each service from related GSPs
6:         compute SS(t) according to Eq. 6 and 7
7:         distribute excess/deficiency in budget over the unscheduled tasks
8:         make advance reservation of t on SS(t)
9:         for all (t_c ∈ children of t) do
10:            if (all parents of t_c are scheduled) then
11:                add t_c to Queue
12:            end if
13:        end for
14:    end while
15: end procedure
```

$$BD = B - (\sum_{t_i \in T} sub-budget(t_i) + \sum_{e_{i,j} \in E} sub-budget(e_{i,j}))$$

This budget difference can be positive or negative, in each case it should be distributed over all tasks and edges of the workflow. To compute the share of each task or edge, the ratio of its sub-budget to the total sub-budgets of all task and edges is multiplied by the BD. Finally, the computed share is added to the current sub-budgets.

3.6 Planning Algorithm

In the planning phase, we try to select the best service for each task of the workflow to create an optimized schedule which executes the whole workflow within the user budget. In the budget distribution phase, each task and each edge were assigned a sub-budget. If we schedule each task such

that the total execution and transmission cost of that task and all of its incoming edges is less than or equal their total sub-budgets, then the whole workflow will finish within the users budget. Our algorithm is based on a *Greedy* strategy that tries to create an optimized global solution by making optimized local decisions. At each stage it selects a ready task, i.e., a task which all of whose parents have already been scheduled, and then schedules it on the fastest service which can execute it within its total sub-budget. The total sub-budget of a task t_i, $TB(t_i)$, is the sum of its sub-budget and the sub-budgets of all of its incoming edges. The selected service for a ready task t_i, $SS(t_i)$, is computed as follows:

$$\begin{aligned} &\min_{s \in S_i} \{AST(t_i, s) + ET(t_i, s)\} \\ &s.t. \sum_{t_p \in t_i's\ parents} TC(e_{p,i}, SS(t_p), s) + EC(t_i, s) \leqslant TB\,(t_i) \end{aligned} \tag{6}$$

where the Actual Start Time of a task t_i on service s, $AST(t_i, s)$, is the maximum between the latest data arrival time of the parents of t_i to the service$s$, and the start time of the suitable free time slot on the service s. There may be extra budget for a scheduled task which should be distributed over unscheduled tasks in proportion to their total sub-budgets.

It is also possible that no service can execute t_i within its sub-budget, In that case, we just select the service with the minimum cost, i.e., $SS(t_i)$ is computed as follows:

$$\min_{s \in S_i} \sum_{t_p \in t_i's\ parents} TC(e_{p,i}, SS(t_p), s) + EC(t_i, s) \tag{7}$$

This deficiency in budget should be distributed over unscheduled tasks in proportion to their total sub-budgets. The *Planning* algorithm is shown in Algorithm 4.

4. Performance Evaluation

In this section we will present the results of our simulations of the Budget-PCP algorithm.

4.1 Experimental Setup

We have used GridSim [10] for simulating the utility grid environment for our experiments. We simulate a grid environment like DAS-3 [11] which is a multicluster grid in the Netherlands. It consists of five clusters, which are: Vrije University with 85 nodes, University of Amsterdam with 41 nodes, Delft University with 68 nodes, MultimediaN with 46 nodes and Leiden University with 32 nodes. The average inter-cluster bandwidth is between 10 to 512 MB/s. We have changed the processor speeds to make a 10 times difference between the fastest and the slowest cluster. Besides, we assigned fictitious prices to each cluster that follow the rule that a faster cluster has a higher price.

We compare our algorithm with the *Loss* algorithm proposed by Sakellariou et al. [12]. They proposed two algorithms for scheduling a workflow under budget constraint, i.e., *Loss* and *Gain*. The Loss algorithm initially schedules the workflow using a time minimization algorithm such as HEFT[13], and then refines the schedule until its budget constraint is satisfied. The Gain algorithm initially assigns each task to its cheapest resource, and then tries to refine the schedule to shorten the execution time while meeting the budget constraint. Their experiments show that the Loss algorithm has a better performance than the Gain, so we have compared our algorithm to the Loss.

In order to compare the performance of these two algorithms, we test them using five synthetic workflow applications which are described in [14]. These workflows are based on real scientific workflows: Montage, CyberShake, Epigenomics, LIGO and SIPHT. Furthermore, there are different sizes for each workflow application in terms of total number of tasks, from which we select the large size one with about 1000 tasks. We assume that all services are installed on all clusters, such that each task can be executed on every cluster. In addition, we assume that all clusters are empty in the beginning.

To assign a budget to each workflow, we employ the same method which is used in [12], i.e., nine possible budgets are assigned to each workflow as follows:

$$B = C_c + \alpha(C_H - C_c)$$

where C_c is the cost of the cheapest assignment which assigns each tasks to its cheapest resource, C_H is the cost of scheduling the workflow with the HEFT strategy which is the most expensive assignment, and α is the *budget factor* which ranges between 0.1 and 0.9. Obviously, if the budget is less than C_c then the problem has no solution, and if the budget is more than C_H then the solution is the schedule which is created by the HEFT strategy. Also, the coefficient k in Eq. 5 is set to 0.8 in our experiments.

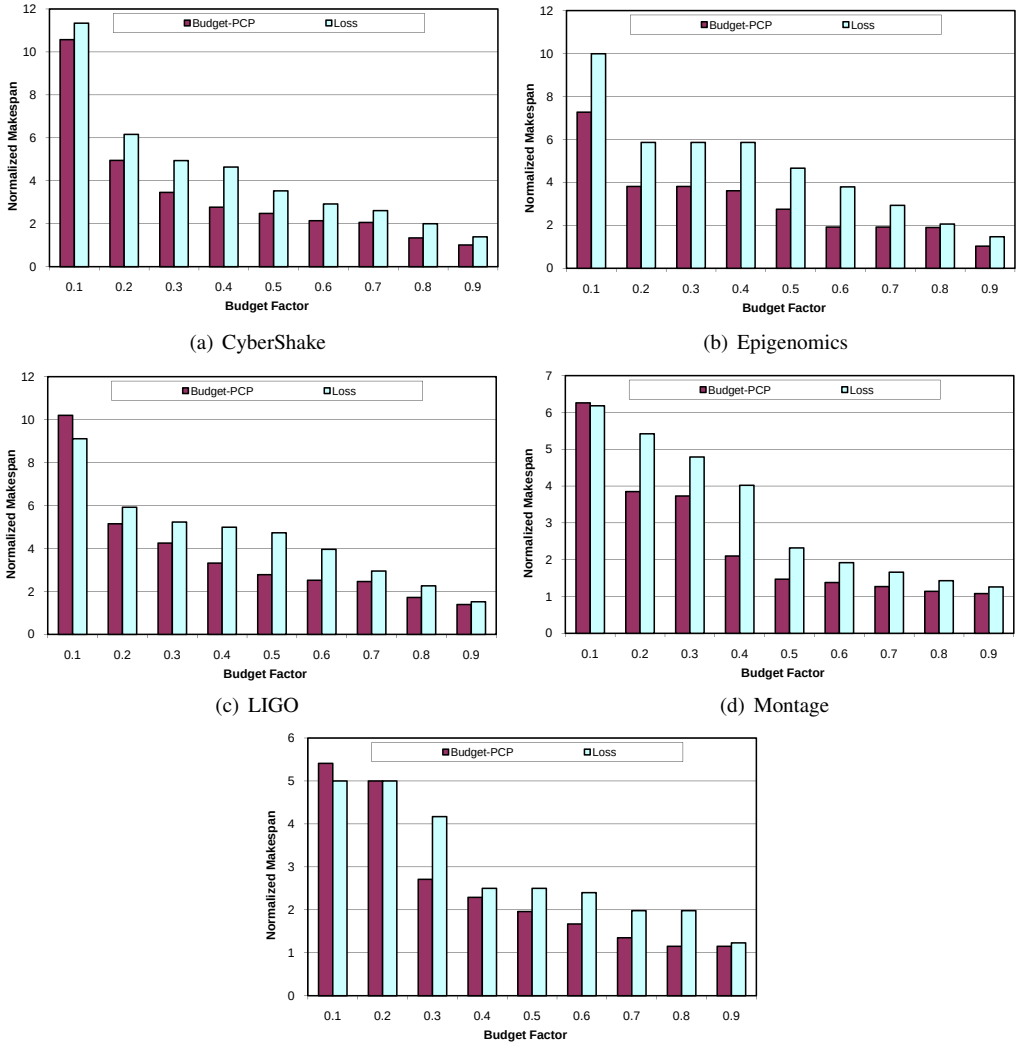
4.2 Experimental Results

In order to compare the performance of the algorithms for each of the five workflows, we define the Normalized Makespan (NM) of a workflow execution as follows:

$$NM = \frac{schedule\,makespan}{M_H} \tag{8}$$

where M_H is the makespan of executing the same workflow with the *HEFT* strategy.

Figure 1 shows the normalized makespan of scheduling each workflow with Budget-PCP and Loss algorithms. The figure shows that the Budget-PCP outperforms the Loss algorithm, except for $\alpha = 0.1$ in LIGO, Montage and SIPHT workflows. The average makespan decrease of the Budget-PCP over the Loss is 26.06% for CyberShake, 33.12% for



(a) CyberShake

(b) Epigenomics

(c) LIGO

(d) Montage

(e) SIPHT

Fig. 1: Normalized Makespan of scheduling of the workflows with Budget-PCP and Loss

Epigenomics, 19.99% for LIGO, 24.49% for Montage and 18.61% for SIPHT.

5. Related Work

There are few works addressing workflow scheduling with QoS in the literature. We have already mentioned the Loss and Gain algorithms proposed by Sakellariou et al. [12]. Yu et al. [3] used Genetic Algorithm to solve both problems: cost optimization under deadline constraint, and execution time optimization under budget constraint. In another work, Yu et al. [15] used three multi-objective evolutionary algorithms, i.e., NSGAII, SPEA2 and PAES for this problem. In this work, the user can specify his desired deadline and maximum budget and the algorithm proposes a set of alternative trade-off solutions meeting the user's constraints from which he can select the best one according to his needs. In a similar research, Talukder et al. [16] proposed a Multiobjective Differential Evolutionary algorithm which generates a set of alternative solutions for the user to select. Chen and Zhang [17] proposed an Ant Colony Optimization (ACO) algorithm, which considers three QoS parameters, i.e., time, cost and reliability. They enable the users to determine constraints for two of these parameters, and the algorithm finds the optimized solution for the third parameter while meeting those constraints. Finally, Tao et al. [6] considered a multi dimensional QoS parameters like time, cost, reliability, availability, reputation, and security, and tried to optimize the weighted sum of these parameters using Particle Swarm Optimization (PSO).

6. Conclusions

Utility Grids enable users to obtain their desired QoS (such as deadline) by paying an appropriate price. In this paper we propose a heuristic algorithm named Budget Partial Critical Paths (Budget-PCP) for workflow scheduling in utility Grids that minimizes the total execution time such that the total execution cost does not exceed the user specified budget. We evaluate our algorithm by simulating it with synthetic workflows which are based on real scientific workflows with different structures. The results show that Budget-PCP outperforms another well-known algorithm called Loss.

References

[1] E. Deelman, "Grids and clouds: Making workflow applications work in heterogeneous distributed environments," *Int. J. High Perform. Comput. Appl.*, vol. 24, pp. 284–298, August 2010.

[2] D. Laforenza, "European strategies towards next generation grids," in *Proc. of The Fifth Int'l Symposium on Parallel and Distributed Computing (ISPDC '06)*, 2006, p. 11.

[3] J. Yu and R. Buyya, "Scheduling scientific workflow applications with deadline and budget constraints using genetic algorithms," *Sci. Program.*, vol. 14, no. 3,4, pp. 217–230, 2006.

[4] M. Wieczorek, A. Hoheisel, and R. Prodan, "Towards a general model of the multi-criteria workflow scheduling on the grid," *Future Generation Computer Systems*, vol. 25, no. 3, pp. 237 – 256, 2009.

[5] I. Brandic, S. Benkner, G. Engelbrecht, and R. Schmidt, "QoS support for time-critical grid workflow applications," in *Int'l Conference on e-Science and Grid Computing*, 2005, pp. 108–115.

[6] Q. Tao, H. Chang, Y. Yi, C. Gu, and Y. Yu, "Qos constrained grid workflow scheduling optimization based on a novel pso algorithm," in *Eighth International Conference on Grid and Cooperative Computing*, 2009, pp. 153 –159.

[7] I. Brandic, S. Pllana, and S. Benkner, "Specification, planning, and execution of qos-aware grid workflows within the amadeus environment," *Concurr. Comput. : Pract. Exper.*, vol. 20, pp. 331–345, 2008.

[8] J. Yu, R. Buyya, and C. K. Tham, "Cost-based scheduling of scientific workflow applications on utility grids," in *First Int'l Conference on e-Science and Grid Computing*, July 2005, pp. 140–147.

[9] S. Abrishami, M. Naghibzadeh, and D. Epema, "Cost-driven scheduling of grid workflows using partial critical paths," in *Proceedings of the 11th IEEE/ACM Int'l Conference on Grid Computing (Grid2010)*, 2010.

[10] R. Buyya and M. Murshed, "Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing," *Concurr. Comput. : Pract. Exper.*, vol. 14, no. 13, pp. 1175–1220, 2002.

[11] The distributed asci supercomputer. [Online]. Available: http://www.cs.vu.nl/das3/

[12] R. Sakellariou, H. Zhao, E. Tsiakkouri, and M. D. Dikaiakos, "Scheduling workflows with budget constraints," in *Integrated Research in GRID Computing*, ser. CoreGRID Series, S. Gorlatch and M. Danelutto, Eds., 2007, pp. 189–202.

[13] H. Topcuoglu, S. Hariri, and M. Wu, "Performance-effective and low-complexity task scheduling for heterogeneous computing," *IEEE Trans. on Parallel and Distributed Systems*, vol. 13, no. 3, pp. 260–274, 2002.

[14] S. Bharathi, A. Chervenak, E. Deelman, G. Mehta, M.-H. Su, and K. Vahi, "Characterization of scientific workflows," in *The 3rd Workshop on Workflows in Support of Large Scale Science*, 2008.

[15] J. Yu, M. Kirley, and R. Buyya, "Multi-objective planning for workflow execution on grids," in *Proceedings of the 8th IEEE/ACM International Conference on Grid Computing*, 2007, pp. 10–17.

[16] A. K. M. K. A. Talukder, M. Kirley, and R. Buyya, "Multiobjective differential evolution for scheduling workflow applications on global grids," *Concurr. Comput. : Pract. Exper.*, vol. 21, pp. 1742–1756, 2009.

[17] W. N. Chen and J. Zhang, "An ant colony optimization approach to grid workflow scheduling problem with various QoS requirements," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 39, no. 1, pp. 29–43, 2009.

An Efficient Model to Minimize Makespan On Data Grids Considering Local Tasks

Asst. Prof. Nahed M. El Desouky[1] , Dr. Bayoumi. M. Ali Hassan[2] ,
Dr. Sahar A. Gomaa[1], Afaf Abd El-kader Abd El-hafiz[1]
[1]Departement of Mathematics, Computer Branch, Faculty of Science , Al-Azhar University(girls)
Cairo, Egypt.
[2] Cairo University,Fci-Cu, DS Dep., Cairo, Egypt.

Abstract - *Scheduling divisible loads on the resources of a grid has gain a great attention in last years. In grids, load scheduling plays a crucial role in achieving high utilization of resources. Many scheduling algorithms assume that grid resources such as CPU power is constant which mean that they are static algorithms. In dynamic, non_ dedicated environment such as grids, this assumption is not suitable where in grids, distributed computers (workers) are assumed to process local tasks in addition to grid incoming tasks. M. Othman and et. al. presented in [1] the IDLT (Iterative Divisible Load Theory) model that takes into account the divisible load communication time as well as the divisible load computation time in such a way that the entire processing time of the load is minimum. In this paper we propose a new model LIDLT (Local tasks Iterative Divisible Load Theory) that study the effect of workers' local tasks on the makespan. We compare the makespan produced by the two algorithms. The results shows that the proposed model LIDLT is good as the IDLT model.*

Keywords: Scheduling divisible loads**,** grid computing, local tasks

1 Introduction

Many applications in scientific and engineering domains are structured as large numbers of independent tasks with low granularity. Such applications have been called divisible loads because a scheduler may divide the computation among worker processes arbitrarily, both in terms of number of tasks and of task sizes.

Divisible load scheduling has been an active area of research for the last fifteen years. A vast literature offers results and scheduling algorithms for various models of the underlying distributed computing platform. Broad surveys are available that report on accomplishments in the field[2]. Scheduling divisible loads in grids is one of the active topics in the literature nowadays. A critical issue for the performance of a grid is the task scheduling problem i.e. the problem of how to divide a task into many parts and assign

them to computers of the grid, so that the execution time of the entire load(makespan) is minimum.

Many algorithms [4, 5, 6] for scheduling divisible loads assume that computational resources are dedicated. This assumption makes these algorithms not practical in environments such as grids. In grids, computational resources are expected to execute local tasks in addition to grid tasks i.e. they are non-dedicated workers[3]. A shortcoming of these algorithms is that they do not take the dynamicity of Grids into account. An efficient scheduling algorithm should factor in such changes in CPU capacity depend on affection by local tasks.

The UMR algorithm [4] is one of the most valuable algorithms for scheduling divisible loads. Authors of [3] improve the UMR by improving it with the mixed tendency-based strategy for predicting the CPU utilization of workers. A load distribution model IDLT (Iterative Divisible Load Theory) is designed in [1] by taking into account the communication time as well as the computation time in such a way that the entire processing time of the load is minimum.

The rest of the paper is organized as follows: section 2 describes the model of applications and platforms used here. The IDLT algorithm is briefly described in section 3. In section 4, we incorporate the local tasks in the IDLT model. The proposed algorithm is given in section 5. Experimental results are given in section 6. Finally, conclusion is given in section 7.

2 Applications Model

This work considers applications that consist of a workload, W_{total} that is continuously divisible. The scheduler determines the size of chunk (the portion of workload that is given to a worker). Both the transfer of application data input and the transfer of application data output will be considered.

The model considered here is a star-shaped master/worker model with N workers. The total load W_{total} is submitted to the master into rounds $round_j$, where $W_{total} = \sum_{j=1}^{n} round_j$. Chunks are sent out to workers over a network by the master which uses its network connection in a sequential fashion. It does not send chunks to workers simultaneously. This is a common assumption and is justified either by the master's implementation, or by the properties of the network links (e.g. a LAN)[3]. The speeds of network communications to each worker need not be identical.

Therefore, the platform topology consists of network links with various characteristics to the set of heterogeneous processors.

It is also assumed that the workers can receive data from the network and perform the computation simultaneously.

3 The IDLT Model

In many data intensive grid applications, data can be decomposed into many parts to be executed on the resources of a grid. Authors of [1] design a load distribution model IDLT by taking into account the communication time and the computation time in such a way that the entire processing time of the load is minimum. They proposed new optimal closed form formulas for scheduling divisible load on large scale data grid system. Closed-form expressions for the processing time and the fraction of workload for each processing node are derived. Since, the closed form did not consider the communication time in [8]. The new model considers computation time as well as communication time. A new closed form solution is proposed for obtaining the optimal fraction a_i as follow:

Let T_i be the time at which the worker w_i of speed s_i and bandwidth b_i finishes its work. And let a_i be the fraction of load that is given to worker w_i

$$a_i / s_i + a_i / b_i = T_i \quad , \quad i=1, 2, \ldots, n \qquad (1)$$

$$T_1 = T_2 = ... = T_M = T \qquad (2)$$

$$a_i / s_i + a_i / b_i = T \qquad (3)$$

$$a_1+a_2+\ldots+a_{M-1}+a_M = 1 \qquad (4)$$

From Eq. (3)

$$a_i = T / (1/s_i +1/ b_i) \quad , \quad i=1, 2, \ldots, n \qquad (5)$$

$$T / (1/s_1 +1/ b_1) + T / (1/s_2 +1/ b_2) + \ldots + T / (1/s_n +1/ b_n) = 1 \qquad (6)$$

$$T =1 / \sum_{i=1}^{n} 1/(1/s_i +1/b_i) \qquad (7)$$

and the fraction a_i given to worker w_i is

$$a_i= T /(1/s_i +1/ b_i) \quad , \quad i=1, 2, \ldots, n \qquad (8)$$

After we get T, we can get a_i by Eq. (5).

4 The LIDLT Model

The proposed algorithm LIDLT (Local tasks Iterative Divisible Load Theory) considers local tasks in this model. Let d_{ji} is the size of the local task that will be executed on w_i then the finishing time of worker w_i can be calculated using (1) as follows:

$$a_i / s_i + a_i / b_i = T_i + d_i / s_i \quad , \quad i=1, 2, \ldots, n \qquad (9)$$

According to the optimality criterion in [8] equation (3) become

$$a_i / s_i + a_i / b_i = T + d_i / s_i \qquad (10)$$

$$a_i = (T+ d_i / s_i) / (1/s_i +1/ b_i) \quad , \quad i=1, 2, \ldots, n \qquad (11)$$

$$(T+ d_1 / s_1) / (1/s_1 +1/ b_1) + (T+ d_2 / s_2) / (1/s_2 +1/ b_2) + \ldots + (T+ d_n / s_n) / (1/s_n +1/ b_n) = 1 \qquad (12)$$

$$T =(1-\sum_{i=1}^{n} (d_i / s_i) /(1/s_i +1/ b_i)) /(\sum_{i=1}^{n} 1/(1/s_i +1/b_i)) \qquad (13)$$

and the fraction a_i given to worker w_i in round j is

$$a_i= (T+ d_{ji} / s_i) /(1/s_i +1/ b_i) \quad , \quad i=1, 2, \ldots, n, \ j=1, 2, \ldots, M \qquad (14)$$

After we get T, we can get a_i by Eq. (14). By calculating the a_i, the optimal time will be calculated[1].

4.1 The LIDLT algorithm

The LIDLT can be formally stated as follows. Let u_t be the utilization of the current worker at time t and u_{t-1} be its utilization at time t-1.

```
for j := 0 → m
  for  i  := 0 → n
    ave := (u_{t-1} + u_t ) / 2.0;
    utilization[i] := ave;
    sort utilization in decreasing order
    chunk array[i]:= ( T+ d_ji / s_i ) /(1/s_i +1/ b_i ) round_j  ,
          i=1, 2, …, n,  j=1, 2, …, M
    sort the chunk array in increasing order
  for i  := 0 → n
       assign chunk[i] to worker [i]
makespan ← the length of the longest queue
```

The results shows that the proposed algorithm has an equivalent performance as the IDLT algorithm.

5 Experimental Results

A simulation is performed for the LIDLT algorithm and the IDLT algorithm to analyze the performance of each algorithm. The simulation is written using the C++ language.

The model consists of n workers with n varying from 5 to 50. The round length is generated randomly using the gamma distribution function with beta varying from 1G to 50G. The speed of processors, the computation and communication overheads $clat_i$, $nlat_i$ respectively and the values of bandwidth b_i are all chosen randomly.

The local tasks is also generated using gamma function with fixed arrival time lunda=.5 and beta = 400. To show the benefit of IDLT against LIDLT, Figure 1 plots the makespan versus the workload for both the IDLT and LIDLT algorithms.

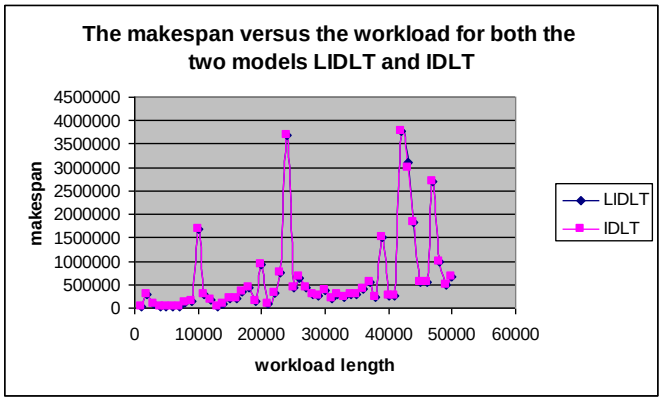


Figure 1: the makespan versus the workload for both the IDLT and LIDLT algorithms

6 Conclusion

The IDLT (Iterative Divisible Load Theory) model for scheduling divisible loads takes into account the communication time as well as the computation time in such a way that the entire processing time of the load is minimum. This paper proposes a new model LIDLT (Local tasks Iterative Divisible Load Theory) that study the effect of workers' local tasks on the makespan. We compare the makespan produced by the two algorithms. The results shows that the proposed model LIDLT is good as the IDLT model.

7 References

[1] Othman, M., M. Abdullah, H. Ibrahim and S. Subramaniam, "Load Allocation Model for Scheduling Divisible Data Grid Applications"; Journal of Computer Science 5 (10): 760-763, 2009.

[2] O. Beaumont, H. Casanova, A. Legrand, Y. Robert, and Y. Yang. "Scheduling Divisible Loads on Star and Tree Networks: Results and Open Problems"; *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 16(3):207–218, 2005.

[3] Said Elnaffar and Nguyen The Loc. "Enabling Dynamic Scheduling in Computational Grids by Predicting CPU Utilization"; WSEAS Transactions on Communications Issue 12, Volume 4, pages 1419-1426. December 2005, ISSN:1109-2742

[4] Yang Yang and Henri Casanova, "UMR: A Multi-Round Algorithm for Scheduling Divisible Workloads"; Proceeding of the International Parallel and Distributed Processing Symposium (IPDPS'03), Nice, France, April 2003.

[5] Yang Yang and Henri Casanova, "A Multi-Round Algorithm for Scheduling Divisible Workload Applications: Analysis and Experimental Evaluation"; Technical Report CS2002-0721, Dept. of Computer Science and Engineering, University of California, 2002.

[6] O. Beaumont, A. Legrand, and Y. Robert, "Scheduling Divisible Workloads on Heterogeneous Platforms"; Parallel Computing, Volume 29 , Issue 9 September 2003.

[7] Bharadwaj, V., D. Ghose and T.G. Robertazzi, "Divisible load theory: A new paradigm for load scheduling in distributed systems"; Cluster Comput., 6: 7-17, 2003.

[8] Wong, H.M., D. Yu and B. Veeravalli, "Data intensive grid scheduling: Multiple sources with capacity constraints"; Proceeding of the IASTED Conference on Parallel and Distributed Computing and Systems, Nov. 3-5, Marina Del Rey, USA., pp: 7-11, 2003.

Using BitNBD for Provisioning Virtual Machines in OpenCirrus Testbed

Yong-Ju Lee, Hag-Young Kim
Dept. of Cloud Computing Research
Electronics and Telecommunications Research Institute
South Korea
{yongju, h0kim}@etri.re.kr

***Abstract*—BitTorrent is a convenient tool for downloading and distributing very large files. This feature throws us a clue if virtual machine images in distributed environments are possible to move from one site to another. Users also gain the ability to have their computer that follows them rather than be tied to one physical location. In this paper, we explore how the BitTorrent protocol can be applied so that BitTorrent-based provisioning for virtual machines can improve cloud resource management and reduce response time.**

Keywords: BitNBD, VM Provisioning, Cloud Computing

I. INTRODUCTION

Cloud computing is the emergent technology that promises on-demand, dynamic, and easily accessible computing power. Dynamic provisioning of computing resources using Operating System (OS) and Virtual Machine (VM) technologies can be considered.

With the advent of virtualization technologies like Xen [1], VMware [2], and KVM [3], data centers have been going through a revolution in design. Virtualization enables multiple operating systems and applications, to run concurrently on a single server without interfering with each other. Today, it leads to turn general clusters into an Infrastructure-as-a-Service (IaaS) cloud. In terms of VM images management, VM images are cloned from a base image at first. The cloning process is executing a copy command on the raw image file. The cloned images are then copied using secure shell (SSH) or shared via NFS onto the virtualization nodes. The time taken to provision an instance is not neglected. There are open source IaaS cloud solutions like Eucalyptus [4], Open Nebula [5], Nimbus [6], ASPEN [7], and Tashi [8]. In general, image management uses local file copy and distribution via NFS. This potentially means that a more efficient method can be used for VM images distribution.

Efficient resource distribution mechanisms provide just-in-time resource provisioning in cloud computing, and it would affect the execution of a cloud service. In this paper, we explore how the BitTorrent mechanism can support provisioning virtual resources, and reveal benefits of the on-demand provision of virtualized resources as a service.

II. RELATED WORK

The OpenCirrus testbed [9] in figure 1 is a collection of federated data centers for open-source systems and services research. Currently, the testbed is composed of 10 sites in North America, Europe, and Asia. The project is a joint initiative sponsored by HP, Intel, and Yahoo!, in collaboration with other institute. ETRI has joined in 2009.

Figure 1. OPENCIRRUS SITES

A typical OpenCirrus service stack consists of physical resource set (PRS), Tashi, Ganglia [10], and other application framework services (e.g., Hadoop [11], Pig [12]). A PRS is a set of VLAN-isolated compute, storage, and networking resources. At this time, a cluster is partitioned into one or more PRS dominants, dynamically allocated, and managed by a PRS service, at the request of PRS clients.

This research contributes to advancing the resource set whether they are physical resources or virtual resources. Our BitTorrent-based provisioning shows the benefits of the on-demand provision of physical or virtualized resources as a service and energy-efficient consolidation for cloud computing. It provides faster deployment to meet growing demand and improve performance.

Figure 2 illustrates testbed pictures of the cluster being installed in late 2010 at ETRI. The platform currently consists of 512 nodes (4096 cores), each with quad-core X3320 processors and 4GBs of main memory. Each compute node has 500GB storage and two 1Gb/s link to each core switch.

Figure 2. ETRI SITE

Figure 3 illustrates BitTorrent-based Network Block Device (BitNBD) architecture for provisioning virtual machines efficiently[13]. A client has both NBD client module and our BitNBD module. The NBD client module communicates with the BitNBD module for reading and writing blocks. The BitNBD module transfers virtual machine image with other peers and provides read/write operations.

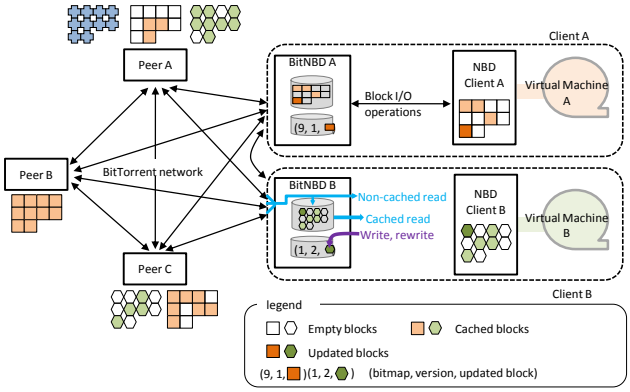


Figure 3. BITTORRENT-BASED NETWORK BLOCK DEVICE (BITNBD) ARCHITECTURE

The BitNBD also uses the "split read/write" mechanism where the data is read from other peers while the data is being written locally. Read/reread blocks are obtained by neighbor peers or local caches. Write/rewrite blocks can be written as separately. It contains block bitmap, block version number, and updated block's data to allow the BitNBD to record which blocks have modified.

In figure 3, the steps for constructing a virtual machine in a configuration of the BitNBD: 1) A users uses a network block device to request a VM. Internally, the network block devices already mounted. 2) The VM template disk image is shared to other compute nodes. 3) The BitTorrent module in client side connects the image server for obtaining appropriate peers to distribute a VM. 4) The virtual machine emulator(e.g., QEMU) requests booting blocks to the network block device module. 5) The network block device module also requests the blocks to the BitNBD module. 6) The BitNBD module processes the block I/O operations.

III. CASE STUDY OF PROVISIONING VMS

A. Windows Provisioning Issues for VMs

Our image server is based on Linux platform. For that reason, Windows mkfs and mount utilities under Linux is necessary to deploy Windows images in our image server. Cloning Windows images has problems including master boot record(MBR), different disk ids, Windows volume serial number, and extended attributes(ACL, Junction, and so on). Especially, A Disk Identifier (or Disk Signature) applies to an entire hard disk drive (not a single partition).

A Disk Identifier/Disk Signature is a 4-byte (longword) number that is randomly generated when the Master Boot Record/Partition Table is first created and stored. The Disk Identifier is stored at byte offset 1B8 (hex) through 1BB (hex) in the MBR disk sector. Windows Vista/7 uses the Disk Signature to locate boot devices so changing it can prevent them from booting. So far as I know Grub and Linux don't use the Disk Identifier.

Especially, extended attributes saved a script file(e.g., gc_fattr.sh). In Windows Vista/7, the default locations for user data and system data have changed. For example, user data that was previously stored in the %SystemDrive%\Documents and Settings directory is now stored in the %SystemDrive%\Users directory. In order to minimize the amount of application compatibility problems users would experience due to the change in folders names, such as C:\Documents and Settings moving to C:\Users, Microsoft employed junction points. A Junction Point is a physical location on a local hard disk that points to another location on that disk or another storage device.

A volume boot record (also known as a volume boot sector or a partition boot sector) is a type of boot sector, stored in a disc volume on a hard disk, floppy disk, or similar data storage device, that contains code for booting programs stored in other parts of the volume.

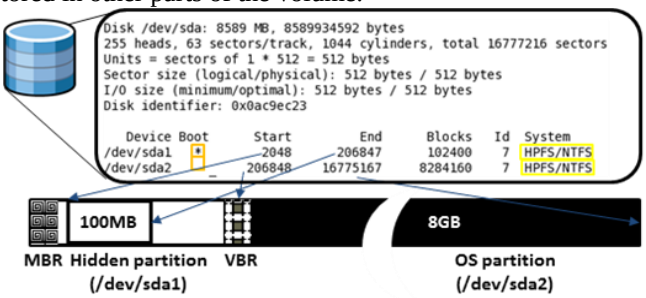


Figure 4. DISK LAYOUT IN WINDOWS 7

Figure 4 illustrates a disk layout in Windows 7. Windows 7 has two partitions. The first partition is a hidden partition which has a boot loader. The second partition is the OS partition including user data. The steps or obtaining entire files in two partitions: 1) the Windows golden client boots via dhcp, generates information(i.e., partition information, disk identifier from basic disk, and volume serial number from partitioned disk) and then invokes rsync daemon for a image server. 2) The image server gets all files in Windows client via rsync protocol. The cloning process is done. 3)

The client machine for installation boots via dhcp, creates partitions, updates the disk identifier, makes file system layout, updates volume serial number in each partition, mounts, and copies all file from the image repository in the image server.

B. *Intercloud Issues of the BitNBD*

The OpenCirrus testbed, in nature, federates numerous, far-away sites with various hardware, services, and tools. Figure 5 illustrates the conceptual view of the Intra/Intercloud for deploying the BitNBD. The intracloud(e.g., ETRI site) in South Korea has abundant network bandwidth. Thus, the BitNBD can achieve its role for distributing VM images efficiently. The Intercloud(e.g., between ETRI site and others), however, potentially offers an alternative way to manipulate cross-cloud transfers and to minimize possibly worries about security. The BitNBD for Intercloud will in near future employ the locality-awareness strategy to create near-optimal sets of collaborative network block devices.

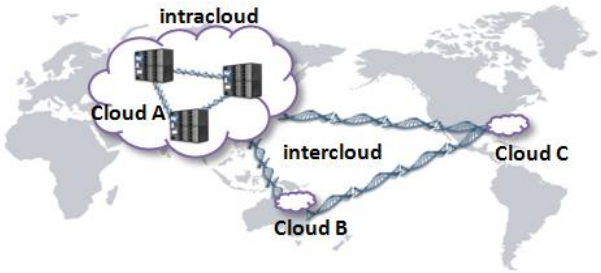


Figure 5. INTRA/INTERCLOUD

Figure 6 illustrates the locality-aware BitNBD diagram. There is the gatekeeper (for example, G1 and G2) that interconnects between Cloud A and Cloud B. It conducts two kinds of activities for sharing pieces.

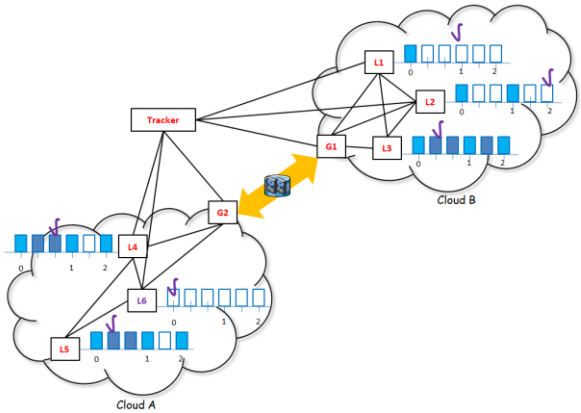


Figure 6. LOCALITY-AWARE BITNBD DIAGRAM

One thing is breaking the data transfer bottleneck. Our main idea for fast transport prtocol is inherited from UDT that is a reliable UDP based application level data transport protocol. It is a highly configurable framework that can accommodate various congestion control algorithms. The other is bias BitNBD selection located in the tracker. Basically, the tracker responses the peer list that is based on the random selection. In our case, the tracker in the image server achieves biased BitNBD selection which is based on cross-ISP hops. It will choose its neighbor mostly from those within the same ISP, and only have a few that are outside the ISP.

C. *Dynamic Service Provisioning*

Initial deployment and subsequent dynamic reconfiguration of a software is difficult because of many interdependdent factors including software dependencies and installing orders. We have developed a dynamic SW provisioning technique for carrying out the deployment of softwares. A service consists of several software packages that are used rpm-based installation procedures. For rapid deployment of services, dynamic deployment policy is required.

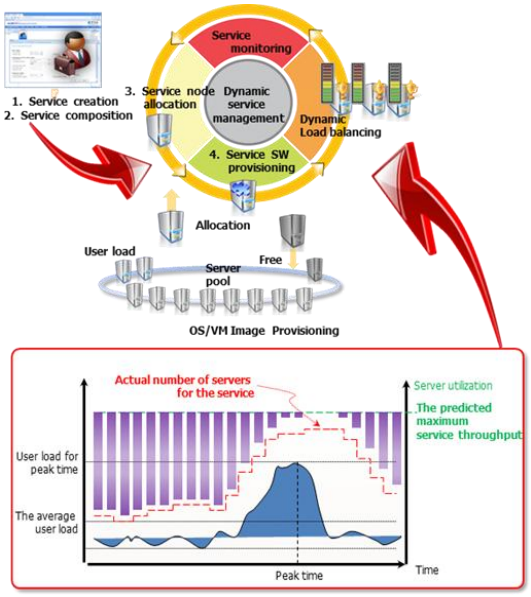


Figure 7. SERVICE PROVISIONING CYCLE WITH DYNAMIC LOAD BALANCING

Figure 7 shows an example of service provisioning cycle that initiated by a user. The user creates a cloud service via web-based GUI. Then, the service is submitted to the system through the service composition command. The service manager determines appropriate nodes at the initial time and automates software provisioning based upon a set of role definitions for a user. As the nodes are overloaded or underloaded, load balancing occurs dynamically. Therefore, dynamic service management increases service utilization.

IV. PERFORMANCE EVALUATION

Figure 8 shows booting time of instances ranging from 10 to 50 VMs. Local boot takes about 1 minute 25 seconds. Their images are located in 50 physical nodes independently. In NFS boot case, all the images are

located in a NFS server. Thus, NFS boot leads to network congestion when a large number of VMs accesses simultaneously. In BitNBD boot case, single image is located in a BitNBD server. BitNBD boot has little interference with a large number of VM instances.

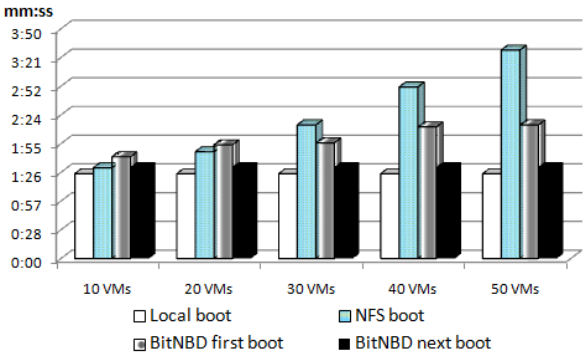


Figure 8. AVERAGE BOOT TIME OF VM INSTANCES

It shows that VMs can start up in 104 to 134 seconds. Our BitNBD method is better than NFS boot method in terms of VM startup. Most of all, Local boot is superior to other methods in terms of boot times. After first boot, BitNBD receives booting related blocks from other peers. "BitNBD first boot" has a slight startup delay compared to NFS boot at the case of 10 or 20 VMs. "BitNBD next boot" means that the BitNBD has locally cached blocks related booting stage. The boot time of "BitNBD next boot" is exactly same as the time of "Local boot".

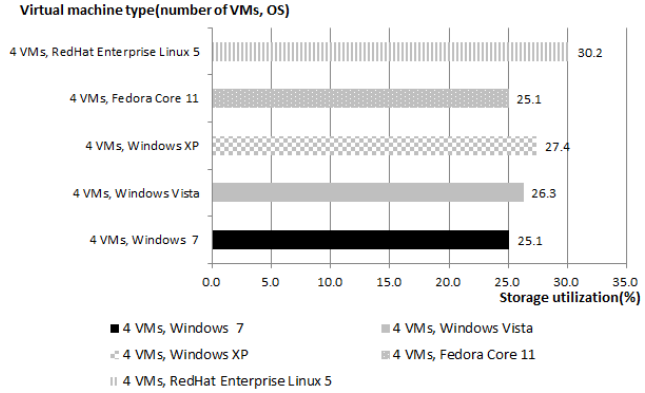


Figure 9. Storage utilization of the BitNBD(4VMs)

Figure 9 shows comparison of total storage usage stored by traditional method and BitNBD method. It contains the 4 VMs and OS types (five kinds of OS distributions). For example, "Windows 7, 4 VM" of BitNBD storage means that four VMs running Windows 7 are executed concurrently. With the BitNBD mechanism, just one copy of the base image of the virtual machine would be stored, plus any user-specific differences. As the number of VMs in BitNBD case increases, the total size of storage used in VMs is steady. Traditional storage case, however, increases in proportion to the number of VMs. For example, a traditional storage mechanism would require 10 gigabytes for each of the 1000 virtual machines, equaling 10 terabytes of storage. However, the BitNBD mechanism only uses 2.5 terabytes, which is 4 times less than 10 terabytes with traditional storage mechanisms. As a result, our BitNBD method is a cost effective way to improve storage utilization and offer fast deployment.

V. CONCLUSIONS

This paper gives you a solid idea that virtual machine images are possible to distribute efficiently via the BitTorrent protocol. Our BitTorrent-based provisioning shows the benefits of the on-demand provision of virtualized resources as a service. In the case study of provisioning VMs in OpenCirrus testbed, we proposed windows provisioning issues, Intercloud issues for deploying VMs. Moreover, it enhanced dynamic service provisioning for service-level management.

As part of our future work, we plan to develop an optimal scheduling algorithm that supports SLA-based resource allocation to meet Quality of Experience (QoE) requirements.

REFERENCES

[1] N. Fallenbeck, H.J. Picht, M. Smith, and B. Freisleben. Xen and the art of cluster scheduling. Proceedings of 1st International Workshop on Virtualization Technology in Distributed Computing, 2006.
[2] VMWare. http://www.vmware.com/.
[3] Kernel Virtual Machine. http://kvm.qumranet.com/kvmwiki.
[4] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov. The Eucalyptus open-source cloud-computing system. Proceedings of Cloud Computing and Its Application, 2008.
[5] B. Sotomayor, R. S. Montero, I. M. Llorente, and I. Foster. Virtual infrastructure management in private and hybrid clouds. IEEE Internet Computing, 13:14–22, 2009
[6] Nimbus, http://workspace.globus.org.
[7] R. Curry, C. Kiddle, N. Markatchev, R. Simmonds, T. Tan, M. Arlitt, and B. Walker. ASPEN: an automated service provisioning environment for data centres. Proceedings of the 15th HP Software University Association Workshop, 2008.
[8] Kozuch, M. A., Ryan, M. P., Gass, R., Schlosser, S. W., O'Hallaron, D., Cipar, J., Krevat, E., López, J., Stroucken, M., and Ganger, G. R. Tashi: location-aware cluster management. ACDC 2009.
[9] http://www. opencirrus.org
[10] Matthew L. Massie, Brent N. Chun, David E. Culler. The ganglia distributed monitoring system: design, implementation, and experience. Parallel Computing, 15 June, 2004
[11] http://hadoop.apache.org/
[12] http://hadoop.apache.org/pig/
[13] Yong-Ju Lee, Hag-Young Kim, Cheol-Hoon Lee, BitNBD: BitTorrent-Based Network Block Device for Provisioning Virtual Machines in IaaS Clouds, IEICE Transactions on Information and Systems, Vol. E94-D, no. 1, pp. 60-68, January 2011.