People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University Abdelhamid Mehri, Constantine 2
Faculty of New Technologies of Information and Communication
Department of Fundamental Computer Science and its Applications
Laboratory of Modelling and Implementation of Complex Systems
MISC, Scal Group

Year : 2017
N :.....
Serial :.....

Doctoral Thesis in Computer Science

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Sciences

Efficient, Scalable and Economics-based Resource Management in Peer-To-Peer Grids

By

Nabila Chergui

Thesis committee members :

President :	Zizette Boufaida Pr. Univ Abdelhamid Mehri - Constantine2
Supervisor :	Pr. Salim Chikhi Pr. Univ Abdelhamid Mehri - Constantine2
Examiners :	
	Abdelmalik Bachir Pr. Univ Mohamed Khider - Biskra
	Sadek-Labib Terrissa Associate Pr. Univ Mohamed Khider - Biskra
	Mohamed Gharzouli Associate Pr. Univ Abdelhamid Mehri - Constantine2
Invited :	Mohand-Tahar Kechadi Pr. University College Dublin - Ireland

Defended on : 07-11-2017

Abstract

Desktop Grid Computing is a type of distributed computing systems that is built on the idea of harnessing idle CPU cycles, storage space and other resources of networked computers to solve complex problems on a larger scale with low cost. Traditional server-client Desktop Grid architectures have inherent problems in reliability and scalability. The convergence between Peer-to-Peer (P2P) Computing and Desktop Grid Computing leads to the creation of P2P Desktop Grid Computing that has been developed to address the above raised issues. In this environment, resources are heterogeneous, have different individual qualities and distributed over the Internet; yet, P2P Grid requires a complex resource management process to handle the heterogeneity, the dynamicity and the scalability of the system, and to offer good services to its users. In the current study, we proposed an efficient, scalable and economics-based resource management for P2P Grid Computing.

First of all, we addressed the problem of resource discovery. We applied Semantic Web technologies and ultra-peers paradigms to construct a three layered overlay network based on a Simple Knowledge Organization System (SKOS) lightweight ontology; that describes domains of applications in P2P Desktop Grid Computing, and a semantic clustering of nodes according to their domains of interest. Additionally, we relied on an efficient semantic-based routing process that improves the overall system efficiency and response time.

Second, we designed a scalability-aware approach based on workload prediction and management for ultra-peers systems to prevent the presence of bottleneck in these structured systems; moreover, help each ultra-peer to find its steady state. The proposed solution allowed to the ultra-peer to scale with the growth of the network size.

Finally, we suggested a distributed and economics-based resource allocation approach for P2P Grid. We modelled the system components by utility functions and used a multilateral bargaining mechanism to optimise both users and providers objectives.

To evaluate the effectiveness of the recommended solutions, we have created a discrete-event simulator composed of several components to cover the studied resource management processes. Then, for each projected approach, we conducted intensive simulations to evaluate its performances. Results demonstrated the efficiency of the proposed approaches.

Keywords: P2P Grids, Resource Management, Resource Discovery, Overlay Architecture, Semantic Web, SKOS, Ultra-peer Architecture, Scalability, Workload Prediction, Resource Allocation, Economic approach, Bargaining, Optimisation.

ملخص

الحوسبة الشبكية (شبكة حواسيب سطح المكتب) هي نوع من الحوسبة الموزعة، التي بنيت على فكرة إستغلال الدورات غير المستخدمة من وحدات المعالجة المركزية،من مساحات التخزين وغيرها من موارد الكمبيوترات المتصلة بالشبكة، لحل المشاكل المعقدة على نطاق واسع وبتكلفة منخفضة. الهندسة التقليدية من نوع الزبون و المزود لشبكات حواسيب سطح المكتب تنطوي على عدة مشاكل متعلقة بالمتانة، الموثوقية وقابلية التوسع. أدى التقارب بين حوسبة النّد للنّد وحوسبة شبكة حواسيب سطح المكتب إلى إنشاء ما يسمى بشبكة حوسبة النّد للنّد، و التي طورت للإجابة على المشاكل المذكورة آنفا. في مثل هذه البيئة، غالبا ما تكون الموارد غير متجانسة، لها صفات فردية مختلفة و موزعة عبر الشبكة العنكبوتية؛ شبكة حوسبة النّد للنّد تتطلب عملية معقدة لإدارة الموارد للتعامل مع عدم التجانس، الديناميكية و توسع النظام وذلك لتقديم خدمات أفضل للمستخدمين. في هذه الأطروحة اقترحنا منهجا فعالا و قابلا للتوسيع معتمدا على الاقتصاد لإدارة الموارد في شبكة حوسبة النّد للنّد.

أولا، تعاملنا مع مشكلة اكتشاف الموارد. لذلك استخدمنا تقنيات الويب الدلالي ونماذج فائقة النظير لبناء شبكة افتراضية مكونة من ثلاث طبقات باستخدام أنثولوجيا على شكل نظام تنظيم المعرفة البسيط لتمثيل مجالات التطبيقات في مثل هذه البيئة و من أجل تجميع العقد وفقا لمجالات إهتمامهم ، بالإضافة إلى عملية فعّالة لتوجيه طلبات البحث عن الموارد على الأساس الدلالي أيضا مما سمح على تحسين كفاءة النّظام.

ثانيا، قمنا بتصميم نهج واع للتوسيع على أساس التنبؤ بأعباء العمل وإدارته للأنظمة فائقة النظير للوقاية من مشكلة الإحتقان في مثل هذه الأنظمة المهيكلة، مما يسمح لكل عنصر من النظام على إيجاد الاستقرار. النهج المقترح عزز بشكل كبير قابلية التوسع لهذه الأنظمة كلما زاد حجم الشبكة.

أخيرا، اقترحنا طريقة موزعة مبنية على طرق النّظريات الإقتصادية لتخصيص الموارد لشبكات النّد للنّد. مثّلنا مكونات النّظام عن طريق دوال المنفعة، و استخدمنا آلية المساومة لأمثلة أهداف المستخدمين ومقدمي الخدمات.

لتقييم نجاعة الحلول المقترحة، قمنا بإنشاء برنامج للمحاكاة باستخدام الأحداث المنفصلة متكون من عدة أجزاء لتغطية جميع عمليات إدارة الموارد المدروسة. بعد ذلك، قمنا بإجراء سلسلة مكثفة من التجارب لتقييم كفاءة كل حل من الحلول. أظهرت النتائج المحصل عليها فعالية النُّهُج المقترحة.

الكلمات المفتاحية: شبكة حوسبة النّد للنّد، إدارة الموارد، اكتشاف الموارد، الهندسة الافتراضية، الواب الدلالي، SKOS، الهندسة فائقة النظير، قابلية التوسع، التّنبؤ بأعباء العمل، تخصيص الموارد، النهج الاقتصادي، المساومة، الأمثلة.

Résumé

Les Grilles de calcul des PCs (Grilles de Desktop) sont un type des systèmes informatiques distribués construit sur l'idée d'exploiter les cycles non utilisés des CPUs et de l'espace de stockage, en plus des autres ressources des ordinateurs disponible sur le réseau, pour résoudre des problèmes complexes à une plus grande échelle et à faible coût. Les architectures traditionnelles de type client-serveur pour les Grilles de Desktop ont des problèmes liés à la fiabilité et au passage à l'échelle. La convergence entre les systèmes de pair à pair $P2P$ et les Grilles de Desktop conduit à la création des Grilles de P2P, qui ont été développés pour répondre aux questions ci-dessus. Dans ce type d'environnements, les ressources sont hétérogènes, ont des qualités individuelles différentes et distribuées sur Internet ; les Grilles de P2P nécessitent un processus de gestion des ressources complexes pour gérer l'hétérogénéité, la dynamicité et l'évolutivité du système et pour offrir de bons services à ses utilisateurs. Dans cette thèse, nous avons proposé une approche efficace, évolutive et économique pour la gestion des ressources dans les Grilles de P2P.

Tout d'abord, nous avons abordé le problème de la découverte de ressources. Nous avons appliqué les technologies du Web sémantique et les paradigmes d'ultra-pairs pour construire un réseau virtuel à trois couches basé sur une ontologie légère au format du SKOS(Système Simple d'Organisation de Connaissances) qui décrit les domaines des applications dans les Grilles de P2P, et un groupement sémantique de nœuds en fonction de leurs domaines d'intérêt, en plus à un processus de routage sémantique qui améliore les performances du système.

Deuxièmement, nous avons conçu une approche consciente à l'évolutivité basée sur la prédiction et la gestion de la charge de travail pour les systèmes ultra-pairs, afin de prévenir la présence du goulot d'étranglement dans ces systèmes structurés et aide chaque ultra-pair a trouvé son état de stabilité. La solution proposée a permis à l'ultra-pair de passer à l'échelle avec la croissance de la taille du réseau.

Finalement, nous avons proposé une approche économique et distribuée pour l'allocation des ressources pour les Grilles de P2P. Nous avons modélisé les composantes du système par des fonctions d'utilitées, et nous avons utilisé un mécanisme de négociation multilaterale pour optimiser les objectifs des utilisateurs et des fournisseurs.

Pour évaluer l'efficacité des solutions proposées, nous avons créé un simulateur à événements discrets composé de plusieurs composantes pour couvrir les processus étudiés de la gestion des ressources. Pour chaque approche proposée, nous avons effectué des simulations intensives pour évaluer ses performances. Les résultats obtenus ont démontré l'efficacité des approches proposées.

Mots clé : Grilles de P2P, Gestion de Ressources, Découverte de Ressources, Architecture Virtuelle, Web Sémantique, SKOS, Architecture d'Ultra-pair, Passage à l'Echelle, Prédiction de la Charge du Travail, Allocation de Ressources, Approche Economique, Négociation, Optimisation.

Acknowledgements

This work would not have been possible without the inspiration and the generous help offered by many people. I owe a huge debt to my PhD supervisor Mr. Salim Chikhi, Professor at Abdelhamid Mehri University Constantine 2 and Mr. Mohand-Tahar Kechadi Professor at the University College Dublin, not only for the freedom they gave me in my research; but also for their guidance, patience, motivation and enthusiasm.

Thanks to Professor Salim Chikhi who continually inspired, trusted and supported me during many years of our joint work. His guidance helped me to complete this work.

I want to express my gratitude to Professor Mohand-Tahar Kechadi, for hosted me into his lab, providing the best conditions to my internships, and for giving me all his time, attention and interest. The knowledge and skills I acquired were very valuable to me for the realisation of this work.

I would like to thank Ms Zizette Boufaida, Professor at Abdelhamid Mehri University Constantine 2, for having doing me the immense honour to preside over the jury of my viva.

I warmly thank Mr. Abdelmalik Bachir, Professor at the University of Biskra, Mr. Sadek-Labib Terrissa, Associate Professor at the University of Biskra and Mr Mohamed Gharzouli, Associate Professor at Abdelhamid Mehri University Constantine 2 for having agreed to report this thesis. I express my gratitude for their interest in this work, their remarks will surely contribute to enrich our thinking.

I would like to express my respect and thanks to all members of the team of MISC laboratory with whom I have collaborated, as well as, to all the lecturers at Ferhat Abbas University Setif 1 with whom I work. Additionally, I want to express my gratitude to Karim and Ismail for their comments, advises, recommendations and unconditional help.

Special thanks also go to Soumia, Radia, Zakaria, Ramdhan, Billel, Tania and Khalid for their help and support.

My deepest gratitude goes to my family. Thank you for being my endless sources of love and encouragement.

Last but not the least, my heartfelt gratitude goes to my friends for sharing with me good and bad times, encouraging, and supporting me.

Contents

List of Figures

List of Tables

To my mother, to my family

Chapter 1

Introduction

"Science is a way of thinking much more than it is a body of knowledge."
—— Carl Sagan

A distributed computing system is a system architecture that makes a collection of heterogeneous computers, workstations, or servers acts and behaves as a single computing system [Har+04]. This latter deals with all forms of computing, information access, and information exchange across multiple processing platforms connected by computer networks [Ksh+11]. In such a computing environment, users can uniformly access and name local or remote resources, and run processes from anywhere in the system, without being aware of which computers their processes are running on [Har+04]. Actually, we are interested in two major types of distributed computing systems, Grids and P2P computing systems.

1.1 Grid Computing

Grid computing has emerged as a natural response to the increasing demands in term of resources from computing applications which are becoming more complex and increasingly collaborative and interdisciplinary. The Grid couples a wide variety of geographically distributed and heterogeneous computational resources (PCs, workstations, mainframes, and clusters), data storage devices, databases, and dedicated scientific instruments. So, by nature, it is a distributed, heterogeneous and a dynamic environment. The most common definition provided to the Grid is:

"*Grid computing is a coordinated resource sharing and problem-solving in dynamic, multi-institutional virtual organisations*" [Fos+01].

Where the resource sharing means:

" *The sharing that we are concerned with is not primarily the file exchange, but rather direct access to computers, software, data, and other resources, as is required by a range of collaborative problem-solving and resource brokering strategies emerging in industry, science, and engineering. This sharing is, necessarily, highly controlled, with resource providers and consumers defining clearly and carefully just what is shared, who is allowed to share, and the conditions under which sharing occurs. A set of individuals and/or institutions defined by such sharing rules form what we call a virtual organisation*" [Fos+01].

There exist two different trends in the development of Grid systems: Service Grids and Desktop Grids [Bal+08].

1.1.1 Service Grids

Service Grids were created to be accessed by a large number of users. A resource can become a part of the Grid by installing a middleware. However, the middleware is very complex to set up and maintain for non-expert individuals and often requires costly expert effort to maintain. Therefore, they are economically worthwhile only for large organisations and institutions where professional system administrators take care of the hardware, middleware, software environment to ensure high availability of the Grid. Examples of such large-scale Service Grids are Grid'5000[1], the NorduGrid[2], etc. OGSA, the Open Grid Services Architecture [Fos+05] is a service-oriented Grid computing, developed from the idea of [Fos+02], it is composed of interoperable Web Services representing the available resources. The OGSA specification defines a set of services for identity management, authentication and authorisation, service level agreement negotiation and monitoring, management and communication within virtual organisations, integration of data resources into computations, managing, and monitoring collections of services. Globus Toolkit, gLite [gLi], The European Middleware Initiative EMI [EMI] are examples of service grids that adhere the OGSA.

1.1.2 Desktop Grids

Desktop Grids systems [Cho+08] provide high computational power at low cost by reusing a potentially large set of existing infrastructure of resources owned by many independent individuals. These computing infrastructures aim at harnessing idle CPU cycles, storage space and other resources of networked desktop computers; instead of dedicated clusters and supercomputers to work together on a particularly computational intensive application. Though, they represent a low-cost alternative to traditional Grids. Small-scale installations such as comprising the workstations of a department [Chi+03], as well as, large-scale Internet-wide approaches [And04] have been successfully implemented. Desktop Grids differ significantly from traditional distributed systems. Particularly, the aggregated resources join and leave the Grid in an unpredictable manner, the phenomenon referred to as volatility [Kon+04; Bha+03].

1.2 P2P Computing

P2P computing consists of interconnected peers or nodes that are able to self-organise into network topologies where each peer can collaborate, exchange and share data and services with a set of other peers. When a peer consumes resources of the other peers, it becomes a client; and while it provides its own resources to other peers, it turns to a server.

1. https://www.grid5000.fr/mediawiki/index.php/Grid5000
2. http://www.nordugrid.org/

P2P employs distributed resources to perform a function in a decentralised fashion. These resources can be computing power, data (storage and content), network bandwidth, etc. A function can be a distributed computing, data sharing, platform services, etc [Mil+03].

The most common definitions provided to this concept are:

"*P2P systems and applications are distributed systems without any centralised control or hierarchical organisation, in which each node runs software of equivalent functionality*"[Sto+01].

This means that no centralised control is allowed in these systems and all nodes must have an equal role.

P2P systems are capable of adapting to failures and accommodating transient populations of nodes while maintaining acceptable connectivity and performance, without requiring the inter-mediation or support of a global centralised server or authority [And+04].

In Schollmeier's definition [Sch01], the P2P systems are divided into two categories. Those that use no centralised services are described as pure P2P networks. Yet, the ones that incorporate some level of centralisation are identified as hybrid systems.

From the above definitions, we can perceive that there is no global knowledge about the whole system, and that the decentralisation may apply to algorithms, data, and meta-data, or to all of them, although it does not prohibit the existence of some centralisation in some part of the system and the applications. Thence, the absence or the restricted existence of centralisation makes the system attractive and offers to it several advantages:

— Scalability: P2P systems are inherently scalable in several ways: the load on the existing centralised servers is reduced, the computation and network traffic are distributed among the involved peers; thus scalability is enhanced. In addition, each new peer joins the system brings new resources which make the system scales gracefully and enriches it;

— Fault tolerance: lack or a restricted central point of failure, in addition that regardless the peer failures, P2P networks are able to self-repair and reorganise; hence, providing robustness to the infrastructure.

Nevertheless, P2P systems have some drawbacks. Peers and communication links are not reliable. Both node and communication failures are to be expected; besides, nodes may join and leave the system at any time, the thing known as churn.

1.3 Taxonomy of P2P Networks

P2P networks usually implement an abstract overlay topology over existing physical networks. Nodes in a network overlay are connected through logical links. Where each link corresponds to a path in the underlying physical network. To join the network, a peer must connect to another peer already participating in the network, so to acquire knowledge about others peers populating the network [Kav13].

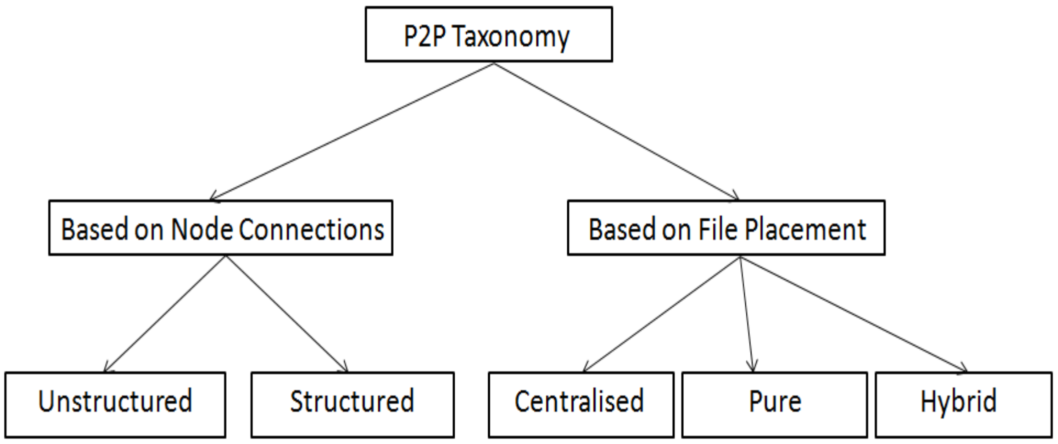


FIGURE 1.1 – P2P Taxonomy.

P2P overlay networks can be classified see Figure 1.1 on centralised, pure (or fully) distributed and hybrid according to the system architecture and how the data files are placed [Ebe+05]. Or they can be categorised based on network structure and node connections into structured and unstructured P2P systems. Yet, both structured and unstructured concepts can be used in hybrid overlay organisations. Figure 1.2 from [Ebe+05] depicts the P2P classification based on the system architecture and file placement.

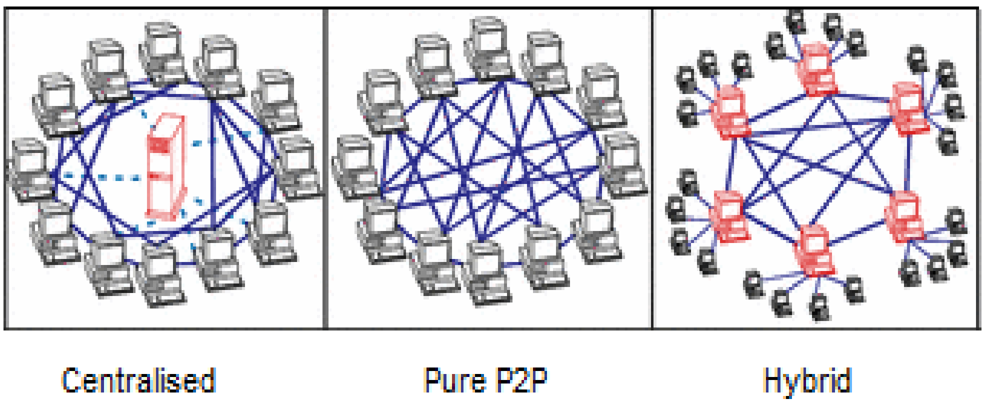


FIGURE 1.2 – P2P network architectures.

The structure refers to how the nodes in the network overlay are linked to each other [Kav13]. The links between peers in an unstructured network are arbitrary and there is no relationship between the data and where peer resides. Each peer is randomly connected to a fixed number of other peers (neighbours); accordingly, the overlay network is created non-deterministically. However, in structured P2P, the content is placed not at random peers but at specified locations that will make subsequent queries more efficient [Lua+05]. Peers are interconnected and organised in a deterministic structure as well as the file indexes.

1.3.1 Unstructured P2P

Unstructured P2P systems do not use any rules to organise peers and the placement of data files. Peers do not have any vision about the position of contents which are placed at random peers. They are connected to a random number of other peers called neighbours. The only way to locate a resource is to send a query to all nodes in the system through the query flooding technique, or to a set of neighbours via the technique of random walk. Examples of the popular unstructured systems are Gnutella, Kazaa, [Wal+00],Edutella [Nej+02] and FreeHaven [Din+00].

1.3.2 Structured P2P

Structured P2P overlays are a tightly controlled topology. The contents are not placed at random peers, but rather at specified locations that make subsequent queries more efficient [Lua+05]. Peers are structured based-on one or a combined geometrical structure; where each peer has a responsibility for specific objects. Consequently, it uses appropriate routing mechanisms to locate resources. The queries do not need to be flooded through the overlay network, but can be routed directly to the responsible peer. Examples of structured P2P systems include distributed trees [Zha+05; Fre+02], and layered random graphs [Law+03], the distributed hash table (DHT), ex:Can [Rat+01], Chord [Sto+01], kademlia, Tapestry and Pastry[Zha+04; Row+01].

1.3.3 Centralised P2P

Centralised P2P systems use a central index server to store information about all peers. These latter consult the central server to find resources, then communicate directly with the relevant peers to access those resources, example Napster. Centralised systems do not inherit the failure tolerance. The loss of the index server will mean that resources cannot be found. The scalability of a centralised P2P system may also be limited by the load placed on the central server by large numbers of peers. The server would eventually be a bottleneck.

Except the fact of sharing resources; by today's standard and according to the previous definitions, this architecture is no longer a P2P system.

1.3.4 Pure P2P

Pure P2P networks are a fully distributed system, as their name indicate. They organise all the peers at one level without using any sort of hierarchy. All the peers participating in the overlay network have equal responsibility and no single node having a global picture. Example, Gnutella v.0.4 (the earlier version of Gnutella) [Rip01] and the basic form of Chord are examples of this category.

1.3.5 Hybrid P2P

Hybrid P2P networks combine different concepts in the overlay network organisation in order to improve search efficiency. From the architectural perspective, this network is a mixture of centralised and pure P2P where peers are organised into different levels based on their capabilities, reliability, network bandwidth, etc. In addition, peers may take different roles; some powerful peers are placed on the high-level and designed as super or ultra-peers, or even as central index servers. Peers placed at the lower-level are named peers. From the overlay topology and nodes connections perspectives, this network can utilise different network topologies at each level of the overlay; for example, the upper-level can be a structured P2P overlay while the lower-level can be an unstructured P2P overlay. The search requests are handled on the ultra-peers at the upper-level, resulting in much fewer messages than in centralised and unstructured systems. The hybrid systems provide more fault tolerance and scalability.

1.4 Taxonomy of Desktop Grid Computing

There are several taxonomies of Desktop Grids [Ven16; Fed15; Zha+11; Fed+08; Cho+07; Chi03]. From the architectural and the design aspects, Desktop Grid computing can be categorised into centralised or distributed Desktop Grids.

1.4.1 Centralised Desktop Grid

Centralised Desktop Grid system consists of three logical components: a master server, workers and clients [Ven16]. Clients are jobs' submitters, and workers are resources providers and the components running on the Desktop Grid which are responsible for executing jobs. The master server coordinates a set of workers by assigning them some jobs from clients. It receives jobs submissions from clients and distributes them to its workers according to the scheduling policy. Then it collects the corresponding results and ultimately assembles them. It usually assumes other responsibilities such as resources management and fault tolerance. Typical examples are Seti@home [And+02], Condor [Tan+02], BOINC [And04], XtremWeb [Fed+01], and Entropia [Chi+04] etc. To obtain significant benefits, non-negligible efforts may be required to set up and maintain the master server [Naz+03]. This design could potentially face issues with scalability and single point of failure [Fed10; Cho+07]. The convergence between P2P computing and Desktop Grid computing leads to the creation of distributed or P2P Desktop Grid computing. They have been developed to address the above issues of centralised systems.

1.4.2 Distributed Desktop Grid

Distributed Desktop or P2P Grid Computing (we will simply refer to it as P2P Grid for all the rest of this document) constructs computational overlay networks using tree, graph or distributed hash table. The fully distributed architecture is based on the principle that all the participating entities have the same roles, responsibilities and rights [Ang+10]. Participants to the P2P Desktop Grid share and use a set of resources on a reciprocity basis [Ang+10] and need to maintain partial information about other participants in the Grid environment [Ven16]. Scheduling is performed at each participant or volunteer depending on the computational overlay network where volunteers exchange their information between other volunteers. Typical examples are Compute Power Market [Buy+01c], OurGrid [Cir+06], the Organic Grid [Cha+05], Messor [Mon+03], Paradropper [Zho+03], and the one developed by [Kim+07].

The construction of P2P Desktop Grid on structured P2P overlays improves scalability in large-scale settings [Naz+07; Iam+04]. The volunteers can be organised into a computational overlay network based on criteria such as resource capability, semantics or time zone. This convey to another alternative of P2P Desktop Grid, the hybrid P2P system [Ven16], where the notion of super-peer or ultra-peer is used. This architecture does adopt centralised approach in one form or in another; for example, JXTA and JINI.

1.5 Overall Architecture of P2P Grid Computing

Zhao and his colleagues [Zha+11] propose a typical layered architecture for P2P Grid see Figure 1.3. It represents a general system structure and its core components, and it identifies essential entities in a typical P2P Grid system: users, tasks and resources. The architecture is built with four horizontal layers, in addition to two other vertical layers. Each layer encapsulates core functions and fosters the next layer by several services.

- User management: determines end user operational model;
- Task Management: handles search issues related to user's submitted jobs;
- Security management: at the user level, it mostly relies on user identity verification. At the task level, it concerns with task data privacy protection. Where at the resources management, it cares of link disconnection, incentive mechanisms, secure management, sand-boxing...
- Reliability management: addresses the system design aspects;
- Resource management: covers both P2P network management and computing resource management, the former aims at organising geographically distributed desktop nodes into virtual P2P network topology, then exposes an abstract resource pool to the latter, which in its turn, provides several services like resources discovery, matching and monitoring to respond to users request and to execute their jobs.

1.6 Background and Motivation

P2P Grids resources' are distributed, heterogeneous and belonging to several administrative domains. Still, they differ significantly from service Grids,

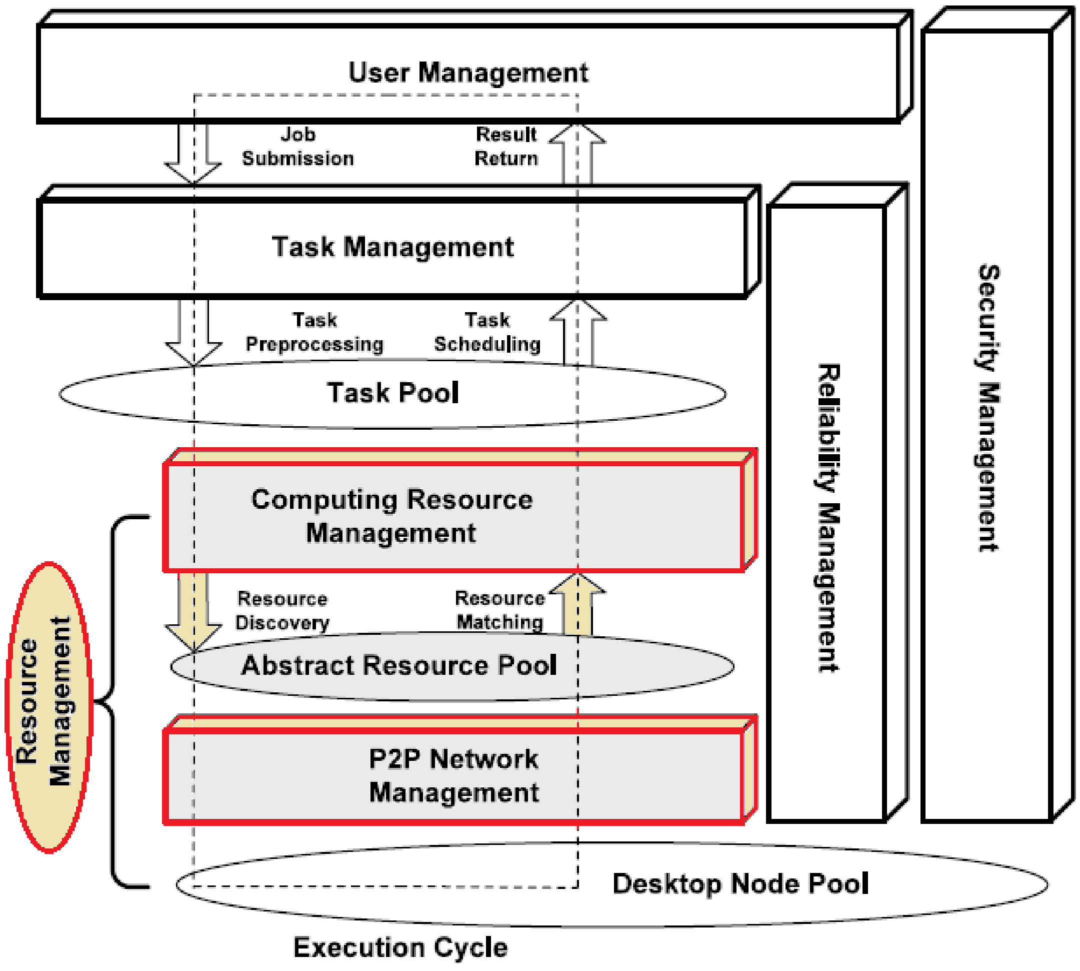


FIGURE 1.3 – The overall architecture of P2P Grids.

particularly, from the volatility of the aggregated resources, means that resources join and leave the system in an unpredictable manner. In addition, they differ too from the non-dedication of the resources and from the fact that resources are limited due to constraints determined by the resource owners, which are the users of the computers [Sch+09]. Since P2P Grid capacity grows with the number of resources connected to it; consequently, it should be expanded to a larger scale to construct a powerful system. However, due to the nature of its resources, and to support a substantial number of participants; P2P Grid requires a sophisticated resources management process to handle the heterogeneity, the dynamicity and the scalability of the system, and to offer good services.

1.6.1 Resource Management

Resource management *RM* for P2P Grids is a complex undertaking and particularly challenging amongst Grid resource management because of the heterogeneity in the system and the sharing of resources with other users, as well, P2P Grid must support thousands to millions of computers with low management overhead.

The RM encapsulates several processes as shown in Figure 1.3. It has to efficiently find adequate resources that match users jobs' which may have different resource requirements, with available heterogeneous computational resources dispersed over the Internet. This process is known as the resource discovery. It has also to scale to large configurations and heavy workloads, all with a restricted centralised control and information about the whole system. This requires thinking about the system conception and nodes organisation and management. In addition, the RM has to allocate users jobs to the adequate resources and it needs to continuously adapt to changes in the availability of resources, to offer an acceptable Quality of services QoS.

On the other hand, in these systems, the resource owners and resource users have different goals, objectives, strategies, and requirements that adds other challenges to the RM system. In order to achieve all the above-mentioned challenges and requirements, sophisticated techniques for RM are needed.

For these reasons, first, the economical use of resources and system based on market approaches for RM are of primary interest in P2P Grids [Sch+09]. It offers an incentive to resource providers to enhance the quality and the quantity of the shared resources and for contributing more to the system. Resource consumers need a utility model, representing their resource demand and preferences, where resource providers require a utility model to express their expected benefits from offering resources. Consequently, an economy-based RM would help to build a large-scale computational Grids, by encouraging more resource providers and yielding a fair basis for access to resources, distributing decision-making process, and aiding to minimise free-riders impacts. Besides, it enables both consumers and producers to maximise their utility and ameliorates the regulation of the demand and supply [Buy+02].

Second, the requirement to support a significant number of participants emphasises the importance of scalability in this context. P2P Grids means the construction of Desktop Grid structures on P2P overlays, which improves scalability in large-scale settings [Naz+07; Iam+04]. Hence, nodes must be grouped to create a reliable and robust resource. It is further improved by the concept of ultra-peers which combines advantageous properties of client/server systems and unstructured P2P topologies [Yan+03]. In the same way, coupling this kind of architecture with Semantic Web technologies gives more expressiveness to users requests', nodes and their resources which enhances the matchmaking algorithm and ameliorates its efficiency, and makes the search for resources more accurate.

1.7 Dissertation Contributions

In this dissertation, we propose an approach for RM where we divide the approach into three principle processes; resources discovery, resources and network management, and computing resource allocation. We support the following thesis by efficient, scalable and economics-based resource management for P2P Grid computing.

In accordance with, our system is made efficient by applying Semantic Web technologies with overlay networks to build the system architecture and construct the resource discovery process. We adjust it to be scalable by using the ultra-peer overlay network and workload prediction and management so that it can scale gracefully as more nodes are injected into the system. Moreover, we make it economic incentive through the modelling of the system components by utility functions and by using a bargaining mechanism to perform the allocation.

Figure 1.4 presents an overall view of our contributions for P2P Grid RM.

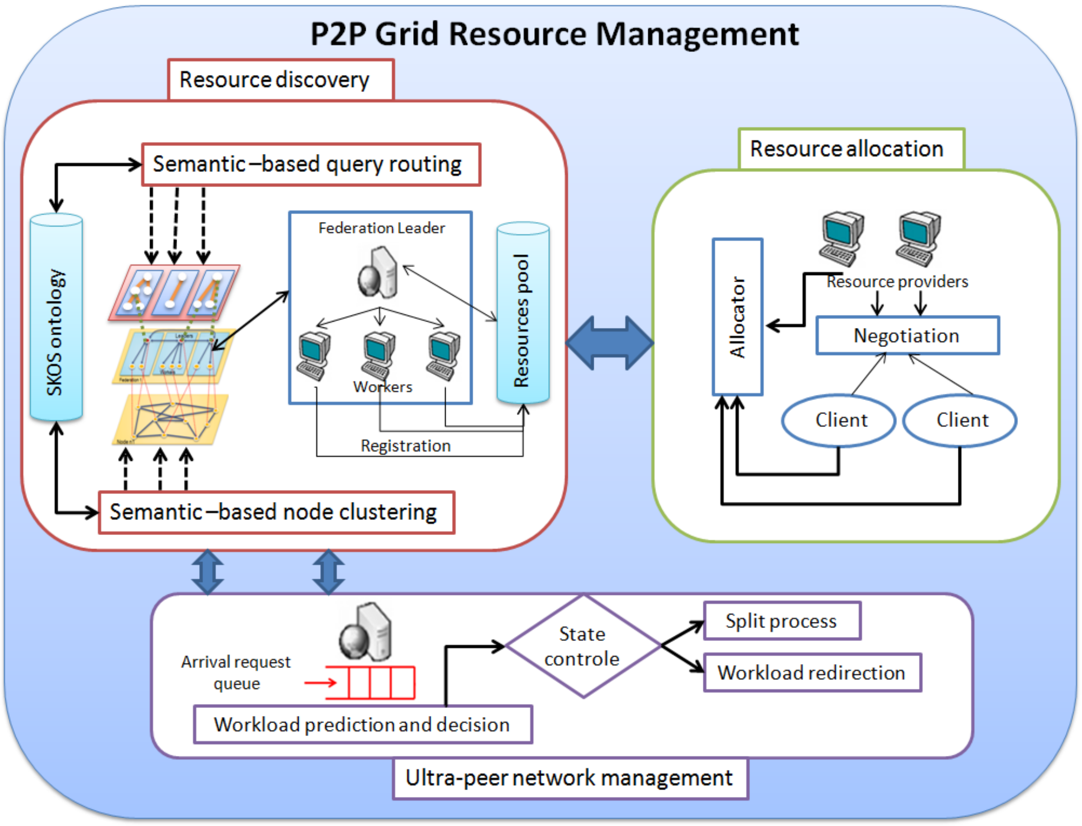


FIGURE 1.4 – Global view of the P2P Grids resources management architecture.

In addition, we have created a simulator to validate the effectiveness of our contributions, which will be reviewed at the end of this section.

More specifically, this dissertation makes the following contributions:

1.7.1 Contribution1: Overlay Architecture and Resource Discovery

The first contribution which will be discussed in detail in Chapter 4 takes into account the system architecture and the resource discovery *RD* process.

RD is a very important process for every RM system that aims at finding adequate resources according to the user request. In Desktop Grid computing, the P2P technology is utilised to achieve decentralisation, self-organising, and scalability. Ultra-peer overlays are an important type of structured P2P overlays which take into account the heterogeneity of peers in the overlay and organise them into an ultra-peer layer and a client-peer layer according to several criteria. This will enhance the scalability and the efficiency of the *RD* because ultra-peers overlays are considered among solutions that guarantee the efficiency and the scalability at the same time. However, the quality of resource discovery is determined not only by efficiency and scalability; but also by its accuracy that measures the quality of the discovered resources in terms of relevance and precision. The lack of a good and effective representation of nodes in this highly heterogeneous environment results in an irrelevant affectation of nodes to the right ultra-peer group. In addition, syntactic keyword and taxonomy-based matching are not sufficient to achieve high-precision resource discovery because of the disagreement with the meaning, interpretation or intended use of terms.

This is considered one of the reasons for which queries can fail to find relevant resources. Because of these factors, we have proposed to combine the ultra-peer technique with Semantic Web technologies to build a layered architecture for P2P Grid RD. Nodes are clustered based on their domain of interest to form groups that we have called federations. We suggested a new process for constructing federations and a three layers overlay network, in addition to a mechanism of routing queries to target federations in an efficient and a scalable way. The proposed process is based on a Simple Knowledge Organisation System *SKOS* lightweight ontology that describes domains of applications in the P2P Grid and gives domains of interest of nodes a well-defined meaning which has improved the effectiveness of nodes' information, resources and query representation, thus the efficiency and the accuracy of searching.

This contribution has been a subject of the publications [Che+15; Che+11b; Che+11c; Che+11a].

1.7.2 Contribution2: Scalability and Workload Management

The second contribution that aims with the scalability of the system will be detailed in Chapter 5.

Scalability is a factor of increasing importance in the design of distributed computing systems and organisations that are expected to grow rapidly like P2P Grids. As mentioned above, we have proposed an ultra-peer overlay architecture for RD and management. In large scale environments, characterised by the huge number of nodes, the ultra-peers models may suffer from scalability problems. Each ultra-peer in the system can become a bottleneck because of the high density of nodes belonging to it and their engendered workload, which may make it suffer from insufficient resources to handle requests, new nodes and their consequent queries too. Controlling the workload is essential since the system scales up rapidly and accommodates a dynamic change in the number of users, resources, etc. Thence, each ultra-peer in the system could become a subject of a drastic change in its components and the incoming requests' rate from its composing nodes or from the other entities, and receive a heavy workload, which will make it suffer from the bottleneck.

We have designed and implemented a scalability-aware approach for ultra-peers like networks, where each ultra-peer can prevent the presence of bottleneck, then regulate and maintain its state to stay in a steady state. For this end, we have used neural networks in conjunction with queueing theory to create a process of prediction and decision. At each prediction cycle, the process estimates the future workload for each ultra-peer, then it takes decisions on whether the next period will become a bottleneck situation or not. If the ultra-peer expects to receive an unmanageable workload, which will cause bottlenecks problems, it redirects an amount of the next incoming workload to another connecting node that functions as a recovery-peer. This strategy enables the ultra-peer to momentarily alleviate the workload on each ultra-peer; otherwise, the ultra-peer should split its cluster and create another new ultra-peer cluster. This strategy makes the proposed model very scalable, allows each ultra-peer to scale with the growth of the network size, and permits to add new ultra-peers' clusters to the system easily.

This contribution has been previously published in [Che+17].

1.7.3 Contribution3: Economics-based resource allocation for P2P Grid Resource Allocation

The third contribution concerns the computing resource allocation *RA* mechanism, it will be detailed in Chapter 6.

RA is a challenge to P2P Grids due to the fact that resource providers and consumers may have different goals, policies, and preferences, and in the most cases are self-interested entities. Resource allocation is the process of distributing limited available and suitable resources among users' tasks, based on some predefined rules of selection to meet the required specifications [Qur+14; Ism07]. Economic mechanisms [Buy+05; Wol+03] are almost based on the fact that user optimisation requirements aim with the budget constraints that he/she is willing to pay as well as the resource cost that is set by the owner.

We have proposed a distributed and economic approach for RA based on the multilateral bargaining mechanism in distributed computing environment. We have described the global system architecture and its main components: users and providers, and we have modelled each component by a utility function to express its preferences, needs and desires. Then we have presented algorithms for multilateral bargaining to optimise the user and provider utilities.

Consequently, this economics-based RA would help to build a large-scale computational P2P Grids by encouraging more resource providers and yielding a fair basis for access to resources, to distribute decision-making process, and to aid to minimise free-riders impacts. In addition, it enables both consumers and producers to maximise their utility. Economic incentives provide motivations to users and providers to pursue their preferences.

1.7.4 Contribution4: Validation of the Results

The validation of the feasibility and effectiveness of our contributions was passed through intensive simulations. Simulation of P2P Grid allows the creation of a controlled and a simplified view of the environment, allows a better observation of the behaviour of the different components and their interactions. In addition to, it permits a deep analysis of the results by reproduction of the same experiment or getting new ones by a slight changing of the environment parameters.

We have created a fully object oriented discrete-event simulator to implement the different proposed algorithms. Its first component is a network topology generation that is responsible for the creation of the three layered overlay architecture, federations and their interconnections with the other federations and ordinary nodes. Besides, it creates nodes with their resources and basic functionalities. Its second component is P2P Grid RM elements which encapsulates all the studied processes of the RM.

Figure 1.5 presents an overall view of the simulator for P2P Grid RM.

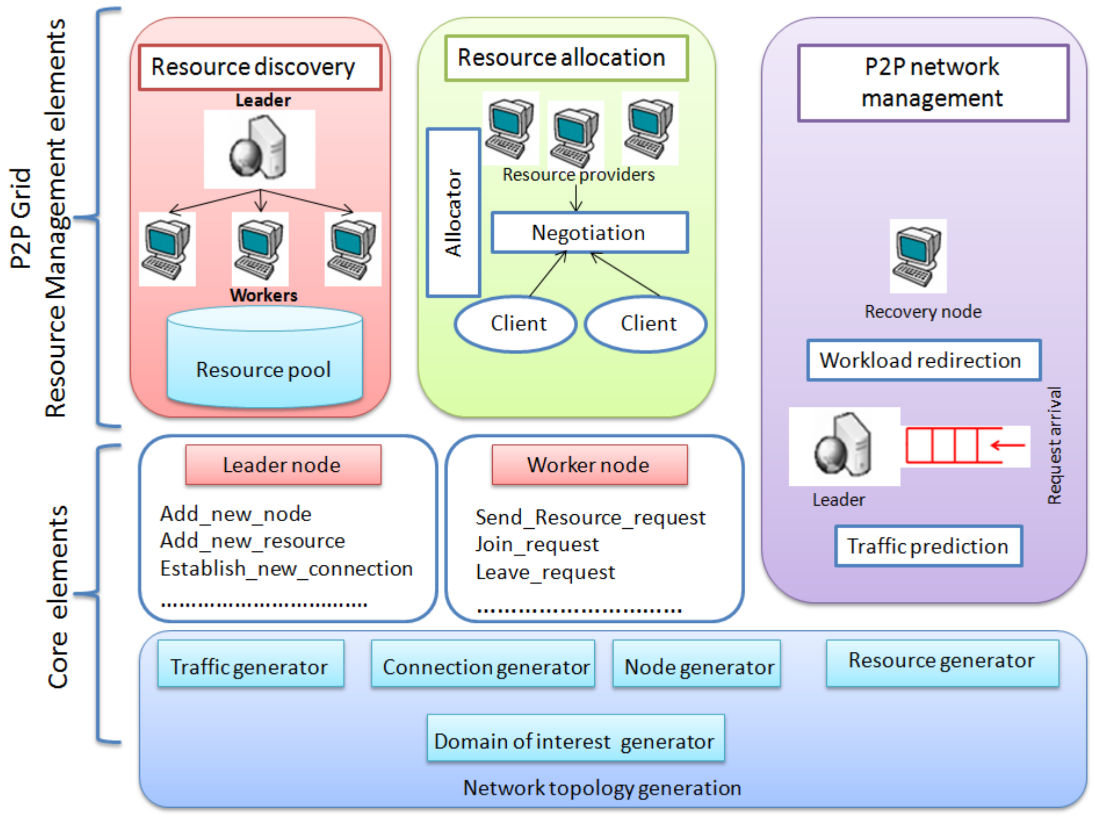


FIGURE 1.5 – Global view of the P2P Grids simulator for resources management.

This simulator has been used to validate all the proposed approaches. Its parameters have been adjusted according to the simulation process requirements. With each contribution, a descriptive table of the used parameters and the corresponding processes will be provided.

1.8 Dissertation Overview

The rest of this dissertation is structured as follows.
Chapter 2 presents a brief summary of the theoretical foundations, methods and technologies used in the next Chapters.

Chapter 3 provides the literature review of resource management techniques in P2P Grid Computing. It first discusses a state of the art of resource discovery techniques, then it introduces ultra-peer overlay networks constructions approaches and discusses them from the scalability aspect. The chapter ends by providing an overview on resource allocation in P2P Grid.

Chapter 4 describes our overall system architecture and routing algorithm for P2P Grid resource discovery. It displays the layered architecture and the overlay network construction that is based on semantic clustering of nodes according to their domains of interests. The semantic clustering utilises the SKOS lightweight ontology which describes the fields of applications in P2P Grid computing, so that to affect nodes to their adequate groups to organise them into a hierarchical structure. Then the routing algorithm exploits the structural relations of the overlay to route query between different groups in an efficient and scalable way which provide a guaranteed look-up. Extensive simulations are used to validate the effectiveness of the proposed approach.

Chapter 5 introduces our technique to improve the scalability of ultra-peers networks by predicting and analysing the workload that are exercised on each ultra-peer node. The chapter highlights the problematic of scalability and bottleneck and gives a preview on the studied system and its characteristics, and then presents the proposed approaches. Finally, it performs a large scale experiment to show the ability of our system to scale gracefully as more nodes and their consequent requests are injected.

Chapter 6 presents the distributed resource allocation based on multilateral bargaining. It presents the system components and their interactions, then it models the resource allocation problem of the system component by utility function. After that, the chapter describes algorithms for optimising the system elements utilities based on a bargaining mechanism. At last, preliminary results are presented to validate the proposed approach.

Finally Chapter 7 presents conclusions, summarises works and results obtained from this dissertation, and points out possible directions for future work.

1.9 Published Works

The work of this dissertation is supported by the following journals and conferences publications:

1. N. Chergui, S. Chikhi and M.T. Kechadi: Semantic Grid Resource Discovery based on SKOS Ontology. Accepted for publication in 2015. To appear in the International Journal of Grid and Utility Computing. Inderscience publisher.
2. N. Chergui, M.T. Kechadi and S. Chikhi: Scalability-Aware Mechanism based on Workload Prediction in Ultra-Peer Networks. Journal of Peer-to-Peer Networking and Applications. Springer. DOI 10.1007/s12083-017-0542-z. 2017.
3. N. Chergui, S. Chikhi and M.T. Kechadi: An Architecture based SKOS Ontology for, Scalable, Efficient and Fault Tolerant Grid Resource Discovery. Third IEEE International Conference on Next Generation Networks and Services. NGNS'11 Hamammet Tunisia, December 18-20-2011. pp 78-83.
4. N. Chergui and S. Chikhi: A Scalable Hybrid Architecture based SKOS Ontology for Resource Discovery in Grid. International Conference on Information and Communication Systems ICICS, Irbid Jordan. May 22-24-2011. pp 113-118.
5. N. Chergui and S. Chikhi: Toward an Efficient and Scalable Architecture Based on SKOS Ontology for Resource Discovery in Grid, Guelma, Algeria. 8ieme Colloque International sur l'Optimisation et les Systemes d'Information COSI'11. April 24 -27. pp 229-240.

Chapter 2

Preliminaries

This chapter presents a brief summary of the theoretical foundations on which the contributions of this dissertation are built upon.

2.1 Semantic Web and Knowledge Organisation and Representation

The amount and the variety of information in every knowledge domain which are continually increasing, the demand and the need of relevant information which capture the real world diversity, represent serious challenges face to processes of information retrieval, and knowledge representation. Hence, Semantic Web and knowledge modelling, organisation and representation play an important role because they are able to offer methods and tools to model, to organise and represent knowledge in a context sensitive to capture the diversity of information.

"*The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation.*" [Ber+01]

This means, Semantic Web represents information more meaningfully for humans and computers alike by enabling the description of contents and services in machine-readable form. It allows the annotating, discovering and publishing services to be automated. It thereby facilitates the interoperability and the sharing of knowledge over the Web. Therefore, its main goal is to make information on the Web accessible and understandable by humans and computers.

On the other hand, Knowledge Organisation System *KOS* is a term that summarises knowledge structures such as controlled vocabularies, taxonomies, thesauri and ontologies for organising information and information retrieval. It provides the basis for describing knowledge domains [Hod00].

"*The term Knowledge Organization Systems is intended to encompass all types of schemes for organising information and promoting knowledge management. Knowledge Organization Systems include classification and categorisation schemes that organise materials at a general level, subject headings that provide more detailed access, and authority files that control variant versions of key information such as geographic names and personal names. Knowledge Organization Systems* *also include highly structured vocabularies, such as thesauri, and less traditional schemes, such as semantic networks and ontologies.*" [Hod00].

2.1.1 Semantic Web Knowledge Representation and Languages

In Semantic Web, knowledge can be represented and organised in several forms ranging from simple term lists to complex ontologies, where the knowledge representation languages are usually dependent on the knowledge representation form. Among all forms of knowledge representation, we are interested in ontologies.

The term *ontology* is borrowed from philosophy where ontology means a systematic account of existence. It has been widely exploited by the artificial intelligence and knowledge representation community before becoming part of the standard terminology of a much wider community indicating information systems modelling [Gua+09]. In artificial intelligence, according to Tom Gruber [Gru93], an ontology is:
" *"The specification of conceptualisations, used to help programs and humans share knowledge".*"

In this usage, an ontology is a set of concepts such as things, events and relation that are specified in some way in order to create an agreed-upon vocabulary for exchanging information.
There are many definitions of ontologies among other we cite those of:

- Gruber [Gru93]: originally defined the notion of an ontology as an: *"explicit specification of a conceptualisation".*
- Borst [Bor97]: defined an ontology as a: *"formal specification of a shared conceptualisation".*
- Studer [Stu+98]: merged the previous two definitions to state that: *"an ontology is a formal, explicit specification of a shared conceptualisation".*

A brief examination of these definitions is [Stu+98; Gru93]:
- Explicit means that the concepts used must be explicitly defined.
- Formal means that the ontology should be machine readable interpretable.
- Shared means that the knowledge represented in an ontology (concepts) must be agreed and not just a small group's views.
- Conceptualisation means an abstract, simplified model of the world that we wish to represent, it consists of the relevant concepts and the relationships that exist in a certain situation.

Although, other two interesting definitions are given by:

- Neches [Nec+91]: *"An ontology defines the basic terms and relations comprising the vocabulary of a topic area, as well as the rules for combining terms and relations to define extensions to the vocabulary".*
- Swartout [Swa+96]: *"An ontology is a hierarchically structured set of terms for describing a domain that can be used as a skeletal foundation for a knowledge base".*

Like their definitions, many kinds of ontologies are existing, usually classified depending on the type of concepts, constraints and how expressive an ontology is. With respect to the expressiveness, there are two types of ontologies:

[Miz03]:

— Lightweight ontologies: are less expressive; usually are taxonomies which consist of a set of concepts and hierarchical relationships among the concepts;
— Heavyweight ontologies: are high expressive; use axioms which allow not only obtaining a richer semantic model but also inferring new knowledge from it.[Gom+04]

The choice of the type of knowledge representation (in this case, ontology type) to be used is essentially dependent on the requirements of a particular application. Therefore, the choice of the type of language to be used is as well dependent on the knowledge representation type and its purposes.

Among others, OWL[1] [Sah07] and SKOS[2] are two of the Semantic Web knowledge representation languages used to represent domain knowledge at different levels and for different purposes due to different problem requirements.

OWL is well known for its capability of expressing a rich axiomatic logic based ontologies. This is due to its precise semantics that allows explicit modelling and description of a domain and enables automated reasoning.

Where SKOS is designed to represent knowledge organisation systems whose representation has weak semantics, that is used for information retrieval and navigation tasks.

In this thesis, we are interested in less expressive knowledge representation (lightweight ontology), as a consequence with SKOS.

2.1.2 SKOS: Simple Knowledge Organisation System

This part is mainly excerpted from the web site[3]

SKOS has been accepted as W3C Recommendation in August 2009. It provides a standard way to represent knowledge organisation systems using the Resource Description Framework *RDF*[4]. Encoding this information in RDF allows it to be passed between computer applications in an inter-operable way.

The basic element in SKOS is a concept that can be viewed as a unit of thought; ideas, meanings or objects that are subjective and independent of the term used to label them. Each concept can be linked to one or more lexical labels to refer to them in natural language through *prefLabel*, *altLabel* or *hiddenLabel*. The terms are also semantically linked to each other through hierarchical broader/narrower relations and associative related relations. SKOS introduces the

class *skos* : *Concept* that allows implementors to assert that a given resource is a concept.

1. http://www.w3.org/TR/owl-features/
2. https://www.w3.org/TR/skos-primer/
3. https://www.w3.org/TR/skos-primer/
4. https://www.w3.org/RDF/

The key elements of SKOS are summarised as following:

— Concept: A SKOS concept is the fundamental SKOS vocabulary element that can be viewed as an idea or notion – a unit of thought. It is used to define an atomic conceptual resource. It is defined as an instance of class skos:Concept;

— Concept Schemes: A SKOS concept scheme can be viewed as a collection of one or more SKOS concepts and links (semantic relationships) between the concepts. $skos : ConceptScheme$ is defined as an instance of $owl : Class$. There are three properties associated with concept schemes, namely $skos : inScheme$, $skos : hasTopConcept$ and $skos : topConceptOf$, defined as instances of $owl : ObjectProperty$. The property $skos : inScheme$ is defined to be a super-property of $skos : topConceptOf$. It is used to indicate that a particular SKOS concept is belongs to the specified concept scheme;

— Lexical Labels: SKOS concepts need expressions to refer to them in natural language: labels. SKOS defines three properties of lexical labels that can be associated to each SKOS concept; namely preferred labels $skos : prefLabel$, alternative labels $skos : altLabel$ and hidden labels $skos : hiddenLabel$. These properties are formally defined as being pairwise disjoint. This means, for example, that it is an error if a concept has the same literal both as its preferred label and as an alternative label. The preferred label property refers to the ordinarily used label in natural language for a SKOS concept. Only one preferred label per language tag is allowed for a SKOS concept. The $skos : altLabel$ property makes it possible to assign an alternative lexical label to a concept. This is especially helpful when synonyms need to be represented, in addition to near-synonyms, abbreviations and acronyms which can be represented in the same way. The $skos : hiddeLabel$ property is only visible to the search engine, and not visible to the user. It is usually used to represent misspelt variants of other lexical labels;

— Semantic Relations: SKOS supplies three standard properties to represent a semantic relation between concepts. These properties are classified into two types: hierarchical and associative. The hierarchical relation is used to represent hierarchical links between two concepts in which one concept is more general than the other through $skos : broader$, or that one concept is more specific than other through $skos : narrower$. An associative relation is used to represent links between two concepts in which the two are related, but not in a hierarchical manner. For this type of relation, SKOS provides a property called $skos : related$ to assert associative links between concepts where neither one is more general or more specific.

2.2 Neural Network

An Artificial Neural Network ANN is a mathematical model that tries to simulate the structure and functionalities of biological neural networks. Basic building block of every artificial neural network is an artificial neuron, that is, a simple mathematical model (function). The motivation for the development of neural network technology stemmed from the desire to develop an artificial system that could perform "intelligent" tasks similar to those performed by the human brain. Neural networks resemble the human brain in the following two ways:

— Knowledge is acquired by the network through a learning process;

— Inter-neuron connection strengths known as synaptic weights are used to store the knowledge [Hay94].

A neural network is an intelligent data-driven modelling tool that is able to capture and represent complex and non-linear input/output relationships. It has, also, the ability to learn these relationships directly from the data being modelled and capable of approximating unknown functions and processes. Thus, it is an attractive and good candidate for use in prediction, as well as for many other applications such as function approximation, pattern recognition and classification, memory recall, optimisation, etc.

Artificial Neural Networks $ANNs$ have several types, each one has been developed for specific purposes. Time-Delay Neural Network ($TDNN$) [Vri+90] uses the context as part of the upcoming inputs for modelling time-series data; it stores the previous inputs as part of the next inputs. The context can be presented as a time series of input buffers, each buffer represents a time step; the information is then shifted from one buffer to another every time step. However, TDNN has a difficulty of dealing with sequences of variable length, due to its fixed input window [Bod01]. Recurrent Neural Networks (RNN) are a class of neural network, which augmented with one or more additional context layers storing output values of one of the layers delayed by one step and used for activating other layers in the next time step. But, they suffer from heavy computing overheads [Hua+05].

2.3 Queueing Theory

Queueing theory was developed by A.K. Erlang in 1904 to help determine the capacity requirements of the Danish telephone system. It has since been applied to a large range of service industries including banks, airlines, and telephone call centres,etc.The Classical queueing theory has been a useful tool on modelling systems where jobs arrive randomly at static service stations of given service capacities. It provides analysis of the system's properties such as stability and sojourn time. It is mainly seen as a branch of applied probability theory and it has wide application in many scenarios of operations research. In particular, its application in studying computer networks and operating systems led to a generalisation of queueing theory to model a network of queues and many different service policies. [Kle75; Bas+75]

Queueing network models are also applicable to the performance analysis of steady state system and have been popular mathematical tools to model many real-world computer networks. For a P2P network, the station/node in the queueing network could be a peer or a phase of peer behaviour by abstraction. Hence, queueing networks are quite flexible to design.

2.3.1 Classification of Queueing Models using Kendall Notation

Kendall's notation [Ken53] signifies the standard system used to describe and classify the queueing model that a queueing system corresponds to. The notation was first suggested by David George Kendall in 1953.

The simplest form of Kendall's notation consists of three letters $a/b/c$, where the more comprehensive one consists of 6 letters $a/b/c/d/e/f$ where:

— a: represents the probability distribution of customers arrival;
— b: represents the probability distribution of service time;
— c: means the number of servers;
— d: signifies the capacity of the queueing system, usually infinite;
— e: indicates the size of population;
— f: describes the queueing discipline.

The arrival and service processes (a/b) have the following abbreviations:

— M: Poisson process;
— D: Deterministic process;
— G: General process.

2.3.2 Markov Chain Model

In this part, which is mainly excerpted from [Con11] , we introduce a brief explanation of Markov chain process.

Let T be a set, and $t \in T$ a time parameter. Let $X(t)$ be a random variable $\forall t \in T$. Then the set of random variables $\{X(t); t \in T\}$ is called a stochastic process.

We usually interpret $X(t)$ to be the state of the stochastic process at time t. If T is countable, for example, if we let $t = 0; 1; 2; ...$, then we say that $\{X(t); t \in T\}$ is said to be a discrete-time process.

A stochastic process $\{X(n); n \in N\}$ is called a Markov chain if [Ser09], for all times $n \in N$ and for all states $(i_0; i_1; ...i_n)$

$$P\{X_n = i_n | X_0 = i_0; ...; X_{n-1} = i_{n-1}\} = P\{X_n = i_n | X_{n+1} = i_{n+1}\}$$

This equation called the Markov property.

Any stochastic process satisfying the Markov property will be a Markov chain, whether it is a discrete-time, or continuous-time process.

The process starts in one of the states $(i_0; i_1; ...i_n)$ and moves successively from one state to another. Each move is called a step. If the chain is currently in state i_{j-1}, then it moves to state s_j at the next step with a probability denoted by $p_{ij}(n)$ 2.1, called by the transition probability. This probability does not depend upon which states the chain was in before the current state [Gri+09].

In other words, given the present state of the system, we may make predictions about its future state without consulting past states.

We call the conditional probability

$$P\{X_n = j | X_{n-1} = i\}; i, j \in S \tag{2.1}$$

A Markov chain is called time-homogeneous if $p_{ij}(n)$ does not depend on n. In other words:

$$P\{X_n = j | X_{n-1} = i\} = P\{X_{n+m} = j | X_{n+m-1} = i\}$$

For $m \in N$ and $m \geq -(n-1)$.

The construction of the transition matrix P for the Markov chain passes through the use of the transition probabilities.

P is an $N * N$ matrix where the $(i; j)$ entry P_{ij} is p_{ij} . In order for a matrix to be the transition matrix for a Markov chain, it must be a stochastic matrix. In other words, it must satisfy the following two properties:

$$0 \leq \mathrm{P_{ij}} \leq 1; 1 \leq \mathrm{i}; \mathrm{j} \leq \mathrm{N}$$
$$\sum_{\mathrm{j}=1}^{\mathrm{N}} \mathrm{P_{ij}} = 1, 1 \leq \mathrm{i} \leq \mathrm{N}$$

Given a transition matrix P, an initial probability distribution d which specifies the starting state. Usually this is done by specifying a particular state as the starting state. Where $d(i) = P\{X_0 = i\}; i = 1; ...; N$,

We can find the probabilities that the Markov chain will be in a certain state i at a given time n. We define the n-step probabilities p_{ij}^n as the following:

$$p_{ij}^n = P\{X_n = j | X_0 = i\} = P\{X_{n+k} = j | X_k = i\}$$

The latter part of the equation follows from time-homogeneity. Then we have:

$$P\{X_n = j\} = \sum_{i \in S} d(i) p_n(i, j) = \sum_{i \in S} d(i) P\{X_n = j | X_0 = i\}$$

where S is the state space.

2.4 Optimisation Theory and Dynamic Programming

Optimisation is the act of obtaining the best result under given circumstances [Rao09], it has become ubiquitous in the modern world, it is applied to many real-world problems such as economics, engineering, communications, logistics, Internet commerce, transportation, network management, supply chain management, energy efficiency, and many others. It is central to any problem requiring decision making [Cho+01]. The decision making entails choosing between various choices, where the choice is governed by the desire to make the best decision. In other words, the ultimate goal of the decision making is either to minimise the cost required or to maximise the desired benefit. The cost required or the benefit desired can be expressed as a function of certain decision variables, called an objective function. Thus, optimisation can be defined as the process of finding the conditions that give the maximum or minimum value of the objective function [Rao09].

Since the optimisation covers different real-world problems, the optimisation problems have different types: they can be simple or constrained, static or dynamic, deterministic or stochastic, and can have a different level of complexity. Therefore, there is no single available method for solving optimisation problems efficiently. Hence a number of optimisation methods have been developed for solving the different types of optimisation problems [Rao09].

Methods to solve optimisation problems are known as mathematical programming techniques and are generally studied as a part of operation research. The existence of these methods can be traced to the days of Newton, Lagrange, and Cauchy.

Dynamic Programming is a powerful technique that allows solving many different types of optimisation problems in a polynomial time. The term Dynamic Programming DP was originally used in the 1940s by Richard Bellman to describe the process of solving problems where one needs to find the best decisions one after another. DP is a technique for solving a large complex problem whose solution satisfies recurrence relations with overlapping sub-problems [Hri+05], the idea is to break the problem down into a sequence of simpler sub-problems, solving each of those sub-problems only once and storing the results rather than solving overlapping sub-problems over and over again. DP algorithm will examine the previously solved sub-problems and will combine their solutions to give the best solution for the given problem.

Algorithm 1 Knapsack Dynamic Programming

```
1: {values are stored in Array $v$ and weights are stored in Array $w$)}
2: {$m$ is an array to record results}
3: for $j = 0$ To $n$ do
4:     $m[0, j] \leftarrow 0$
5: end for
6: for $i = 1$ To $n$ do
7:     for $j = 0$ To $C$ do
8:         if $w[i] \leq j$ then
9:             $m[i, j] \leftarrow max(m[i-1, j], m[i-1, j-w[i]] + v[i])$
10:        else
11:            $m[i, j] \leftarrow m[i-1, j]$
12:        end if
13:    end for
14: end for
```

The following example illustrates a typical complex problem, the knapsack problem, solved by the DP.(see the above algorithm 1).

The Knapsack problem can be described as follows: We have an empty bag with (weight) capacity C, along with n items. The items have weights $w(i)$ and values $v(i)$, for $i = 1, 2, ..., n$. Find the subset of items which can be carried into the knapsack with weight limit C. It is required that the cumulative value of the items in the knapsack is maximum value possible.

2.5 Summary

This chapter presented a brief summary of the methods, tools, and technologies used to achieve the objectives of this thesis. It is considered as a useful basis for monitoring the contributions that will be presented in the succeeding chapters.

Chapter 3

Literature Review

P2P Grid is heterogeneous, dynamic and a distributed environment in which Resource Management (RM) system is the central part. The RM is composed of two principal processes, resource discovery and resource allocation. To well manage this environment, the RM must meet several requirements among others, it has to be efficient, scalable and adaptable.

In this chapter, we will mainly limit ourselves by reviewing the following axes:

— Related work on resource discovery;
— Techniques and approaches for construction of ultra-peer overlay networks used to build P2P Grids;
— Related work on workload analysis and prediction;
— Related work on economics-based resource allocation.

In addition, we close the study by providing a discussion on key differences between the presented works and our proposed ones.

3.1 Resources Discovery

One of the fundamental requirements of P2P Grid computing is an efficient resource discovery mechanism. Resource discovery is a prominent process as it is responsible for finding suitable resources that match the user requests. The centre of interest of almost every solution of resource discovery is the efficiency, the scalability and the fault tolerance.

The conception of resource discovery approaches is widely influenced by the underlying P2P network choice between different architectures and structures. Figure 3.1 depicts a taxonomy of resource discovery process, which can be varied between the type of architecture and the used search technique. In this thesis, we will pay attention to another factor which is the use of Semantic Web (SW) technologies or not, because, SW can be used with either the architecture for clustering or with the search techniques (keyword-based or semantic-based).

In this section, we present some important services and resource discovery mechanisms, and then discuss the efficiency and the scalability of each work.

3.1.1 Centralised Approaches

In the centralised approaches, information about the provided resources are stored in one or several servers. A node wishing to find a certain resource has to

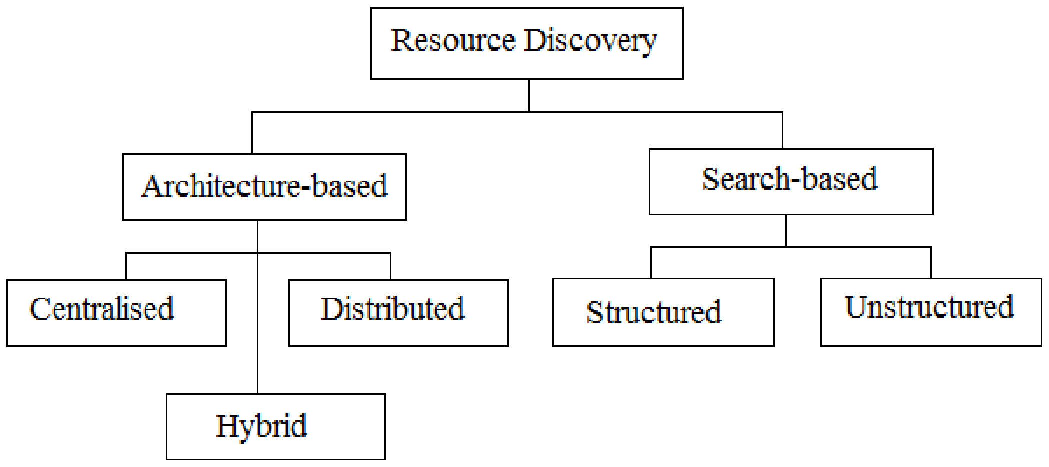


FIGURE 3.1 – Resource discovery taxonomy.

send a request to the server which has to find the appropriate resources according to the request.

- XtremWeb [Fed+01] is a system influenced by user demands, towards a computational peer-to-peer system, XtremWeb, as multi-users, multi-applications, in its current architecture is closer to the conventional centralised and unstructured desktop Grid computing, designed [Zha+11] for research and production, aiming at executing external applications on participant resources. In XtremWeb nodes are functionally classified into three categories: clients, workers and coordinators. A set of clients submits task requests to the system coordinator, which will execute them on workers. The coordinator is responsible, among other services for receiving task requests coming from several clients, distributing them to workers according to a scheduling policy, transferring application code to workers if needed, supervising task execution on workers and delivering task results to the client.

- Entropia [Chi+04; Chi+03] implements a node manager in which the clients register their resources, to present them later to a sub-job resource scheduler, which in its turn, used them once needed to match sub-jobs to available client machines according to machine's attributes such as memory capacity, operating system type. The node manager provides a centralised interface to manage all of the clients on the Entropia desktop Grid, which is accessible from anywhere on the enterprise network. It tracks the status of all of the connected machines, includes the connectivity of the client, the amount of the performed work from the client. The interface is designed to provide scalable management to vast numbers of clients, requiring minimal effort per client added to the Grid.

- BOINC [And+06; And04], the Berkeley Open Infrastructure for Network Computing, uses the idle cycles from volunteered computing resources. The BOINC system employs a centralised server to deploy a larger number of independent jobs across available client machines on the Internet. Therefore, it does not provide the matchmaking functionality, a BOINC server simply deploys jobs across client machines and the Grid scheduler takes the responsibility of assigning the tasks to the volunteers based on their availability. The BOINC architecture is highly modular and scalable. If the server becomes inundated with client requests, additional servers can be added to the project, each handling only a fraction of the total incoming requests.

- The QADPZ [Con08; QAD] system allows the management and use of the computational power of idle computers in a network. The users of the system can send computing tasks to these computers to be executed.The system adopts a client-master-slave architecture and consists of three components, one master (the central component of the whole system), many slaves and multiple clients, and it implements three processes, master, slave and client. Messages between the components of the system are in XML format and can optionally be encrypted for security reasons. The master computer is a most likely a dedicated computer with some UNIX/Linux, in which the master process run on, it is responsible for jobs-tasks-slaves accounting, and for maintaining the current availability status of the slaves, and for starting and controlling the tasks. A slave computer is one of many computers where the distributed and collaborative computation takes place, examples: a UNIX server, workstation, or any personal computer, where the slave process which runs on the slave computer as WinNT-service, communicates with master when it joins or leaves the system, or receives, starts or finishes slave user process. Without slave running, the slave computer cannot take part in a collaborative computing. The client computer is any computer that the user uses to start his user application. It can be a notebook connected to the network using a dial-up connection, a computer in the office, lab, etc. The client process running on a client computer; communicates with the master to start and control jobs and tasks of a specific user. It is also responsible for scheduling the tasks of user jobs as required by a particular user application (beyond the scope of scheduling done at master). A client does not communicate with the slaves directly, instead, it sends all its requests to the master.

- Condor [Tan+02; Ram+98], is historically the pioneered in using the idle time of distributed owned workstations to do sequential and parallel computing. It provides a powerful and flexible resource management services, including a job queueing mechanism, scheduling policy, priority scheme, resource monitoring. Workstations are dynamically placed in one of the multiple resource pools whenever they become idle and get removed from the resource pool when they become busy. Condor uses a matchmaker with a central server to process queries where the matchmaker collects information about the state of resources. It receives the users' job requests, then it matches the resource description and usage policies specified by the resource owners with the jobs' needs. After that, it places them into a queue, and decides where and when to schedule them, carefully monitors their progress, and ultimately informs the user upon completion.

These types of mechanisms are fast, efficient and provide the facility to build and to access the Grid services. However, their main drawbacks are their inability to scale well and the bottleneck problems that can occur in servers when frequent updates and large numbers of requests exist, in addition to the single point of failure.

Another kind of mechanism is the hierarchical and tree-based resource discovery, in which servers are organised in a hierarchical fashion. Hence queries are processed hierarchically, and each server will be a responsible for partitions of resource information [Nav+14].

- Chang et al. [Cha+10] propose a method that uses tree mechanism in resource discovery. It transforms resources' attributes and queries to bitmap representations. Each leaf node in the tree will store the information about its local resources in a local resource bitmap, and the index bitmap registers information about its children nodes and information about its local resources as well. Requests are forwarded to indexing servers to find whether there is a matched resource. If it can find matched resources in a node, the request will be forwarded to children nodes. Otherwise, the search will forward up the tree until it reaches the root. This method scales better and reduces the traffic and the cost of updates; however, it suffers from extensibility and a single point of failure.

In general, hierarchical approaches are more scalable than the centralised ones [Nav+14], they reduce the network traffic and can guarantee low search delay due to the use of multi-layer architecture, but in cases where the query is not answered at the lower level, the response time may become more interesting. In addition, single point of failure at the tree root decreases the system fault tolerating feature.

3.1.2 Distributed approaches

Practical approaches towards more scalable solutions are offered by P2P models and decentralised approaches. Gnutella is constructed to be decentralised and followed the classical concept of an unstructured P2P system. The resource discovery process is performed by a flooding strategy, where a query is propagated to all neighbours within a certain number of hops, which makes it efficient and guarantees to find an available resource if there exists one. Therefore, the number of queries increases exponentially and causing huge overhead, and when the network grows, it gets saturated and often caused enormous delays.

- Iamnitchi et al. [Iam+01] propose a fully decentralised P2P approach, it organises nodes' information into a flat unstructured P2P network and random walk based methods are used for query forwarding. The approach suffers from higher numbers of required hops to resolve a query and provides no look-up guarantees.

- Other variations of P2P techniques that exploit a DHT to organise a structured network have been proposed. Among others, Talia et al. [Tal+06] propose a DHT-based resource discovery mechanism for large-scale Grids based on Chord and multiple DHTs, though the cost of keeping multiple structured DHTs is very high.

- Pipan [Pip10] presents a decentralised solution using a TRIPOD overlay network for resource discovery, based on a hybrid overlay network. The approach used a K-Tree structure which allows the implementation of an efficient proximity query algorithm, based on the Bloom filters for the purpose of routing network messages to reduce the network's traffic. The implementation of the algorithm is based on the iterative deepening and divide and conquer approach. The first allows for the iterative expansion of the search parameter, as the algorithm first identifies the closest suitable resources and proceeds only if more resources than those identified are required. The second approach improves the response time of the query, as the query is transmitted to several branches at the same time, thus greatly reducing the expected execution time of a query.

- A decentralised learning automata is proposed by Torkestani [Tor12] for large-scale P2P Grid resource discovery. The proposed method forwards the resource queries through the shortest path ending at the Grid peers which probably have the requested resource, where each peer is equipped with a learning automaton and network of learning automata is responsible for routing the query toward the resource provider through the shortest path. This algorithm was designed to relieve the negative impacts of the global flooding problem on the network performance.

- Ferretti [Fer13] uses gossip protocol with local information on the network to build a resource discovery mechanism on top of unstructured P2P overlay networks. Peers maintain local knowledge about their neighbours where each node is aware of its resources and those of nodes which are directly linked to it. The discovery process passes through the links that compose the overlay. A node generating a query can ask its neighbours only. In turn, each time a node receives a query, it relays the related message to those neighbours holding resource items matching the query, and it gossips the message to other neighbours, so the query can be disseminated through the overlay. The network topology can be set by defining the node degree distribution probability. Depending on the network topology, the resource availability and the gossip probability, it is possible to understand if the query reaches a limited amount of nodes, or it might reach an infinite amount of nodes.

3.1.3 Hybrid Approaches

Super-peers (ultra-peers) approaches have been originally proposed to reach a balance between the efficiency of centralised search, and the autonomy, load balancing and fault-tolerance offered by distributed search [Yan+03].

- Gnutella v.0.6 uses ultra-peers architecture. It has been introduced to reduce the scalability problems in Gnutella v.0.4 caused by the flooding search. It classifies nodes into leaf nodes and ultra-peer. A leaf node connects to one ultra-peer, and an ultra-peer connects to both its own leaf nodes and other ultra-peers. Leaf nodes initiate service requests, receive associated responses, and respond to the requests that they can exactly answer. Ultra-peers insure the same functionality of leaf nodes, in addition, they handle routing, storing and forwarding information on behalf of leaf nodes [Yan+03]. The flooding-based mechanism is used for query routing and look-up requests among ultra-peers.

- Kazaa divides users into two groups, super-nodes and ordinary nodes. Super-nodes process data requests from the slower ordinary nodes. Kazaa uses a Time-To-Live (TTL) for query routing, usually seven. The super-node communicates with other super-nodes, which in turn connect to regular nodes that in turn connect to even more regular nodes, to fulfil the request until the Time to Live runs out. Once the correct file has been located, it is transferred directly from the file owner to the requester using HTTP without passing through the super-node.

In addition, Kazaa dynamically elects super-peers that are more powerful to form an unstructured overlay network, and a regular node can connect to one or more super-peers to query the network resources [Lia+04].

- Mastroianni et al. [Mas+05] propose a mechanism in which some nodes are selected as super-peers, which act as directory services. Each resource node is connected to one super-peer to share its resource information. The queries are accepted only by the super-peers. When a query is submitted, it is routed to the nearest super-peer first and if the query cannot be satisfied by a super-peer, it is flooded to other super-peers in the network. When a super-peer realises that the query is satisfied by a node within its group, it returns the address of the available resource to the requester. This algorithm presents more scalability; but, the time and message complexities are still considerably high due to the use of flooding.

- Ali et al. [ALI+12] propose a framework based on hypercube computational Grid, it is constructed of two layers called Hypercube Service Node (HyperSN) which is connected using ring topology and Circle HyperSN which is a set of HyperSN. It provides a preserving locality protocol based on a distance metric for the HyperSN overlay construction. This framework preserves administrative control and autonomy because routing between various administrative organisations is permitted according to the policies defined by the target organisation. This proposal is scalable in terms of time because it keeps the maximum number of steps required to resolve range queries. However, it uses flooding in node joining phase; therefore, it suffers from high traffic.

- Padmanabhan et al. [Pad+10] present a self-organised grouping (SOG) framework that forms and maintains autonomous resource groups in a leader/worker fashion to enable discovery of dynamic Grid resources and to handle multi-attribute range query. In this framework, each group dynamically aggregates a set of resources together with respect to similarity metrics of statistical resource characteristics. It exploits the Hilbert Space-Filling Curve's locality preserving and dimension reducing mapping to handle multi-attribute range queries, and the lightweight gossip protocol to construct and maintain SOG overlay network. The major problems of such approaches are the bottleneck on the central clusters, and the complex clustering procedure.

- The OurGrid [Our; Cer+12] is a cooperative distributed P2P Desktop Grid system, where the users are grouped into communities (labs). The OurGrid is based on the idea of aggregating resources existing in universities' labs, in which labs donate their idle computational resources in exchange for accessing other labs' idle resources when needed. The system mainly focuses on compute intensive and independent tasks. Each lab in the Grid corresponds to a peer in the system, and each peer has direct access to a set of resource and contributes to the system by its idle resources. In addition, it is responsible for on-site resource discovery. A peer communicates with other peers via the discovery service in order to request additional resources. In case the workers from its community are not enough to satisfy requests demanded by the OurGrid broker (called MyGrid), the OurGrid discovery service is responsible for connecting multiple OurGrid sites so that several peers can interact and exchange computational resources. The user interacts with its OurGrid community through MyGrid in order to submit and monitor jobs, the MyGrid broker is responsible for providing the user with high-level abstractions for resource and computational task, and a resource matching and heuristic scheduling. In addition, OurGrid disposes of a centralised directory for peer discovery mechanism, CorePeer. Peers register themselves with one CorePeer as they come online. Several CorePeers can exist simultaneously to support multiple independent Grids.

- The Organic Grid [Cha+05; Cha+04] is a bio-inspired structured P2P desktop Grid, that organises heterogeneous and geographically distributed nodes for data-intensive scientific applications. Nodes in Organic Grid are functionally identical and autonomous. The network organisation in Organic Grid is a structured tree overlay, in which nodes are periodically checked and pushed upwards the tree for performance enhancement. Large tasks in Organic Grid are divided into small sub-tasks and each sub-task is encapsulated into a mobile agent, which is then released on the Grid and finds appropriate resources based on autonomous behaviours of each agent. Task and resource scheduling is based on swarm intelligence approaches that each agent roaming around the network performs a simple task without any global knowledge of the system.

- Najafi et al. [Naj+15] present a super-peer overlay for resource discovery in a P2P system, it organises nodes into clusters where ordinary peers can communicate only with their corresponding super-peer. Super-peers can communicate with a set of super-peers neighbours in the overlay network. Each super-peer acts to search and query routing operations, and maintains resource information of its connected peers and other neighbour super-peers by means of routing indices called Hop-Count Routing Index (HRI). If the super-peer cannot process the query locally, it forwards the query to the best selected neighbours based on HRI information with respect to requested requirements. If a resource that matches the specified criteria in the query is found, a query Hit is generated and is returned along the same path back to the querying node. Otherwise, the selection process is repeated and the query is forwarded by neighbour super-peers. This method is continued until to reach a result or all neighbour super-peers are searched.

The preceding models use a keyword based search and do not fully support semantic and multi-attribute range queries. They may suffer from the churn effect and network-wide broadcast storm problem as well as to the false-positive errors.

3.1.4 Semantic Approaches

Semantic approaches are those that use Semantic Web technologies for either overlay network construction or for resource representation and query processing. In approaches that implement semantics and ontologies to define resources, each resource must operate according to its machine-understandable semantics. When a new resource is added to the Grid, its semantics must be specified. De Roure et al. [Rou+05] identify the need for a Semantic Grid, and argue for a Semantic Grid as an extension of the current Grid in which information and services are given well-defined meaning. Corcho et al. [Cor+06] propose a reference architecture that extends the Open Grid Service Architecture (OGSA) to support the explicit handling of semantics. Semantic-OGSA (S-OGSA) defines a model, the capabilities and the mechanisms for the Semantic Grid. It defines the associated knowledge services to support a spectrum of service capabilities. It includes semantic provisioning services and semantically aware Grid services.

On the other hand, semantic clustering or semantic hybrid approaches have appeared with the idea of grouping together nodes with similar contents to facilitate the search.
- Nejdl et al. [Nej+03b] use super-peers to cluster nodes. However, the efficient communication mechanism between super-peers is absent in these systems.

- Iamnitchi et al. [Iam+02] propose to cluster nodes with similar interest in communities, without discussing how to define the interest similarity among peers and how to form clusters.

- Navimipour et al. [Nav+14] add semantic short-cuts to group nodes. The short-cut approach relies on the presence of interest-based locality. Each peer builds a short-cut list of nodes that answered previous queries. To find content, a peer first queries nodes on its short-cut list and only if unsuccessful, floods the query.

- Li [Li10] uses semantic clustering to organise the network topology and reduce the search space to related clusters. it uses a complex and costly mechanism to construct and maintain clusters, each time it needs to add a new node to the system.

- Pirro et al. [Pir+12] combine DHTs and Semantic Web technologies to build a semantic overlay network ERGOT, for service discovery. Category ontology and domain ontology are queried via DHTs, and a semantic overlay network is enabled by the clustering of peers that have semantically similar service descriptions. The query processing is based on semantic and TTL, a peer which receives a request begins by matching it within its local service profiles, then it forwards the request over its semantic links. In order to establish which semantic neighbours the query has to be forwarded, the peer computes the similarity between the query and the semantic links in its semantic table (maintained by each peer), if there are no semantic neighbours that satisfy the criteria, the peer may choose to use the underlying DHT to route the request. This means that similarity estimation in ERGOT is the main pillar, it uses it for service ranking and for query routing queries towards semantically similar neighbours.

- Bianchini et al. [Bia+10] propose a P2P-based semantic service discovery framework that organises peers into a semantic overlay. The semantic overlay can be seen as a continuously evolving conceptual map across collaborative peers that provide similar services, which enables effective similarity-based search service. The system defines strategies for request propagation over the P2P network that keeps the network overhead generated by the discovery process low. It uses a probe collaboration model with a bounded TTL to discover peers with semantically related services.

- Javanmardia et al. [Jav+15] propose a reputation model for trust management in a semantic P2P Grid. They use fuzzy theory in a trust overlay network named FR-TRUST that models the network structure and the storage of reputation information. The network structure is organised onto a semantic overlay using OWL ontology that links several virtual organisations (VO). Each VO is organised according to the super-peer model, where there is a special node called the coordinator that is the super cluster of that VO, and other peers are arranged as a P2P system. They use Chord to carry out distributed queries in the P2P environment.

- Zhang et al. [Zha+15] suggest a scalable, high performance distributed system for data and service discovery. The discovery system adopts one-dimensional vector in semantic space to generally identify and locate data and services, and then represent specific services by OWL. Moreover, they exploit Juxtapose (JXTA) architecture to organise network peers into three layers. The physical network layer, friend group layer and location network layer. The physical network layer is the actual Internet topology. A friend group is a semantic cluster consists of resource providers, who are clustered together as a result of their similar content. All the friend groups constitute a friend group layer, and every group leader, who has more capabilities, constitutes the location network layer. The discovery system provides semantic queries as well as the keyword-based queries to better support search accuracy so that they make use of three types of ontological data in their system, namely; resource domain ontology, Quality of Service (QoS) ontology and service description ontology. Also, they present three algorithms to implement data and service discovery process, including getting semantically related group algorithm, locating resource algorithm and service matching algorithm.

- Di Modica et al. [Mod+11] present a P2P-based infrastructure that leverages semantic technologies to support a scalable and accurate service discovery process. They create an overlay network organised in several semantic groups of peers, each specialised in answering queries pertaining to specific applicative domains. Groups are formed by clustering together peers offering services that are semantically related. When a query is issued, it will be forwarded to the cluster which is more related to it from a semantic point of view, thus maximising the probability to find a service that matches the query. The adopted overlay network is based on a hybrid P2P architecture, where leaf peers are clustered into groups managed by some super-peers: the former delegate to the latter the burden of resource searching and query routing throughout the overlay network to reduce the traffic overhead. The infrastructure is built on top of JXTA.

3.2 Ultra-Peers Overlays Construction in Large-Scale Environments

An overlay network is a virtual network of nodes and logical links that is built on top of an existing physical network, each link corresponds to a path, and may be through many physical links in the underlying network, which is typically the Internet. The main purpose of overlays is to implement a network service that is not available in the existing network [Loo+09].

Ultra-peers networks, (known as super-peers too) are a set of connected ultra-peers, where an ultra-peer is a more stable and powerful peer acts as a server to a subset of other peers called ordinary peers (client, users...), we refer to them also by nodes. Usually, the ultra-peer maintains the information system of the cluster and up-to-date information on all the available resources of its cluster, it may also submit and answer queries on behalf of its peers. Many criteria can be used for selecting ultra-peers [Kle+05] including locality, semantic, capacity, network connectivity, reliability, security, privacy and trust as well as other metrics.

In large-scale systems, ultra-peers networks are effective architectures and can be used to build P2P Grid Computing architectures. The construction could be based on either structured or unstructured P2P. Several approaches have been proposed using different techniques, to address various objectives varying between the enhancement of system performance, efficiency, scalability, etc.

Currently, most Ultra-peers approaches for P2P Grid Computing focus on how to construct their overlay networks, how to select the ultra-peer and manage the system to share and use computational and storage resources, and how to maximise the performance of these resources. Less attention is given to other aspects such as the behaviour of the workload generated at each ultra-peer, which could present bottlenecks and congestion and could limit the scalability; hence, degrade the performance of the whole system. Scalability is an important performance metric in the design of theses overlays. The number of ultra-peers has an important impact on the scalability of the ultra-peer overlay, specifically, when the number of ultra-peers becomes significantly large [Liu14]. The size of the ultra-peer cluster has a crucial impact too. Ultra-peers can experience heavy workload and consequently suffer from the restricted scalability, where the workload on ultra-peer can be originally from ordinary peers maintained by the ultra-peer or from its connections with other ultra-peers.

In this section, we present a literature review concerned with techniques used to construct ultra-peers overlays networks in distributed large-scale computing environments. As well as, we discuss the performance of the proposed approaches from their ability to scale.

- Yu et al. [Yu+05] and Jesi et al. [Jes+06] utilise network proximity and locality to build a super-peer overlay network. Where ordinary peers are connected with super-peers based on their distances such that communication latency is reduced.

- Pyun et al. [Pyu+04] present a protocol designated as $SUPs$ for constructing the super-peer overlay topology of scalable unstructured P2P, based on an approximated method of a random graph is presented. Each super-peer autonomously estimates the number of nodes in the overlay topology N, and maintain the minimum vertex degree $O(logN/loglogN)$; hence, the super-peer overlay possesses a lower network diameter which minimises the inter-super-peers communications and enhances the scalability. But, the message flooding scheme over the proposed super-peer overlay was not discussed in detail.

- Li et al. [Li+10] use the properties of perfect-difference-graph (PDG), to construct unstructured super-peer overlay with a smaller network diameter. The scheme uses the broadcasting mechanism, it aims at reducing the number of look-up messages and to minimise the average delay. It improves the performance of existing hierarchical P2P network in terms of network diameter and the number of flooding messages and achieves a good scalability. However, when the degree of connections of super-peers becomes significantly large, the workload on each super-peer increases greatly, which limits again the scalability.

- Xiao et al. [Xia+05b] present a workload model for establishing the optimal size ratio between the super-layer and the leaf-layer, and propose an efficient dynamic layer management (DLM) scheme for unstructured super-peer architectures. In this later, powerful and stable peers are selected to be super-peers, where a leaf peer can has several connections to other super-peers. The workload is divided into two categories; workload on the overall network, and workload on a super-peer, where each category is divided on three sub workloads: connection workload, query workload, and relay workload. They define the optimal layer size ratio of a super layer to leaf-layer. Then propose the DLM algorithm which can maintain an optimal layer size ratio and continuously elect and adjust peers between the super layer and the leaf-layer. The DLM algorithm inevitably incurs a substantial traffic overhead in exchanging information amongst neighbouring peers and a peer adjustment overhead is incurred when a super-peer is demoted to be a leaf-peer. Moreover, they do not examine which topology is suitable for super-peers to maximise their benefits.

- Teng et al. [Ten+14] propose the utilisation a self-similar square network graph to construct a more scalable unstructured super-peer overlay topology, to deal with the continuous growth of participating peers. Peers are regrouped taking account their locality or proximity. The topology is a constant-degree topology in which each node maintains a constant number of neighbour nodes. Moreover, a simple and efficient message forwarding algorithm is presented to ensure each super-peer to receive just one flooding message.

- Garbacki et al. [Gar+10a] propose a super-peer overlay network based on semantic similarity of peers, where ordinary peers with the same interest are connected to the same super-peers. However, a suitable topology of super-peer overlay networks has not been addressed.

- Other works [Tan+12; Li10; Qia+07; Los+04] build semantic-based super-peer overlays where they clustered together peers that have common preferences. Qiao et al. [Qia+07] describe the contents objects by a taxonomy and present an approach for constructing a semantics-based super-peer overlay, and organising clusters into semantic routing overlays. This work achieved a competitive trade-off between search latencies and overheads and maintained a load balancing among super-peers.

- Liu et al. [Liu+13] present an approach for quickly building a super-peer overlay based on a super-peer selection. Each peer maintains a set of super-peer candidates, and periodically rebuilds its set of super-peer candidates through gossip communication and information dissemination with its neighbours in order to select super-peers and client peers, then it decides whether it takes the role of a super-peer based on its set of super-peer candidates. Only peers that have high capacity (compared to neighbour peers) are selected as super-peers.

- Zols et al. [Zol+09] give a cost-based analysis of hierarchical P2P overlay network with super-peers forming DHT and leaf nodes attached to them. However, super-peers are putting more under stress for both inside and between virtual organisations for resource discovery queries, especially if the leaf nodes number increases. Moreover, performances depend on the ratio between the super-peers number and the total number of peers in the system. Furthermore, most existing hierarchical DHT solutions neglect the churn effect and deal only with the improving performance of the overlay network routing. They mainly generate significant additional overhead to large-scale systems.

- Similarly, Khan et al. [Kha+15] propose a topology of the community network for Cloud, each community network is managed by a super-node, which is responsible for the management of a set of the attached nodes contributing Cloud resources. The super-node acts as a centralised unit to manage the cloud services. The requesting process is based on nodes credit; if the requesting node does not have enough credit the request will be rejected; otherwise, the super-node searches for nodes that have resources available. If the demand cannot be satisfied locally, the super-node forwards the request to the other super-nodes in the federated community cloud. Super-nodes are interconnected in a decentralised manner, and relied on gossip-based discovery mechanisms to manage overlay network of the super-nodes in community Cloud. These remain general solutions that suffer from the local scalability and a bottleneck on each super-node, they do not offer any methods to resolve this problem.

- Yang et al. [Yan+03] evaluate performances and present practical guidelines for the design of efficient super-node networks and underlines their main advantages and drawbacks. Montresor [Mon04] proposes an overall mechanism for the construction and the maintenance of super-peers networks; it employs a gossip paradigm to exchange information with peers and to decide how many and which peers can efficiently act as super-peers. It proposes an algorithm to find the optimal number of super-peers in order to reduce maintenance costs. It is based on the information exchange between super-peers through a gossip protocol.

- Zhuang et al. [Zhu+05] suggest a dynamic layer management algorithm to solve two problems in the super-peer systems. The first one consists of the election of the high capacity peer to be a super-peer, and the second problem corresponds to the maintenance of the architecture to keep the ratio between the super-peers layer and leaf-peers layer as optimal as possible to avoid getting a fully distributed P2P system, or a centralised one. It calculates the ratio between number of super-peers and leaf-peers, and the capacity of each peer compared to the others peers, then it proposes a promotion and demotion policies which allow to each peer to decide to adjust its position in the system, between a leaf-layer and a super-peer layer, according to the calculated ration and capacities, which reflects the changing environments. Since this approach gives to the peer the authority to promote or to demote, so we could find several many peers deciding to promote at the same time, which may make the number of super-peer larger than the number of leaf-peers. In addition, this approach takes into consideration only the spatial dimension, without checking if the actual configuration is optimal from the point of management dimension, so it generates an unnecessary additional cost for maintaining the network.

- Li et al. [Li+08] propose an analytical model based on probabilistic distributions and queueing models to investigate the characteristics of typical file sharing P2P communities' system dynamics. It proposes four operational metrics, content availability, search delay, provision delay, and transmission delay, to evaluate the impacts of network scale on the operational performance of the system. This study shows a positive effect on content availability, transmission delay and provision delay, and a negative scale effect on the provision and search delays. In addition, it investigates the impact of the grouping and the creation of communities, and the interconnection structure between these communities, then it defines a utility function based on content availability and delay costs of the network, to determine the optimal size of the community. This is the closest proposal for our work since it studies the impact and the effect of the scale of the system on the ultra-peers. However, our in work, additionally, we propose solutions to make these systems more scalable.

- Lloret et al. [Llo+14] present a super-node architecture for inter-cloud communication that allows exchanging information, data, services, computing and storage resources between all interconnected Clouds. Each super-node based on its computation capacity estimates the maximum number of allowed connections with other super-nodes and the available load. Based on its estimation, if it does not have sufficient resources and at this time, becomes saturated, then it will not be able to satisfy requests sent by its users, at this time it redirects its load to other interconnected clouds, which gives more scalability to the architecture. This architecture is scalable too because it permits to add new Clouds easily.

- Ardagnaa et al. [Ard+12] propose a resource allocation algorithm and load redirection for Cloud systems. It allows to multiple and distributed resource controllers belonging to several Cloud sites to coordinate. The proposed algorithm estimates the local load on a Cloud site and the redirected workload from the other sites, then it periodically broadcasts the estimated value to the other Cloud sites. If resources of a site are insufficient, which means if the local site has an intra scalability issue, incoming requests may be redirected to other sites. The scalability is guaranteed by switching requests from the heavy loaded site to other less loaded sites.

However, the main drawback of works [Llo+14] and [Ard+12], is that each Cloud will be dependent to the other Clouds, if all the interconnected Clouds are saturated, the whole system will suffer from the scalability problems.

3.3 Workload Analysis and Prediction

In this section, we will discuss related work on traffic and workload analysis and prediction approaches used in different environments and for different purposes.Traffic observation and analysis are important tasks for network resource management, as they are used for planning the capacity and provisioning network resources, performance and security management, to identify heavy sources of the traffic on the network which may cause congestion. They, also, help to measure the temporal variation in the traffic load which assists to an adequate dimension of networks [Mah+10]. In addition, studying and measuring the dynamics of traffic helps to test the stability of networks [Zha+01]. Whereas the workload observation and analysis in Internet applications is crucial since it helps to prevent any bottlenecks on Web applications and to guarantee their scalability; also, it helps in the energy consumption's processes. In this part, we will review the most important fields of traffic measurement and analysis, either in networks or in the field of large-scale distributed computing like P2P Grids and Clouds, in addition to the workload anticipation on Internet applications.

- Lai et al. [Lai+15] study the characteristics of network traffic using the Bayesian classifier, to describe the user's traffic behaviours, to detect the abnormality behaviours in the traffic, and to identify the user responsible for these abnormalities. In addition, it predicts the future behaviour to protect the network.

- Park et al. [Par+13] propose a network traffic analysis model for large volumes of Internet traffic accumulated over a long period of time in a data warehouse, using on-line analytical processing and data mining techniques on a multidimensional data cube, which provided a way to construct a multidimensional traffic analysis system for comprehensive and detailed analysis of traffic data.

In P2P systems:

- Sen et al. [Sen+04] analyse and characterises a P2P traffic observed at a single Internet Service Provider (ISP) network, to examine its impact on the underlying network. It makes use of the network routing prefix for characterising P2P traffic patterns to understand the traffic at this level, which can help for ISP traffic engineering by grouping IP addresses that are topologically close together from a network routing viewpoint, and enables capturing locality characteristics in the P2P system.

- Li et al. [Li+11] propose a traffic prediction-based structured routing algorithm over P2P networks. It builds a wavelet neural network predicting model, which predicts the future state of each peer, such as normal or congestion. Each peer estimates the amount of the next traffic on its used path, and updates its routing table according to the predicted value. Where if it finds that the path will be congested, it notifies its source peer; this later activates a backup process which will transmit the rest of the network traffic along a different path, until the original path finds its normal status. Our work is different in the sense that once the congestion is detected, the process of recovery will be activated, we will not need to redirect load to any of the interconnected ultra-peers nodes which may suffer from the same problem too.

For workload analysis and anticipation:

- Urgaonkar et al. [Urg+08] address the problem of dynamically provisioning capacity to a multi-tier Internet application, it can service its peak workload demand while meeting contracted response-time guarantees. It presents a predictive and reactive approach using queueing theory to capture the behaviour of applications. The predictive approach is to allocate resources to applications on large time scales such as days and hours, while the reactive approach is used for short time scales such as seconds and minutes. The technique assumes knowledge of the resource demands of each tier, it estimates the workload and detects which tier will be a bottleneck and which ones will be able to service all the incoming requests. Then, it determines the number of servers to be allocated to each tier based on the estimated workload.

Aside the difference in the context of our work and this one, the two approaches share some similarities like workload provisioning and estimation. But, our approach is different as it offers a short-term solution by redirecting load to the recovery peer if the congestion remains for few hours, and a long-term solution by splitting the ultra-peer and creating and integrating a new ultra-peer group if the congestion persists.

- Similar to [Urg+08], Iqbal et al. [Iqb+11] propose a methodology and a working prototype that detect automatically the presence of bottlenecks in a two-tier Web application in a Cloud, consisting of a Web server tier and a database tier, to provide the $SLAs$ and to guarantee the users response time requirements for any traffic level, it addresses the scaling of the Web server when the traffic grows. It gathers CPU usage statistics with some heuristics, and uses a regression model to predict the amount of resources tiers required to handle the current workload, and to identify the bottleneck, if the CPU utilisation of any instance in the Web server tier has reached a saturation threshold. Meaning means that bottlenecks are identified, the system scales up the Web server by adding another Web server, or another database server in case of the database tier.

- Other similar efforts to the previously cited works, Ali-Eldin et al. [Ali+12] use queueing theory to model a service in Cloud infrastructures. Two adaptive hybrid reactive/proactive controllers estimate future load in order to support elasticity.

- Arabnejad et al. [Ara+16] present an auto-scaling process that automatically scales the number of resources and maintains an acceptable Quality-of-Service in Clouds. The scaling process can be either vertical or horizontal. Vertically, it modifies the number of resources assigned to each VM (CPU and memory). Horizontally, it acquires or releases of VMs. It uses a technique for dynamic resource allocation, where it does not need to rely on the knowledge provided by the users anymore and can handle various load traffic situations, delivering resources on demand while reducing infrastructure and management costs.

- Messias et al. [Mes+16] propose a method to predict number of requests that arrive at the system in the next period of time to allocate the necessary resources before the system becomes overloaded in Cloud computing environment. It uses a forecasting methodology that uses the genetic algorithm to combine time series based forecasting approaches to calculate the workload prediction, the forecasting model is based on five statistical models. After predicting demand, a queue $M/M/m$ model is used to calculate the amount of resources. The goal is to determine the minimum amount of resources to meet the demand without violating service level agreements.

Although, the target environment of our work is a distributed computing environment built on the ultra-peer overlay network like P2P Grids, composed of non-dedicated computers. In Clouds and through the notion of virtual machines (VM), it is common to add a new instance of servers to handle a heavy workload, it is considered a core practice in these environments even for other perspectives like energy consumption. In the studied environment, ultra-peers may be the most powerful machines comparing to the other ordinary nodes, but they remain non-dedicated ones, and their capacities are much lower than the Cloud's hyper-visors, or even web-servers.

3.4 Economics-based resource allocation

In this section, we will present a literature on economic resource allocation in Grid environments.

This part stands as an interdisciplinary area of economics and computer science, especially between economics and their application for resource allocation. Economic resource allocation can offer more choice and better options for potential Grid users, as stated by Buyya et al. in [Buy+01a]:

"*It offers incentive for resource owners to be part of the Grid and encourages consumers to optimally utilise resources and balance time frame and access costs.*"

Figure 3.2, which is adapted from [Gar+11] and [Pou+06], presents a taxonomy of market-based resource allocation. In this part, we will present resource allocation approaches based on economics (market) mechanisms, with a main focus on the bargaining design.

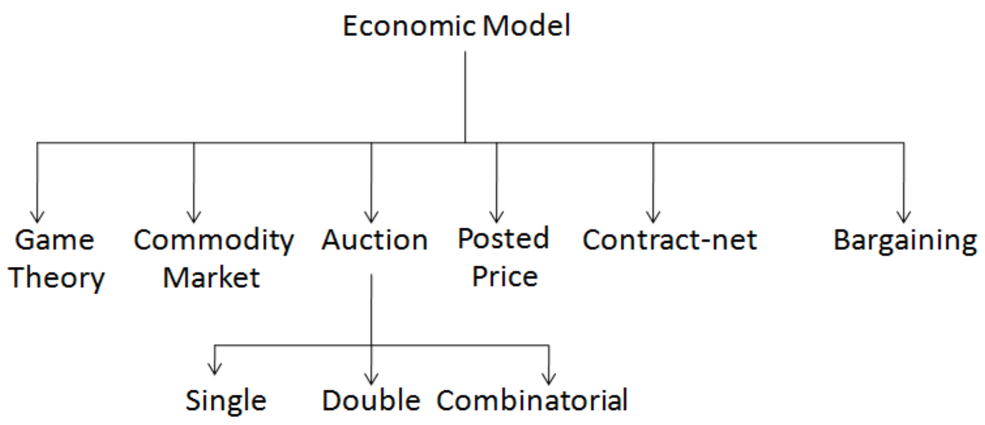


FIGURE 3.2 – Taxonomy og resource allocation.

3.4.1 Game Theory

A game theory approach is used when there is a selfish optimisation and individual utility functions and Grid participants interact in the form of an allocation game employing various strategies [Gar+11; Pou+06]. It has two types, cooperative and non-cooperative methods for resource sharing and allocation, and often employ "Nash bargaining" approach, where bargainers negotiate for a fair contract within feasible solution set. A cooperative game is usually used for resource allocation based on load-balancing with unselfish participants, and a non-cooperative game is used with self-interested participants. The use of games is not very common for resource allocation in market-oriented Grid Computing [Gar+11].

- An application of resource allocation heuristic methods in the non-cooperative game of Grid users in the commodity market model is presented by [Gar+10b]. The authors defined three heuristics, namely, Min-Min Cost Time Trade-off $MinCTT$, Max-Min Cost Time Tradeoff $Max-CTT$ and Suffrage Cost Time Trade-off $SuffCTT$ algorithms, for jointly optimising cost and execution time of user application in utility Grids. The trade-off factor indicates the priority of optimising cost over time. This system can be formulated as an economic model with three main participants: service providers, users, meta-broker. The meta-broker uses the information supplied by the providers and the users to match jobs to the appropriate services. So that, for each user application, the MinCTT algorithm searches the time slot on a resource with the minimum value of cost. From these user application/time slot pairs, the pair that gives the overall minimum will be selected. The MaxCTT searches the time slot on a resource with the minimum value of cost. From these user application/time slot pairs, the pair that gives the overall maximum will be selected. The SuffCTT algorithm assigns the highest priority to the application which would "suffer" the most if not assigned. The SuffCTT searches the time slot on a resource with minimum suffrage value which is the difference between its best and second best value of cost. From these user application/time slot pairs, the pair that gives highest suffrage value will be selected.

- A market resource allocation for hierarchical computational Grid based on a non-cooperative game is proposed by [Kol+11]. The authors propose two general non-cooperative game approaches for modelling Grid user behaviour defined as user requirements. The symmetric non-zero sum game, which means that the privileges to the resources are the same for all users, each user tries to choose an optimal strategy of mapping his tasks to machines in order to minimise the total cost. The asymmetric game in which one user acts as a leader and the rest of players (users) are his Followers, the leader choose first his strategy and the followers minimise simultaneously their cost functions relative to the leader's choice. For solving the games, it designs and implemented a hybrid genetic algorithm to approximate the equilibrium points for both games, and it considers nonzero sum games for which the equilibrium points are the results of minimisation of a multi-cost game function.

3.4.2 Commodity Market

In a commodity market, a number of resource categories are defined for which the market entity suggests a price, and consumers and providers create demand and supply for the different categories [Stu+07]. The pricing policy can be derived from various parameters and can be flat which means once pricing is fixed for a certain period, it remains the same irrespective of service quality, or it can be variable depending on the resource supply and demand. In general, services are priced in such a way that supply and demand equilibrium is maintained, if the demand increases or supply decreases, prices will increase until there exists equilibrium between supply and demand [Buy+01b].

- Nimrod/G [Buy+00] employs a commodity market model for resource allocation, designed for massively parallel applications on the scientific Grid environment, which takes the time and cost as parameters. The buyer is the end-user, the seller is the resource provider, the market is Nimrod/G components (the scheduler under the control of the Parametric Engine), and the good is a right to use the resource. The Selection of computational resources is handled in two ways, either the work is completed within a given cost and deadline, or the buyer is allowed to negotiate for a resource in the Grid. The system can employ resource reservation or trading technique to identify suitable resources. The user is then allowed to renegotiate with a different deadline or cost. The trading technique is that the buyer requests the scheduler via the Parametric Engine to arrange a resource that satisfies his deadline and cost, and the scheduler negotiates with the resource providers and selects one that meets the deadline at a minimum cost.

- G-commerce [**WBPB-2003**; Wol+01], a computational economies for controlling resource allocation in computational Grid. G-commerce uses two different market models, commodities markets and auctions, to measure the efficiency of the economic resource allocation. It defines resource consumers and Grid-aware applications that bid for goods they wish to use, and resource producers which represent resource owners who sell their resources (CPU slots and disk capacities) to the Grid. In addition to a market-maker which determines the winners either by the commodities market or by the auctions. The proposed market in this work is a traditional single-good auction and it is not a combinatorial one, which may not be very suitable for P2P Grids. In the auction model, it proposes to use separate auctioneers for each good, in this sense a consumer that needs a CPU and a disk must make two separate bids, and may risk winning just one of them at a time.

- Depoorter et al. [Dep+14] present an economic resource management system (RMS) that prices resource usage on co-allocated and reserved in advance. The proposed approach supports both network and computational resources. The RMS both allocates and prices resources in line with the demand and supply conditions in the network. The system is composed of several elements, a consumer that submits an application processing requests APR to the RMS that specifies the work-flow, input data requirements, and maximum budget that it is willing to spend for the execution of the application. A work-flow consists of a number of jobs that need to be executed for the application. A request that can be a data dependent APR $DDAPR$ if it requires input data, or a data free APR $DFAPR$ otherwise. A resource provider which will allocate the jobs. In addition to a broker that coordinates the resource co-allocation process, the broker is also given the mandate by consumers and providers to determine the cost for execution of the different APRs, in line with supply and demand conditions on the Grid and the reserve prices set by providers. In this model, consumers do not contact providers directly, all requests are scheduled by a broker. When the broker handles an APR, it sends a request bid to a provider in order to schedule either all jobs (for DDAPRs) or as many jobs as possible (for DFAPRs) at the provider's site. The provider selects the best one according to the closest deadline policy. A broker uses a greedy heuristic for selecting the order in which APRs are planned in. On the other hand, it uses a next highest losing bid strategy $NHLB$ to price individual APRs based on PAPRL, UPAPRL, and the reserve prices of the providers.

3.4.3 Auction

An Auction is the process of trading resources by offering them up for bid and selling the items to the highest bidder[Gar+11]. The auction model supports one-to-many or many-to-many negotiation, between a service provider(s) and many consumers, and reduces negotiation to a single value (i.e., price). In one-to-many auctions, one agent initiates an auction and a number of other agents can make a bid. In many-to-many auctions, several agents initiate an auction and several other agents can bid in the auction [Pou+06]. The auctioneer sets the rules of the auction, acceptable for the consumers and the providers. Auctions basically use market forces to negotiate a clearing price for the service. In economic terms, it is also a method to determine the value of a resource whose price is unknown [Gar+11].

- The first proposal of an auction-based resource management system for distributed computing is called Spawn [Wal+92]. It aims at utilising idle CPU times in a network of workstations. This system is composed of interacting buyers and sellers; buyers are end-users and the seller is the owner of the workstation, where the auctioneer is the Spawn system running on the seller's workstation. In this way, a buyer finds a seller and bids for the CPU time of the workstation, and the seller determines the winner which will use its workstation.

- Regev and Nisana [Reg+00] propose the $POPCORN$ market, which aims to be as an infrastructure for globally distributed computation over the Internet. POPCORN is a market model for matching sellers and buyers using auctions. The sellers provide their resources to a buyer by using Java enabled browser. The auctioneer is an independent market service on the Internet, it is responsible for performing matching between buyers and sellers, transferring task and result between them. Popcorn has a central repository by means of a Web platform for information aggregation. However, this platform may become a communication bottleneck and it does not support multiple kinds of resources and a combination of them.

- Pourebrahimi et al. [Pou+06] propose a market-based mechanism to allocate computational resources (CPU time) with a single central Market in a local Grid, using a double auction model. The system is composed of three entities: Buyer, Seller and Auctioneer. Buyers and sellers are autonomous agents that make their own decisions according to their capabilities and their local knowledge, and the auctioneer is an agent acting as a mediator between the consumer and producer agents. Buyers and sellers agents announce their desire to buy or sell processing power to the market. The market acts as an auctioneer and searches for possible matching between buyers and sellers by matching offers (starting with the lowest price and moving up) with demand bids (starting with highest price and moving down). When the auctioneer receive a new task query, the protocol searches all available resource offers and returns the best match which satisfies the task's constraints (such as resource size, time frame and price). If no match is found, the task query object is stored in a queue. The queries are kept in the queue till the *TTL* has expired or a match has been found. When a resource becomes available and several tasks are waiting, the one with the highest price bid is processed first.

- Denoeud-Belgacem et al. [Den+10] describe a combinatorial auction for resource allocation in the Grid system where multiple providers and multiple consumers may participate and trade in the marketplace, and consumers compete for different types of resources that they may execute their applications. They consider two types of resources, computational and storage resources that are specified by the quantity and quality attributes. For this end, they first specify a bidding language permitting to describe computing resources and a large number of typical users' requests, and they define some allocation rules adapted to the context of Grid networks. They assume that a bundle of resources offered by one provider is partially allocated and shared to satisfy several consumers. Then they provide a mathematical programming problem that formulates these rules and characteristics. After that, they propose an approximate method to treat auctions with large numbers of participants. They also take into consideration the pricing problem, to compute prices that represent the state of the market and bring trustworthy feedback to participants. They propose a pricing model that computes per-item pricing which allows users to deduce the price of bundles that they need by linear summation. As well as, they suggest a second model that computes prices as a function of time, thus permitting consumers to adjust their demand trading off price and time of execution.

- Venkataraman [Ven15] present the Mini-Grid framework, a context aware task allocation and scheduling in ad-hoc desktop Grids, based on auction market model. The Mini-Grid consists of four logical components: resource providers, resource consumers, task-bus and messenger component. Each resource participating in Mini-Grid has a software entity called "client", so, a client can play both resource consumer and resource provider roles. The clients in Mini-Grid communicate with each other using the messenger component. The computational tasks are exchanged between the clients using the task bus. Each client participating in the Mini-Grid environment has sub-systems among others the context awareness sub-system, which aims to describe the context of tasks and to collect and provide context information of resources. Clients playing resource consumer role accepts tasks coming from Mini-Grid applications, distributes the tasks to resource providers according to a scheduling policy based on the auction, transfers application code to executors, collects and stores task results and delivers task results to Mini-Grid application upon request. Client playing resource provider role listens for task announcements, participates in the auction process, executes allocated task and returns completed task along with results. This algorithm can be resumed as follows: a Bag-of-Task application generates tasks with a Task Context description, and submits them to the resource consumer which announces each task to all resource providers currently listening to announcements using messenger component. On receiving the announcement, the resource providers decides to participate or not into the auction. If yes, the resource provider needs to decide the bid that it is going to submit. To define the bid, the resource provider needs to use the resource context. This involves context query and processing, which consumes CPU time. Otherwise, it ignores the announcement. The resource provider submits the computed bid. The resource consumer proceeds and evaluates the submitted bids and selects the optimal resource provider for execution of the task, and announce the winner which will execute the task.

3.4.4 Posted Price

The posted price model is similar to the commodity market model, but providers can announce special offers for a specific period of time in order to attract new consumers. The posted price offers will have usage conditions, but they might be attractive for some users because they are generally cheaper compared to regular price [Buy+01b]. Prices do not vary relative to the current supply and demand but are fixed over a period of time, so there is no negotiation between the participants [Gar+11].

- Li et al. [Li+14] propose a posted price model for Grid resource allocation, they present a scheme which is combined of a Grid Resource Supermarket (GRS), a posted price model based on GRS, and an optimisation based on Multi-Objects Generalised Assignment Problem. The system is composed of GRS resource agent and GRS user agent and GRS manager. The GRS manager gets the permission of Grid resource from the providers or owners of resource in lower price and sell it at higher price to the Grid consumers, in other words, the manager of GRS gains the profits by serving the Grid resource provider and Grid resource consumer, and it can profit from price difference between the cost and sale price of the GRS resource. GRS resource agents take charge of collecting the information of Grid resource that can be shared by any others, negotiating the price and the other items with the resource provider. GRS user agents take charge of responding the Grid consumers and leading to the consumers sharing the resources.

3.4.5 Contract-net(Tendering)

Tender/Contract-Net model is modelled on the contracting mechanism used by businesses to govern the exchange of goods and services. The consumer advertises its demand and invites resource owners to submit bids[Gar+11; Xia+05a; Kal+04; Smi80]. Interested providers evaluate the announcement and respond by submitting their bids, the consumer consolidates all bids, compares them and selects the most favourable ones. The bidding process can be accepted or entirely rejected [Gar+11].

- Compute Power Market *CPM* [Buy+01c], a market-based resource management system for Grid computing, particularly low-end personal computing devices, it is a decentralised computation market with multiple markets and numerous consumers and providers spread across the Grid. The system transforms the meta-computing environment into a computational market in the form of computational power, storage, and special services. The system architecture and its components comprising of a market, resource consumers and resource providers. It supports three economy models: Commodity market, Contract-net/tendering and Auction models. The market consists of a market resource broker, the component of resource consumer and market resource agent. The market resource broker is responsible for negotiating cost with resource providers, of the selected resources based on the information provided by the market based on deadline or budget, and of task distribution to resource providers. The resource provider sells computational power using market resource agent, which updates resource information, deploys and executes tasks.

3.4.6 Bargaining

Bargaining models are employed for negotiations between providers and consumers and generally do not rely on third parties to mediate the negotiation[Gar+11], it is usually employed when market supply and demand and service prices are not clearly established [Buy+01b]. In this model, a negotiation is a form of decision-making with two or more actively involved agents who cannot make decisions independently, therefore must make concessions to achieve a compromise [Sim10; Ker+91]. Both parts have their own objective functions and they negotiate with each other as long as their objectives are met. Through bargaining, resource providers are given the opportunity to maximise their return-on-investment and consumers to minimise the price they pay for utilising Grid resources[Gar+11; Buy+01b]. The bargaining model has two types: bilateral and multilateral. In bilateral bargaining, only one consumer and one provider are involved in the negotiation. In multilateral bargaining, different resources' providers throughout their agents can negotiate with multiple consumers. Multilateral has two types two: one-to-many and many-to-many.

- Haque et al. [Haq+15] focus on developing an adaptive resource management architecture capable of dealing with multiple economic models. They consider five most widely proposed economic models in the Grid: Commodity Market, Bargaining, Auction, Continuous Double Auction and Contract-Net protocol (tendering), so they seek for the opportunity of utilising the potential of different models in different scenarios. The system component are users, resource providers and brokers. A broker performs all the crucial tasks on behalf of a user and it is also considered as job-scheduler; thus, it collects identifier of the available resources from Grid information service and starts communicating with resources and negotiating based on constraints defined by the user and the current economic model. The main contribution of this work is a switching framework that dynamically switches from one economic model to another and is able to adapt to its consequences in the environment throughout the use of a switching agent which automatically decides which model to be used when and for what purpose. Example, if the network notices that its current demand is low regardless of its supply, it can employ commodity model because this model has been identified more strong in this case. Likewise, if the network notices that its supply has been decreased moderately and demand has been increased, it can switch to Contract-Net protocol. As a result, the network would be able to use the potential of different models in different scenarios, thus optimising the defined function in general. Therefore, broker, resource and auctioneers in such an environment must have dynamic capabilities to deal with different models. Thus, it designs the parameters and the organisation of a broker in a way so that it can adapt with changing behaviours in the environment. A resource model has generic capabilities to deal with multiple models. And the auctioneers has the ability to start processing as soon as they are invoked by the system.

- Adabi et al. [Ada+14] propose a bargaining resource allocation for Grids. They design an enhanced market and behaviour-driven negotiation agents $EMBDNAs$, that adopts a fuzzy negotiation protocol. The negotiation protocol focuses on handling multiple trading opportunities and market competition and designing two fuzzy Grid market pressure GMP determination systems $FGMPDSs$ for both Grid resource consumers and Grid resource owners to guide negotiator agents in relaxing their bargaining terms under intense GMP to enhance their chance of successfully acquiring/leasing out resources. The negotiation model has three parts: the used utility models or preference relationships for the negotiating parties, the negotiation strategy applied during the negotiation process and the negotiation protocol.

- Adabi et al. [Ada+13a] suggest a market-based Grid resource allocation using new negotiation model for both bilateral and multilateral bargaining. The work uses multi-agents paradigms to design a market behaviour-driven negotiation. To find the amount of concession that should be made by each type of negotiators, it assumes the knowledge of the number of competitors and behaviour of negotiator's trading partner. To estimate its new price, it uses six factors: number of negotiator's trading partners, number of negotiator's competitors, negotiator's time preference, flexibility in negotiator's trading partner's proposal, negotiator's proposal deviation from the average of its trading partners' proposals, and previous concession behaviour of negotiator's trading partner.

- Haghtalabi et al. [Hag+14] propose a non-cooperative bargaining model for resource allocation in Grids. In this model, every customer declares his needs together with preferences and sellers search for the best and most appropriate proposed price for the requested service. At first, the customer's agent declares the needed service to the resource broker. Customers' agents specify the amount of data, the ability to process the required work and deadline before bargaining starts in the negotiation process. From a customer's point of view a buyer can enter negotiation and bargaining for a specific resource with a few sellers simultaneously and whoever reaches an agreement sooner the other negotiations are cancelled. In multiparty negotiations, a customer must compete against other customers to succeed with the largest profit. On the other side, from a seller's point of view, all other sellers are considered rivals and every seller tries to satisfy their previous customers so that they return for future purchases. To this end, the seller records all the customers' previous purchases and with regard to that record tries to reach an agreement sooner, so regular customers will have higher priority.

- Adabi et al. [Ada+13b] propose negotiation strategies considering the market, time and behaviour functions for resource allocation in computational Grid. The system elements are Grid resource owners $GROs$ and Grid resource consumers $GRCs$, it uses a multi-agent-based negotiation model for interaction between GROs and GRCs in both bilateral and multilateral negotiation strategies. This work considers different concession amount for different negotiator's trading partners by applying a multi-criteria decision function which provides more flexibility in keeping the chance of making deal with more than one opponent by computing rational and sufficiently minimum price. It proposes a Market and Behaviour-driven Negotiation Agents $MBDNAs$ which adjusts the amounts of concession by considering several factors: opportunity, time, competition and previous concession behaviour of negotiator's trading partner. The negotiation model in this work applies a negotiation strategy which models the market conditions, time and concession behaviour of negotiator agent's trading partner to determine the proper amount of concession. The multi-criteria negotiation strategy maximises the negotiators' achieved utility and takes a single-issue (price-only) negotiation, while the time aspect is taking in consideration but for the time spent in negotiation before the allocation process was achieved.

- SIM [SIM06] proposes a negotiation approach for Grid resource co-allocation, they suggest that a relaxed-criteria Grid negotiation $G - negotiation$ mechanism may enhance the success rates of negotiation agents in Grid resource co-allocation. Thus, they propose a relaxed-criteria negotiation mechanism for supporting Grid resource co-allocation by allowing multiple concurrent pairs of negotiations simultaneously and coordinating the concurrent negotiations. The system is composed of market-driven G-negotiation agents representing resource providers and consumers. It uses heuristics to guide G-negotiation agents to slightly relax their bargaining criteria under intense pressure (e.g., acquiring a slightly more expensive resource when many resources are already occupied) in the hope of increasing the chance of acquiring all required resources and acquiring them more rapidly. The negotiation process proceeds in a series of rounds, where a pair of consumer and provider agents negotiates by making proposals in alternate rounds. While it is possible that multiple consumer-provider agent pairs can negotiate deals simultaneously. Each agent initially proposes its most preferred deal (price), if no agreement is reached, negotiation proceeds to the next round. Negotiation between two agents terminates when an agreement is reached, or with a conflict when one of the bargaining agents' deadline is reached. During rounds of the negotiation, negotiators made concessions according to three functions, time, competition and opportunity which are formally modelled in this work.

- An et al. [An+08] present a framework for automated negotiation in dynamic and complex environments like Grids. They consider one-to-many multilateral bargaining from the consumer perspectives. For this end, they designed a decision-making negotiation strategy based on Markov chain to allow the consumer agent to make a decision on when to complete the negotiation. The framework allows the computing of the expected utility for the buyer for the next round of bargaining, then it compares the expected utility with the actual one, upon which the buyer can make decisions. The Markov chain model for decision-making captures the variables that influence the buyer agent's utility values and the uncertainties associated with them.

- An et al. [An+16] present an alternating-offers bargaining in one-to-many and many-to-many settings for distributed and dynamic environments like Grids and Clouds. This work is considered as a general negotiation protocol between buyers and sellers over the price of a good to allocate the resources.

- An et al. [An+10] suggest an automated multilateral negotiation with decommitment for dynamic resource allocation in a dynamic environment like Grid and Cloud computing. The environment components are buyers and sellers, designed by agents that negotiate over both a contract price and a decommitment penalty, where buyers have strict deadlines for their tasks. To accommodate the highly dynamic nature of cloud computing platforms, they introduce a negotiation mechanism where an agent is able to decommit from a contract by paying a penalty to the other contract party. Thus, an agent may find it advantageous to decommit from existing contracts. They associate to each agent a set of actions to enhance its flexibility and adjust its decisions, such as making offers, by reacting to changing negotiation status, while also considering the time constraints, resource competition, and resource cost.

- Chunlin et al. [Chu+12; Chu+09; Chu+08] propose a utility-based resource allocation for Grid computing. They model the system by utility functions to represent the Grid user objective function to complete its tasks with respect to a budget and deadline, the Grid provider objective function to maximise its profit and the whole grid system objective optimisation that provides a joint optimisation of both user and provider. They use Lagrangian theory to optimise the system and its components' objectives functions. Based-on bargaining aspects, they propose a set of negotiations between the user and provider that negotiate over the cost of resources and the deadline for tasks' executions.

- Kong et al. [Kon+15] address the problem of resource allocation in dynamic and open Grid environments. They propose a multilateral bargaining process composed of providers and consumers negotiating through their agents where each agent only has a local view. They consider a loosely coupled resources, such as the Grid computing, the computer storage, or even some virtual resources like the electric data library of some university, in which resources can be used for only one task at a moment. Consumers first seek to find the potential resources, then they begin to negotiate with the resource providers. The proposed process of negotiation supports the decommitment and penalty, and consumers constraints are the starting time of the task, deadline and the maximum reward that the task's owner can gain when the task is allocated successfully. The problem is how to allocate tasks under time constraints in dynamic and open Grid environments. The proposed solution uses the alternating offer protocol and the agent local information for the negotiation.

3.5 Discussion

In this chapter, we have displayed a literature review covering several aspects of resource management in P2P Grid Computing. We have presented related works on resource discovery and overlay constructions and management, where we have focused on two principal metrics to evaluate the proposed works; the efficiency and the scalability. As well as, we have surveyed works on the economics resource allocation.

Our work differs from the cited works in the following points:

First of all, in the ultra-peers paradigms, we define two kinds of scalability, the intra ultra-peers scalability and the inter ultra-peers scalability.

The first one consists of the excessive inter ultra-peers communications that consume bandwidth and introduce latency. This issue may be caused by poor routing algorithms and a poor grouping of nodes to build the ultra-peers clusters. It is resolved by our first contribution where we use SW and define the SKOS ontology to construct the architecture of the system and to build the overlay network. In addition, we use the ontology to route queries between the connected ultra-peers based on semantic relations between them, which makes our proposal more efficient and more scalable. Many works have addressed the problem of overlay construction and routing processing, but no one of them has fully exploited the strength semantic offered by SW and ontologies.

The second issue consists of the high density of nodes belonging to each ultra-peer and their engendered workload, that produce insufficient local resources and cause a bottleneck on the ultra-peer; it is resumed on the local ultra-peer communication. Conversely, there is little attention paid to this issue. Our second contribution directly addresses this issue. When it proposes a scalability aware approach for ultra-peers based on the prediction of the workload on each ultra-peer in the network, and the analysis of the estimated workload to decide whether it causes a bottleneck or not. It proposes also a set of strategies to follow to keep the ultra-peer away from any bottleneck and make the system scalable. Contrary to works presented by [Llo+14] and [Ard+12], in our work, each ultra-peer reacts independently to the other ultra-peers, and it treats its scalability issue as a local problem without any disturbance of the other ultra-peers. Our proposed solutions are very scalable; they allow to support the growth of the environment size, and they can add and integrate new ultra-peers easily.

Third, the economic approaches have been discussed, through which we can draw several remarks illustrated in table 3.1 which is extended from [Nez+16]. In the meanwhile, we can understand that no approach is better than or outperforms the other approaches, deciding which model to use depends on the nature of the environment and the problem specification. In addition, there is no single model can deal with every scenario [Haq+14].

In our proposal, we have adopted the bargaining mechanism for P2P Grid resource allocation because its features are appropriate to the characteristics of the studied environment. In P2P Grid, nodes (users and providers) are characterised by the selfishness and the volatility which do not allow to make a clear vision on the demand-and-supply, hence, to establish prices. In this case, the bargaining will help users and providers to valuate the goods and helps to better understand the requirements of the market participants. On the other hand, the allocation process should be distributed to alleviate the workload on the ultra-peer, negotiations on prices and other requirements do not need a third-party and have to pass directly between the involved entities (users and providers). In the context of the Grid, bargaining is proposed to be suitable because it supports utility-based negotiation between a user and a resource provider, and this may ultimately result in a both sides satisfaction [Ass+07].

Economic model	Execution time	Income	Profit	Centralisation	Communication
Commodity	Very fast	High	Provider	Yes	Low
Posted price	Very fast	High	Provider	Yes	Low
Contract-net	Very fast	Low	User	Yes	Moderate
Bargaining	Fast	Average	Both sides	No	High
Game theory	Fast	High	Both sides	No	High
Combinatorial auction	Fast	High	Both sides	Yes	High
Double auction	Fast	Average	Both sides	Yes	Moderate
Single auction	Fast	Average	Provider	Yes	High

TABLE 3.1 – Comparison of different economic approaches of resource allocation.

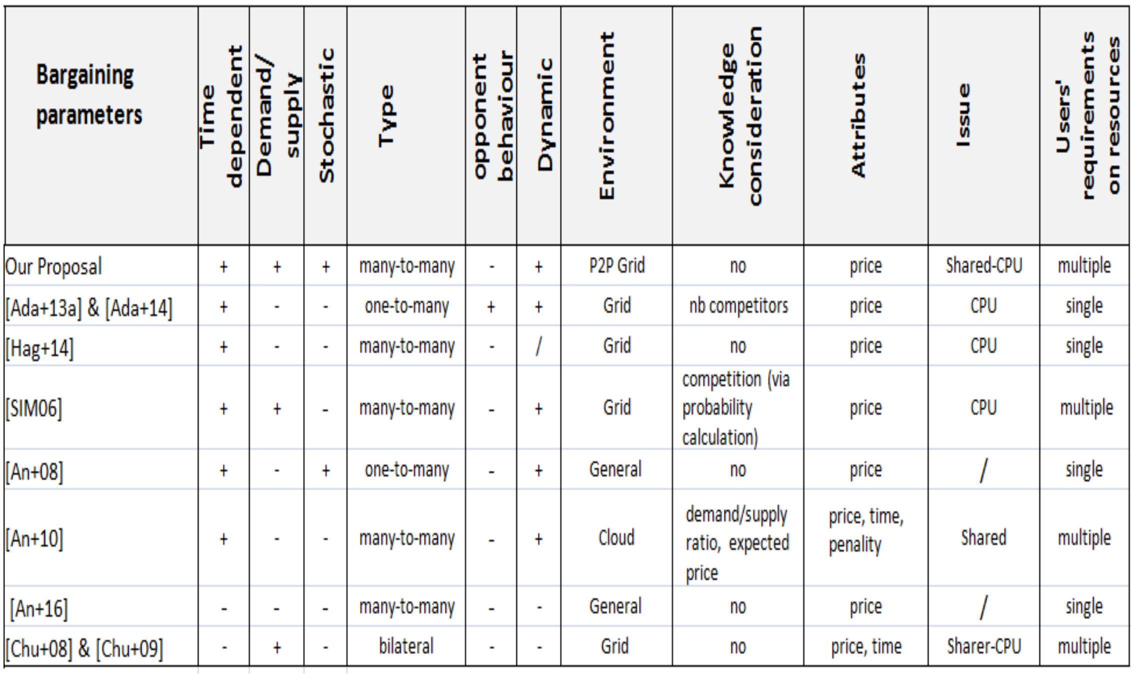
In our work, the proposed distributed and multilateral bargaining mechanism first models the users and providers by utility functions, after that it proposes a bargaining protocol for both parties to allow them to simultaneously optimise their objective functions and negotiate the price with each other. The model takes into consideration different elements that characterise the target environment. It defines the concession (for users and for providers) that must be made at each bargaining round by the time-dependent function, the demand-and-supply and the difference between the offered prices and its price. So the amount of concession will be influenced by the time passage and the demand-and-supply, in addition to the proposal of the other negotiator. This makes the concession more realistic, and narrow the difference between the two different proposals to arrive at an agreement.

Although, key differences between our process of bargaining and existing works are highlighted in the following analytical table 3.3.

The table presents several factors needed to design the bargaining process. However, developing a process that considers all these factors simultaneously is very complicated.

3.6 Summary

In this chapter, we have presented a literature review of resource discovery mechanisms and overlay construction approaches in P2P Grid Computing. We have discussed the related works from their efficiency and their ability to scale in large-scale environments. Then, we have reviewed related works regarding two other axes: traffic and workload analysis and prediction on networks and on Internet applications, and the scalability of systems using ultra-peers overlay networks, in P2P, Grids and Clouds environments. After that, we have introduced



Bargaining parameters	Time dependent	Demand/ supply	Stochastic	Type	opponent behaviour	Dynamic	Environment	Knowledge consideration	Attributes	Issue	Users' requirements on resources
Our Proposal	+	+	+	many-to-many	-	+	P2P Grid	no	price	Shared-CPU	multiple
[Ada+13a] & [Ada+14]	+	-	-	one-to-many	+	+	Grid	nb competitors	price	CPU	single
[Hag+14]	+	-	-	many-to-many	-	/	Grid	no	price	CPU	single
[SIM06]	+	+	-	many-to-many	-	+	Grid	competition (via probability calculation)	price	CPU	multiple
[An+08]	+	-	+	one-to-many	-	+	General	no	price	/	single
[An+10]	+	-	-	many-to-many	-	+	Cloud	demand/supply ratio, expected price	price, time, penality	Shared	multiple
[An+16]	-	-	-	many-to-many	-	-	General	no	price	/	single
[Chu+08] & [Chu+09]	-	+	-	bilateral	-	-	Grid	no	price, time	Sharer-CPU	multiple

FIGURE 3.3 – Analytical table of bargaining-based approaches for resource allocation.

a literature review on economic resource allocation in P2P Grids, and we have focused on the bargaining approaches for resource allocation.

Chapter 4

On the Construction of Semantic Based Overlay for P2P Grid

4.1 Introduction

This chapter highlights the first part of thesis contributions: the resource discovery (RD) and overlay architecture construction that is based on the ultra-peer paradigm more specifically, the master-worker version for P2P Grid Computing using Semantic Web (SW) and a lightweight ontology. We present the SKOS lightweight ontology that describes domains of applications in this environment. Based on this ontology, We introduce:

— A semantic clustering of nodes based on their domains of interest to form groups called federations;

— A new approach for constructing federations and the overlay architecture;

— An efficient and scalable RD process that exploits the richness of the ontology to route the queries between the semantically related federations.

This first part of our contributions is considered as a socle for all the other contributions, in which we will present the basic architecture upon which all the system is built.

4.2 Motivations and Design Objectives

RD involves searching for resources that match the user's applications requirements. In large-scale environments, RD policies are challenged by the potentially large number of resources and heterogeneity of resources and requests [Tan+05]. RD solutions are widely tied to the underlying architecture of the system, the traditional solutions rely primarily on centralised and hierarchical architectures, they have the ability to decrease the average response time and a good efficiency, but they do not scale well. Distributed solutions bring some benefits like adaptation, self-organisation and fault tolerance; however, they are less efficient and suffer from the risk of network congestion and churn effect. Ultra-peers or hybrid approaches which are based on nodes clustering have been proposed [Yan+03] to combine the advantages of both previous architectures; so that to be as efficient as the centralised architecture and to scale well like the decentralised ones.

However, in large-scale distributed environments like P2P Grids, where the number of nodes is constantly increasing and resources are heterogeneous, the quality of RD is determined not only by efficiency and scalability; but also by its accuracy that measures the quality of the discovered resources in terms of relevance and precision. In addition, the lack of a good and effective representation of nodes in a highly heterogeneous environment often results in an irrelevant affectation of nodes to the right group that is considered as one of the reasons for which queries can fail to find relevant resources.

In order to improve the precision of a discovery and query processing between federations, domains of interest of nodes must be given well-defined meaning. In heterogeneous environments like P2P Grids, syntactic keyword and taxonomy-based matching are not sufficient to achieve high-precision RD because of the disagreement about the meaning, interpretation or intended use of terms. To overcome this limitation, the usage of semantic technology is regarded to enhance the quality of RD. Hence, combining hybrid architectures with SW technologies would bring more benefits to RD process.

SW as defined by [Ber+01], attempts to define the meta-data information model for the World Wide Web to aid in information retrieval and aggregation. SW improves the effectiveness of nodes' information, resources and query representation, thus the efficiency of searching. Ontology is an explicit specification of a conceptualisation [Gru93] that serves as a foundation for formal representation of knowledge. However, there is no single way to represent domains of interest of nodes in Grids as concepts into ontology where their semantics may be represented by different labels, or that the same label could mean totally different things. Therefore, two semantically equivalent nodes may belong to two different federations, while one federation can contain semantically different nodes. We believe that if we can give for each node a good representation that is enriched by supplementary information about context and synonyms, this problem will be resolved easily.

Simple Knowledge Organisation System (SKOS) [Mil+09] can describe synonyms and associative relationships, and to add information to concepts, which can be easily used for defining ontological terminologies. It will be the best candidate to represent domains of interest of nodes. It provides a standard way to represent knowledge organisation systems using the Resource Description Framework (RDF). Encoding this information in RDF allows it to be passed between computers' applications in an interoperable way, and to be used in distributed and decentralised meta-data applications [Mil+09].

Since we adopt a hybrid architecture based on the leader-workers paradigm where a leader acts as a centralised resource for a number of nodes and the system can be viewed as a network composed of small-scale or groups which we called federations; therefore we need to cluster nodes to form these federations. Hence, the criterion of grouping nodes is according to their domain of interest because the more nodes share similar domain of interest the more their resources tend to be similar. In addition, there are applications such as biological applications whose computational requirements exceed even the fastest technologies available. It then desirable to efficiently aggregate distributed resources owned by collaborating parties that share the same domains, so that to enable processing of a single application within a reasonable time. For this purpose, we develop a SKOS lightweight ontology of Domains Description (OntDD) that is used to classify P2P Grid domains' applications, so that to use it thereafter to cluster nodes into their corresponding federations and to construct groups of similar interest.

On the other hand, classifying nodes according to their domains of interests and assigning them to the right group to construct the overlay network have to pass by measuring the similarity between nodes, which is primordial. These measures of similarity do not have to be complicated and time-consuming; at the same time, they have to be efficient. Most works in this field suppose the existence of several ontologies to compare [Li10], [Pir+12], then they suggest solutions based on finding the appropriate mappings between two nodes using several measures of similarities. Pirro and his colleagues in [Pir+12] assume that a node receives both the category ontology and the domain ontology from the node that contacts it to join the network. However, in a real scenario, not all nodes are aware of the semantic representation of their domains of interest; yet, this fact may hinder their affectation to the corresponding groups.

For this reason, in this work, we use a mechanism that exploits the richness of a SKOS OntDD to find the semantic similarity between two nodes and to create the overlay network. We also propose an algorithm of query processing that handles queries over semantically inter-connected federations. This will perform an effective searching according to the semantic distribution of nodes into federations and thus to their resources.

4.3 Overview of the System

This section starts with a brief description of SKOS where we demonstrate its key differences with other ontologies' languages. Then we give an overview of the system, and we finally illustrate the proposed architecture of the system.

4.3.1 SKOS and Semantic Web Technologies

Technologies such as RDF and Web Ontology Language (OWL) [Sah07] are seen as key elements for building Semantic Web applications. The SKOS model is built in accordance with RDF and OWL that has a serialisation to the RDF. (See Listing 4.1 for a turtle example).

In general, Knowledge Organisation System (KOS) differs significantly from formal ontologies represented by using OWL [Hor+03]. As they do not contain detailed intentional descriptions of concepts, SKOS provides looser semantics than OWL [Bec+08a]. The SKOS model can be used to structure and represent any knowledge that contains statements about concepts and the relationships between them. SKOS [Mil+09] can describe synonyms and associative relationships, yet to add information to concepts that can be easily used for defining ontological terminologies. OWL is a rigid and very expressive language. It provides a formal knowledge representation language to describe semantic resources; however, it is beyond these capabilities. SKOS and OWL have different semantics; still, they are intended for distinct applications. For our purpose, nodes' domains of interests will be represented as concepts that are ordered in some kind of hierarchy without instantiating them. In addition, the ontology is needed to serve as a classification system; yet, we do not need lots of inferences to make a new piece of knowledge. A lightweight ontology will fulfil these requirements. Therefore, the degree of formalism of OWL is not necessary even its expressiveness is not desired in this kind of application. The reason behind is that the construction and maintenance of the lightweight ontology are much easier than that of the ontology which has maximum expressiveness and supports complicated reasoning [Val+04].

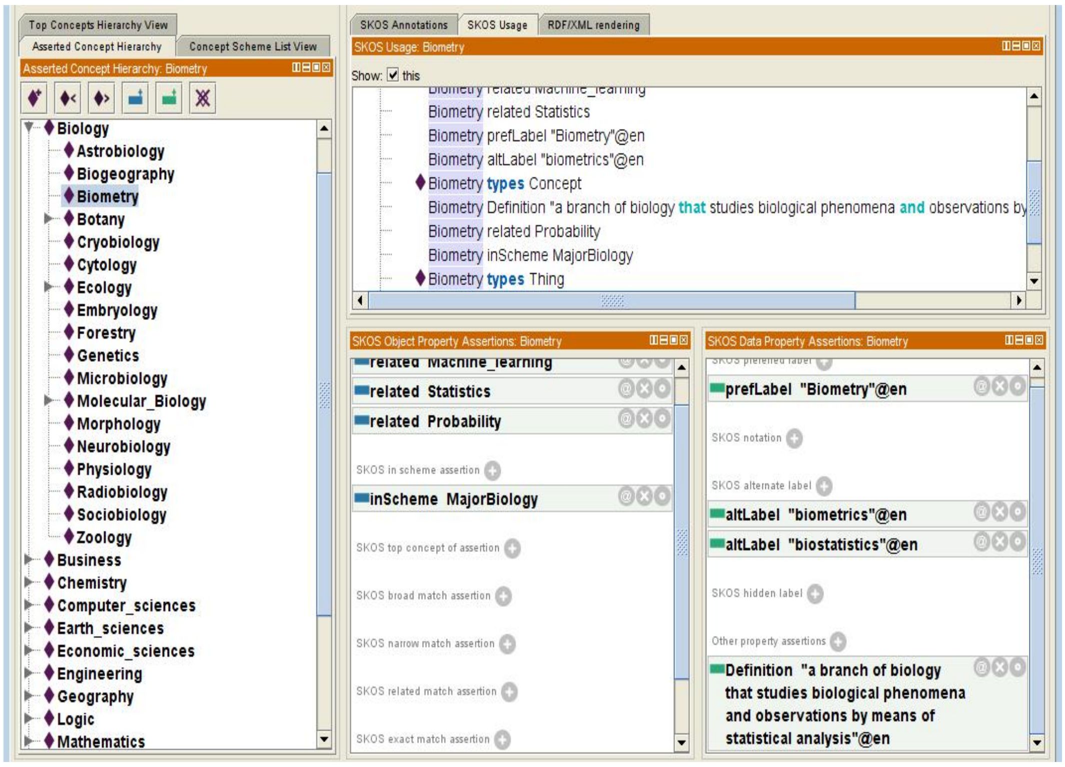
SKOS that used mainly for knowledge organisation has a weaker semantic relationship when compared to OWL such as hierarchical (Narrower/Broader) and associative (Related), lexical relationship for preferred label, alternative label and hidden label. SKOS provides a data model that can be used to express these kinds of relationships between resources and it is designed to be extensible and modular. Central to SKOS is the core vocabulary deemed sufficient to represent most of the common features found in concept schemes. A Concept can be considered as any unit of cognitive thought. Lexical labels allow the association of lexical forms (preferred labels alternative labels and so on) with each concept. Semantic relations capture relationships between concepts, including hierarchical broader-narrower relationships and general associative relationships [Jup+09].

4.3.2 The Construction of Semantic Federations based on SKOS Ontology

The computing Grids can create federations in scientific domains, such as physics, earth science and so on. Each federation is formed by a collection of nodes with the same domain of interest because we believe that the more nodes share the same interest, the more their resources tend to be similar. A federation is managed by a leader and consists of members that serve as workers. Communication and collaboration can operate on top of the federations. To create Grid federations, we need a classification technique to cluster the nodes; since each node has a specific domain of interest. Ontology of Domains Description (OntDD), a lightweight ontology, is created to classify Grid domains' applications in general. We used SKOS vocabulary, SkosEd editor, Skos API and Protégé [Hor+04] to formalise it.

For example, The *Biology* domain will be an individual of Skos:Concept in this ontology. We refer to individuals in this work by concepts. *Biology* may have sub-domains like *Ecology* and *Botany* where each of them is an individual of Skos:Concept too. These latter are related to *Biology* by narrower (more specific) or broader (more general) relations. We consider each concept as a federation. By using $skos : ConceptScheme$ that is viewed as an aggregation of one or more SKOS concepts [Mil+09], we create collections called Major-Domain-Federation (MDF) to regroup concepts belonging to the same inherent category, using the $skos : inScheme$. A $MajorBiologyFederation$ will be a category of federations where the common domain of interest is a field of biological sciences. In this work, each concept would be strictly linked to just one concept scheme.

In this way, domains of interests of the P2P Grid will be organised into categories and inside each category, they will be organised into hierarchies. We


Top Concepts Hierarchy View
Asserted Concept Hierarchy
Concept Scheme List View
Asserted Concept Hierarchy: Biometry
SKOS Annotations
SKOS Usage
RDF/XML rendering
SKOS Usage: Biometry
Show: this
SKOS Object Property Assertions: Biometry
SKOS Data Property Assertions: Biometry

Biology
Astrobiology
Biogeography
Biometry
Botany
Cryobiology
Cytology
Ecology
Embryology
Forestry
Genetics
Microbiology
Molecular_Biology
Morphology
Neurobiology
Physiology
Radiobiology
Sociobiology
Zoology
Business
Chemistry
Computer_sciences
Earth_sciences
Economic_sciences
Engineering
Geography
Logic
Mathematics

Biometry related Machine_learning
Biometry related Statistics
Biometry prefLabel "Biometry"@en
Biometry altLabel "biometrics"@en
Biometry types Concept
Biometry Definition "a branch of biology that studies biological phenomena and observations by
Biometry related Probability
Biometry inScheme MajorBiology
Biometry types Thing

related Machine_learning
related Statistics
related Probability
SKOS in scheme assertion
inScheme MajorBiology
SKOS top concept of assertion
SKOS broad match assertion
SKOS narrow match assertion
SKOS related match assertion
SKOS exact match assertion

SKOS preferred label
prefLabel "Biometry"@en
SKOS notation
SKOS alternate label
altLabel "biometrics"@en
altLabel "biostatistics"@en
SKOS hidden label
Other property assertions
Definition "a branch of biology that studies biological phenomena and observations by means of statistical analysis"@en

FIGURE 4.1 – A screen-shot of SKOS ontology.

enhance and enrich the semantic and the contextual meaning of concepts by creating different alternative label relations to represent their synonyms, in addition to another property assertion labelled associated_term. The latter represents terms associated to the concept that are different from its synonyms, like specific terms used in such a domain. Figure 4.1 shows the ontology written in SKOS which is created with the help of WordNet [Fel06], WikiPedia and Unesco [1].

The code in the listing 4.1 is an example of a concept presented in this ontology encoding in Turtle notation [Bec+08b].

4.3.3 Semantic Similarity

There have been extensive researches focusing on measuring the semantic similarity between two objects and finding appropriate mappings in the field of information retrieval and ontologies integration [Kal+03]. However, these methods are very complicated and computationally intensive, and principally based on the mapping between concepts belonging to two different ontologies. Nevertheless, in a real scenario, not all nodes are aware of the semantic representation of their domains; in accordance with, this may hinder their affectation to the relevant group they have to belong to. That is why applying semantic similarity measures between nodes may not be a good choice. In this thesis, we propose a method to compute the semantic similarity between two nodes, that exploits the richness of our SKOS OntDD ontology. OntDD provides essential inference needed for

1. http://skos.um.es/unescothes/?l=en

this type of application that is expressed by relations such as more specific and more general, and a meaningful classification and representation of domains of interests enriched by synonyms. We will query OntDD using the query language for Semantic Web, the Simple Protocol And RDF Query Language (SPARQL) [Ste+13].

New node wanting to join a Grid has to send a query including its domain of interest (DI), or a hierarchy if it exists. Our strategy of the semantic affectation of a node to the appropriate federation is as following: first, the algorithm tries to find any exact match between *DI* and any existing federation in OntDD; then, it will compare DI with all types of labels of concepts in OntDD to cover any presence of synonyms by running the query in listing 4.2 which would return federations that exactly match the request.

```
<#Ecology>
OntDD: Ecology
a   skos:Concept, owl:Thing;
OntDD:Definition "the branch of biology concerned with the
   relations between organisms and their environment"@en;
skos:altLabel "environmental science"@en, "bionomics"@en;
skos:broader  OntDD: Biology;
skos:inScheme OntDD: MajorBiologyFederation;
skos:narrower OntDD: Paleoecology;
skos:prefLabel "Ecology"@en.
```

Listing 4.1 – The SKOS encoding in Turtle notation for OntDD concept.

```
PREFIX res: <http://www.owl-ontologies.com/Ontology1293258633050.
   owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>}
  SELECT  ?term ?label
  WHERE{
          ?term a skos:Concept.
          ?term skos:prefLabel | skos:altLabel | skos:hiddenLabel
             ?label.
  FILTER (?label = ?key@en)
          }
  ORDER BY (?scheme)
```

Listing 4.2 – SPARQL query to find matching between concepts of OntDD and node *DI* key.

If there is no exact match, the algorithm runs the partial match by substituting the *Filter instruction* in listing 4.2 by the instruction in listing 4.3.

```
"FILTER ((REGEX(?label, replace(?key, \" \",\"| \" ),\"i\")))."
```

Listing 4.3 – The regex instruction.

As a matter of fact in OntDD, we use the term "Environmental Sciences" as a preferable label to represent the field of the science concerned with the relations between organisms and their environment; wherein the DI may be expressed by a natural language as "sciences of environment". The exact match query will return no results where the partial match will return the federation "Environmental Sciences" as a result. According to the number of returned results. If there is only one result, the leader sends the IP addresses of existing federations because we may have many federations for the same domain of interest, the new node tries to contact these federations for a potential join. Otherwise, if there are several distinct results due to a partial match or synonyms, the leader has to interact with the new node to choose the right match. It returns to a new node: definitions and some additional information like concepts narrower and broader to DI where the new node would be able to decide which federations are more convenient. Just after, the leader sends IP addresses of the selected federations. In case where no result is found, the leader requests the new node to refine the search further.

4.3.4 Layered Architecture based on Semantic Federations

As illustrated in Figure 4.2, we construct the overlay network architecture which is composed of three layers, and we design a hybrid layered architecture in which each federation is structured following leader-workers paradigm to perform data retrieval of available resources.

The layers are described as follows:

— The physical layer: it represents nodes in a real network as unstructured network architecture. Edges in this layer represent physical connections;

— The federations' layer: it represents the overlay network to maintain federations and process queries. Each federation captures a concept defined in OntDD and has a leader and workers. A leader is a representative of its federation which is selected among the other nodes. Each leader node has links to a set of semantically related leaders' nodes and links to all of its own worker nodes. Communications are limited to leaders and between a leader and its workers, which reduces the overhead;

— The leaders' layer: since each concept in OntDD represents one node, which is the leader of the federation, leaders can be organised as a hierarchical structure. This hierarchy between federations' leaders is generated by traversing the configured properties $Skos : narrower$ and $Skos : broader$ that are used to express the hierarchical relations between concepts in our case federations' leaders. In addition, with SKOS, we could organise federations (which are individuals of Skos:Concept) into categories using subclasses of Skos:Concept, we call them Major-Domain-Federation MDF. A category serves as a grouping mechanism for concepts (leaders of federations) of the same inherent category, concepts being an instance of one of those

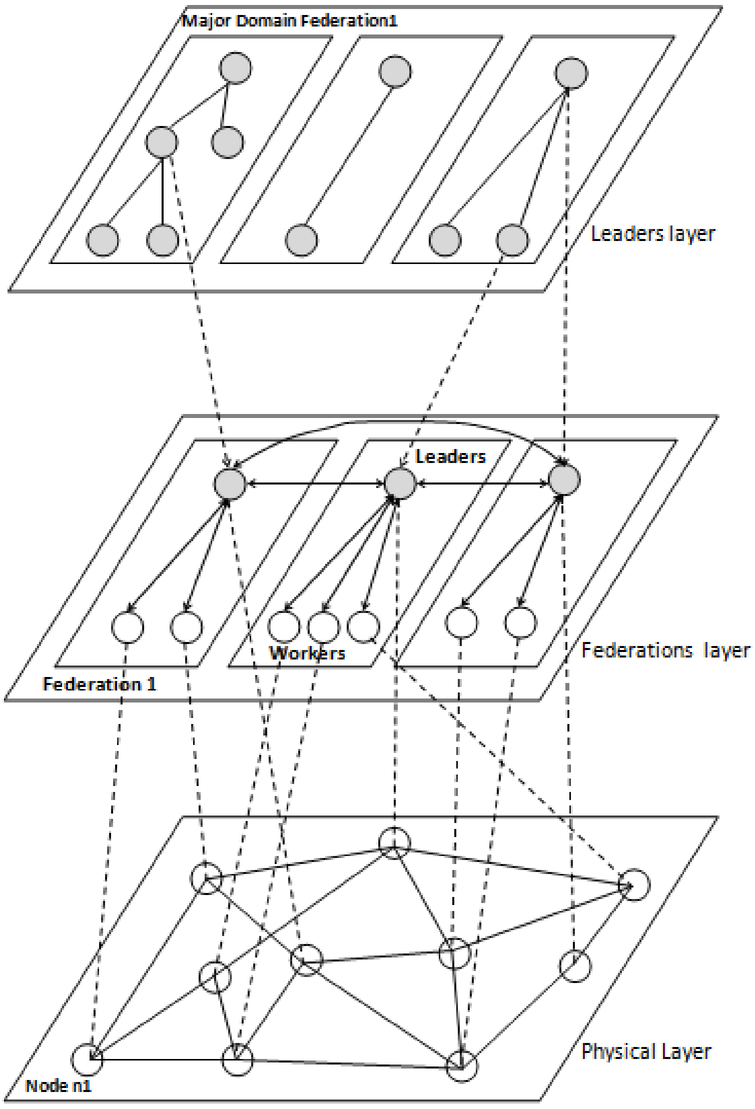


FIGURE 4.2 – The hybrid layered architecture.

categories. For example, *MajorBiologyFederation* is a category of all federations for which the domain of interest is one field of *Biology*.

This hierarchy organisation helps to limit the query search space from the entire P2P Grid to a federation through a single step, and resource location inside the federation in the next step. If the federation is unable to respond to the query, it forwards the query to its semantically interconnected federations within its relative hierarchy that may satisfy the query. This forwarding mechanism between leaders' federations achieves high resource discovery efficiency by keeping resource discovery scope at the federation leader level and between semantically interconnected leaders only.

4.4 P2P Grid Resource Discovery Mechanism

After introducing the overlay network construction, this section shows how to use the aforementioned layered architecture to perform the RD process. We present the algorithmic details and we discuss how federations are maintained in terms of adding and removing nodes, and how queries are handled.

4.4.1 Federations' Maintenance

When a node joins the network, it connects to any existing node in the network by sending a subscribe message. If the receiver node is not a leader node; it transfers the request to its leader. The leader measures the similarity between a domain of interest DI of node and concepts of OntDD, and then assigns it to the appropriate federation following the algorithm 2.

Algorithm 2 Algorithm Join (N, X): Node N joins the Grid through node X

```
Require: $N'$s $DI$
Ensure: $L$ the list of addresses of candidate federations
 1: if $X.isleader$ then
 2:   Search for any match between $N$ and concepts of $OntDD$.
 3:   if $X.DI$ match $N.DI$ then
 4:     $Add(Y)$
 5:     Update knowledge base of resources.
 6:   else
 7:     if $N.DI$ match $other\_federations$ then
 8:       $L.Add()$
 9:       if $L.empty = true$ then
10:         $N$ creates a new federation with itself as a leader.
11:       else
12:         Send $L$ to $N$.
13:       end if
14:     else
15:       Ask $N$ to modify its search criterion $(DI)$.
16:     end if
17:   end if
18: else
19:   Transfer a subscribe message to the leader.
20: end if
```

To leave the system, the node just sends a message "unsubscribe" to its leader. If the leader wants to leave, a replacement of a leader occurs by electing a new leader to preserve the federation's knowledge and integrity, and then the leader can unsubscribe. However, in this work, we will not discuss the election process, and we will limit ourselves by the construction of the layered architecture and the routing algorithm.

4.4.2 Semantic Query Processing Mechanism

The mechanism used in this work uses OntDD to semantically propagate the query between semantic interconnected federations. See algorithm 3. It decides where the query must be sent by using the different semantic relationships seen in Figure 4.2. This mechanism limits the query search space from the entire Grid to three spaces.

— Space 1: it represents the federation itself. In this space, the leader supports the search of resources in response to the query in its knowledge base;

— Space 2: it represents federations inside the MDF, where federations are linked by hierarchical relationships narrower/broader and they are members of the same MDF. These federations are likely capable of performing a query since they have close domains of interest as the leader who sent the query;

— Space 3: it represents federations related to the actual federation by the associative relationship Skos:related. In SKOS, an associative link between two concepts indicates that the two are inherently related, but that one is not in any way more general than the other; e.g., *Biometry* is related to *Statistics*. These federations are possibly capable of performing the query because their domains of interest are related to the leader who sent the query.

A request is submitted to a leader node from one of its workers or another leader node. The leader follows two behaviours depending on the source of the request. With its workers, it tries to find in its federation a worker node that can satisfy the query based on the leader's knowledge. If such a worker is not available, the leader sends the query to leaders' federations in its MDF, which are likely to satisfy the query. If it does not receive any response after a while, it forwards the query to its related leaders' federations as the last resort. If a leader is solicited, it tries to find workers that respond to the request; otherwise, it ignores the query. This strategy of semantic query processing reduces the search time and decreases the network traffic by minimising the number of messages exchanged between the nodes and federations.

4.5 Performance Evaluation

In this section, we discuss the performance of the previously presented resource discovery algorithms, by running the experiment on different network configurations. We focused on the efficiency of the system in terms of the number of messages, the time needed to complete a search and the metric of recall rate which is defined as the number of results returned divided by the number of results actually available in the network. We evaluate the performance of the resource discovery model with semantic and non-semantic cases. We ran series of simulations to evaluate and to compare the performance of the semantic and non-semantic resource discovery models. We used the parameters in table 4.1 for semantic and non-semantic experiments. The simulator constructs the leader-worker network, and simulates

Algorithm 3 Algorithm Handle_Query (Q): Q is sent to leader L from node n

```
    if $n$.isWorker = false then
2:      Find node(s) $N$ in the federation that satisfy $Q$.
        Return $N$ if it exists to the requester.
4:  else
        Find node(s) $N$ in the federation that satisfy $Q$.
6:      if $N$ is not empty then
            The search is succeeded; send a response to a requester.
8:      else
            Forward the query, direct $Q$ to leaders' in $MDF$.
10:         Wait responses for a time $T$.
            if $T = 0$ and no response then
12:             Forward the query, direct $Q$ to the related leaders'.
                Wait responses for a time $T'$.
14:             if $T' = 0$ and no response then
                    The search is failed;
16:             else
                    The search is succeeded; return a response to a requester.
18:             end if
            else
20:             The search is succeeded; return a response to a requester.
            end if
22:     end if
    end if
```

the behaviour of the resource discovery protocol in P2P Grid networks through the network topology generation. More clearly, this core element is used to generate the topology used in this work. It creates federations, MDFs and related federations. We have supposed a fixed number of MDFs. The simulator first generates the number of federations for each MDF, then it affects to them their domain of interest based on OntDD, after that it creates links between different related federations. For any new node, the simulator randomly generates a domain of interest from the existing domains and selects the first federation which it has to contact. At each join process, each federation follows the code in listing 4.2 by just replacing the domain of interest according to the domain of the new node. This simulator only supports the exact match queries, and the nodes join process was simplified in which any new node will find an adequate match with one of the existing federations;

The simulation parameters and their corresponding values, are reported in the table 4.1.

4.5.1 Performance and Discussion

We have used two strategies of evaluation based on the mechanism of query processing between leaders. The first one consists exactly of our proposed mechanism in this work, which means the leader looks first for the required

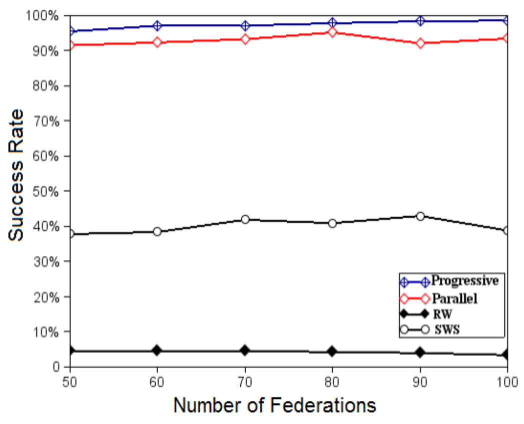
Network parameters	**Value**
Number of MajorDomainFederation	20
Number of federations' leaders **F**	50 to 100
Number of nodes per federation **P**	50 to 300 default 100
Network size (number of nodes (workers) and leaders)	$F * (P + 1)$
Maximum number of related federation per federation	0 to 5
Bandwidth size between a (worker/leader)	Random number in [4 : 6] Mbps
Bandwidth between leaders	Random number in [2 : 6] Mbps
Latency	Random number in [20 : 40] ms
Number of resources per node	1
Number of types of resources	1000
Number of class of resources	20
The transmission time depends on the size of the request, the bandwidth and the latency	$Trans\_Time = Latency + (Packets/bandwidth)$
Request processing (resource discovery)	Poisson distribution rate= 6.0
Mean query generation time	Exponential distribution rate=0.015 (2 requests per minute)
Timeout: waiting time	4 seconds
Time to live TTL for RW algorithm	20
TTL for SWS algorithm	5
Number of best neighbours for SWS	4

Table 4.1 – Simulation environment parameters for resource dicovery.

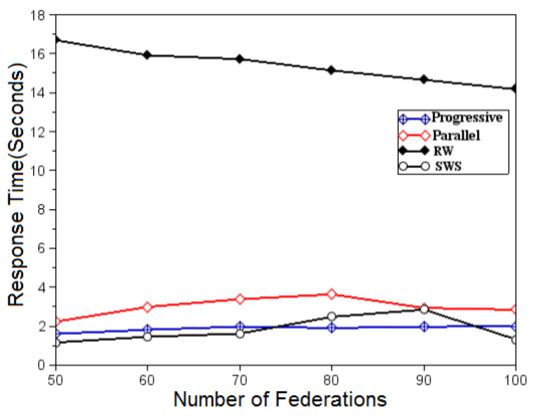
resource inside its own knowledge base, then if the search has failed, it sends a query to leaders in its MDF, after a time if it does not receive any response, it forwards the query to its related federations, we call this mechanism a progressive mechanism. In the second strategy, if the leader is not able to satisfy the request locally, it forwards the query directly to all its semantically interconnected leaders, the leaders inside its MDF and its related federations. We call this strategy a parallel mechanism.

For comparisons, we simulate our searching scheme in conjunction with a super-peer (SWS) model proposed by [Mas+05] and the random walk scheme (RW). SWS is a super-peer model; it was chosen because of its good query processing, but it does not use any semantic in its implementation. It will be very useful to demonstrate the effect of semantic on a resource discovery process. RW was chosen as a reference approach for its simplicity and prevalence, which, in fact, made it a widely used baseline for many previous research efforts. RW acts as follows: If the resource is not found locally, the node generates a request which is randomly forwarded to neighbouring nodes. To avoid unnecessary cycles in the network, nodes that are visited from a request are marked to avoid being re-visited in the future forward of the same request. This algorithm stops when the resource is found or if the TTL is reached.

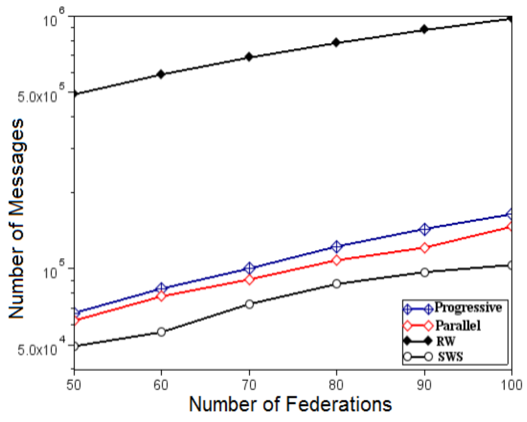
Figure 4.3 depicts the effect of the network size on performances of the proposed system by varying the number of federations that constitute the network.



(a) Success rate.



(b) Response time.



(c) Number of exchanged messages.

FIGURE 4.3 – Effect of the network size on performances.

(a) Represents the relationship between the success rate and the size of the system. We observe that both versions of our query processing algorithms based on semantic have the highest success rate compared to SWS and RW. The success rate remains almost steady with the growth of the network which means that this factor does not have an impact on our algorithms and a leader-worker in general. Although the progressive version outperforms the others and comparing it with the parallel version, it is supposed that more federations bring more resources, which implies a higher success rate, but it means also that a size of the MDF will grow too. Therefore, the number of the interconnected federations will increase at its turn. For this reason, in case of the parallel algorithm, the success rate decreases with the high size of MDF, because of the soliciting mechanism which sends queries to all the interconnected federations inside MDF and related federations at the same time. This makes a network more congested and delays queries as well as their responses, which affect a success rate since a request will be considered as failed if its response arrives after a timeout.

In (b), the response time changes a bit with the size of the network for both algorithms, but it remains almost steady between 1 and 4 seconds, except for the RW where it takes a long time before returning a response. This means that for our proposed scheme, the network size does not have a great effect on the response time because each leader serves only its own nodes and nodes belonging to its interconnected leaders. With the change of the network size, a leader may have more or less connections with the other leaders, which explains the change in response time. However, the progressive and SWS algorithms are faster than the parallel and the RW algorithms. When comparing the progressive algorithm with the parallel one, we find that the progressive is faster than the parallel where it is supposed to be the opposite, because of the graduation mechanism adopted by the progressive algorithm, which avoids generating unnecessary requests that help to decrease the delay of queries.

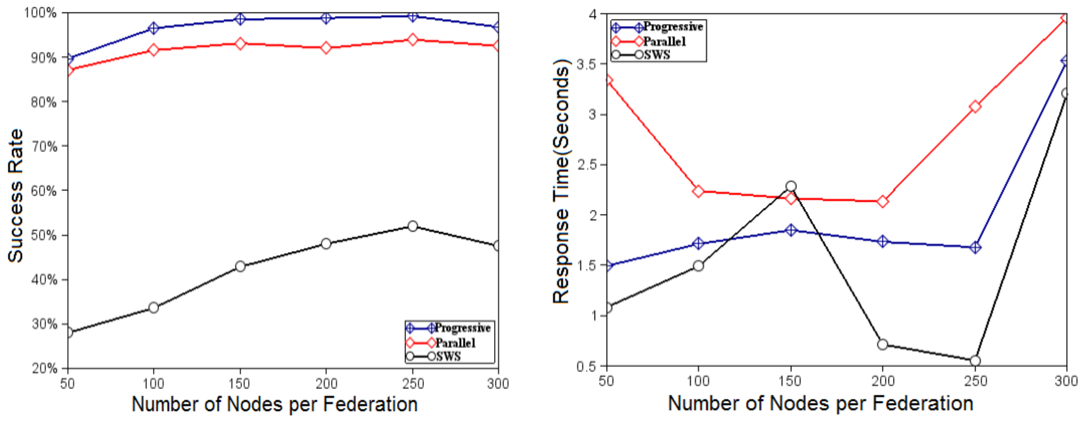
(c) Represents a change in the number of generated messages, including the returned response messages for the whole system according to the network size. We see that the number of messages is directly proportional to the network size which is expected. The SWS algorithm generates fewer messages than the other algorithms, but regarding the success rate of each algorithm; the progressive has the most important success rate which means an important amount of returned results, which explains its number of generated messages.

From this part of experiments, the progressive version of our query processing resolves queries faster than the others, and it locates more results. This indicates that our routing scheme based on SKOS architecture is more accurate and can always find the right side to forward the query to, it effectively reduces the search space, and its overlay architecture based on semantic guides the query in the correct direction. Therefore, it can locate results faster and more accurately.

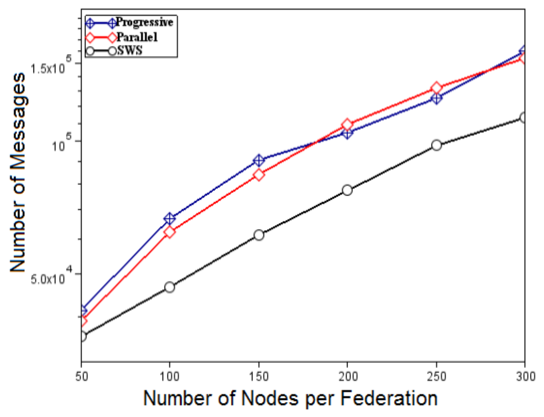
Figure 4.4 examines the effect of the size of the leader's group on performances of the proposed approach. We vary the number of nodes constructing each federation to test the scalability and the efficiency of the routing scheme, and we examine the behaviour of the system for the SWS, progressive and parallel strategies. (a) Represents the relationship between the success rate and size of the federation. The success rate increases with the number of nodes for both algorithms where SWS shows a lower performance compared to the others. The rate reaches its best value when the number of nodes attains 250. More nodes mean more resources, which implies a higher success rate; however, it means a huge number of queries too, which puts a leader or super-peer under stress and makes a network more congested. This can delay queries as well as their responses and affect a success rate for the progressive and the parallel algorithms since a request fails if its response arrives after a timeout.

(b) Shows the change in response time, for both algorithms. At the beginning, there are only a few nodes connecting to a leader or a super-peer which means a small number of requests and consequently, a fast response time. With the increase in the number of nodes, time increases a bit then it drops to the point where the number of nodes is between 200 and 250, this is because when a new node joins a system, it brings additional resources, which raises the possibility of finding resources more quickly. After that, the response time increases proportionally to the number of nodes, because as expected a high number of nodes generates a high number of request too, which delays the response time.

(c) Illustrates the relationship between the size of the leader's group and the number of generated messages. It is obvious that the number of generated



(a) Success rate. (b) Response time.



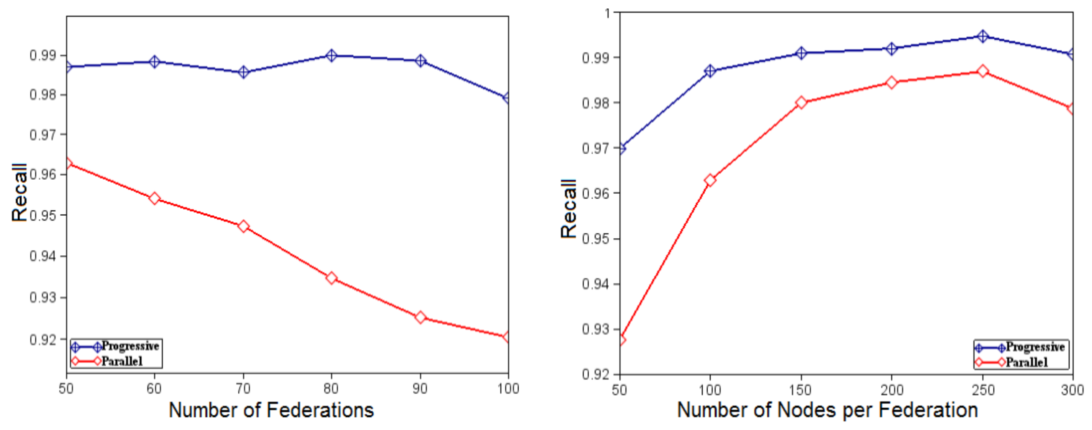
(c) Number of exchanged messages.

FIGURE 4.4 – Effect of the size of the leader's group on performances.

messages is proportional to the number of nodes constructing the federation because, with the growth of federation's size, the number of queries sent to the leader will also increase. We see that SWS generates fewer messages than the progressive and parallel strategies because SWS strategy has a low success rate thus a low number of returned results, which affects the global number of generated messages.

Figure 4.5 demonstrates the recall rate metric for both strategies of our query processing algorithm.(a) Outlines the recall metric according to the network size. The progressive algorithm outperforms the parallel one. It shows small changes where the recall decreases a bit with the increase in network size, which is natural because the network became more congested with the huge number of exchanged messages, the progressive algorithm is not very sensible in a large scale compared to the parallel one, and it performs better.

(b) Describes the recall rate according to the size of the leader's group. The recall increases with the number of nodes. It achieves its best rates when the number of nodes is in the range of 160 and 250, after that it begins to decrease



(a) Effect of network size on recall rate. (b) Effect of size of the leader's group on recall rate.

FIGURE 4.5 – Recall rate metric for progressive and parallel versions.

because the load on the leader became more important, which decreased its performances.

4.6 Summary

This chapter presented a semantic approach for resource discovery on P2P Grid since it is a very important challenge when the scale of the P2P Grid grows-up. Finding an effective and efficient way to organise nodes would, without a doubt, facilitate the discovery and resource queries of these nodes. We have first created a SKOS lightweight ontology to describe domains of applications in P2P2 Grids. Then, we presented a semantic approach of gathering nodes into groups called federations, using the SKOS ontology to construct a three-layered architecture. We have shown that the propagation of queries in this architecture is scalable and efficient since the search space is reduced from the entire Grid to a smaller range, consisting of three semantically related spaces in the worst case, which decreases the cost of the search. In addition, this architecture is scalable since it is based on OntDD ontology, which is flexible by its nature because it allows the addition of new federations easily.

To evaluate the performance of our algorithm with its both progressive and parallel versions, we have conducted extensive simulations, and we have compared the proposed algorithm with a super-peer model that does not use Semantic Web technologies, and the random walk algorithm. The experimental results have demonstrated the efficiency of the implemented system, both in terms of the number of messages, the time to complete the search and the recall rate.

Despite the valuable results obtained by the proposed approach, we have seen that number of nodes belonging to a leader had a significant influence on its performance: with a few numbers of nodes, the leader had efficiency issues, and the success rate was decreased, and most importantly, with more important numbers of nodes, the response time was decreased, thence the leader had suffered from scalability problems.

The next chapter will deal with these drawbacks, it will improve this approach by the development of advanced strategies to detect a potential bottleneck on leaders due to the important number of its workers, and to find away to keep the leader in a good state. This will enhance the scalability of each federation, at the same time, will improve the scalability of the whole system.

Chapter 5

Scalability-aware Mechanism based on Workload Prediction

5.1 Introduction

In this chapter, we present the second part of the thesis's contributions, which is a scalability-aware approach for ultra-peers networks. In large-scale distributed computing environments like P2P Grids, Ultra-peers overlay networks are very emergent because they offer several advantages. Since the system scales up rapidly and accommodates a dynamic change in the number of users, resources etc; controlling the workload on each ultra-peer is primordial because it can become a bottleneck on the ultra-peer at any time. Thence, it limits the scalability. Thus, developing an advanced model based on scalability aspects is a necessity, since it is an important and critical issue when designing such systems.

5.2 Problematic and Motivations

In recent years, scalability has become a factor of increasing importance in the design of distributed systems and organisations that are expected to grow rapidly. According to Neuman in [Neu94], a system is said to be scalable if it can handle additional users and resources without suffering a noticeable loss of performance or increase in the administrative complexity. In this case, the scalability issue has three dimensions: size, spatial, and management. The size dimension consists of the number of users and objects in the system. The spatial dimension means the distance over which the system is scattered. The management dimension consists of the number of organisations that exert control over pieces of the system [Neu94].

P2P Grid Computing paradigm is very popular for building distributed computing systems because of the advantages it offers. It can harness various computer resources including computation cycles, storage and bandwidth, and share information like files, audio, video; in addition to the communication and collaboration such as instant messaging [Li+08]. Ultra-peers networks have been proposed [Yan+03] to achieve a balance between the inherent efficiency of the centralised architectures, and the autonomy load-balancing and fault-tolerant features offered by the distributed ones [Mas+05]. They are a mixture between centralised and distributed models and take advantages of both of them. An ultra-peer is a node that acts as a centralised server to a subset of ordinary nodes.

Ordinary nodes submit queries to their ultra-peer and receive results from it. We refer to clients, users or any peers by nodes. Accordingly, an ultra-peer network is a network of a set of interconnected ultra-peers and the set of nodes connecting to each ultra-peer. These networks are considered among solutions that guarantee the efficiency and offer more scalability. Besides, they allow to several clusters of ultra-peers to interconnect and to work together which will increase the benefits to the users because more resources will be available in addition to a best exploitation of resources.

In this work, we give the following assumptions about the studied system:

— Ultra-peer maintains the information system of the cluster and up-to-date information on all the available resources of its cluster, and performs additional services like RD;

— Interconnected ultra-peers can collaborate among themselves, and nodes may benefit from using resources belonging to another interconnected ultra-peer;

— For each ultra-peer, the workload is defined by all the incoming requests for processing, from its environment, designated by its nodes and its interconnected ultra-peers;

— To be fault tolerant and for more reliability, each ultra-peer designates a node to be its recovery-peer.

— Each node connects to only one ultra-peer;

— It is worth to note that we do not limit this work by the overlay architecture seen in the previous chapter; instead, we do propose a generic solution for ultra-peers scalability that would be applied in conjunction with different types of routing and grouping algorithms.

Despite their benefits, these models suffer from scalability problems, due to the amount of the workload exercised on each ultra-peer. As the workload on the ultra-peer increases, it becomes busy and saturated leading to bottlenecks, and limiting the scalability for environments with a huge number of nodes.

For example, Gnutella, one of the most representative of these systems has one basic problem: when faced with a high query rate, nodes quickly become overloaded and the system ceases to function satisfactorily [Cha+03]. According to some important statistics of the Gnutella network given by Stutzbach in [Stu+08], an ultra-peer have interconnections to a maximum of 32 other ultra-peers, and to a maximum of 30 leaf peers where a leaf-peer connects to a maximum of 3 ultra-peers. The average number of neighbouring ultra-peer of an ultra-peer is 25, whereas the average number of neighbouring leaves is 22. Yet, another important investigation revealed that there were (30% to 38%) of discovered peers are unreachable because of several reasons: (2% to 3%) are departed peers, (15% to 24%) are firewalled, and the remaining unreachable peers (3% to 21%) are overwhelmed ultra-peers. Another important characteristic is that nodes in networks like Gnutella exhibit significant heterogeneity in terms of their capacity to handle queries [SAR+02].

In large-scale networks, and with regard to the importance of scalability for building and designing any competitive system, a scalability should be provided with respect to the aforementioned dimensions.

Thence, through the projection of the three dimensions of a scalable system as mentioned in [Neu94] on the ultra-peers network; we find that the scale of a network affects the performances of each ultra-peer, since the size dimension that represents the number of nodes of each cluster, affects the workload on ultra-peers and the amount of communications. Yet, the management dimension becomes more difficult and it is less practical for ultra-peers to serve all nodes, and to keep it up-to-date where the spatial dimension affects the communication latency.

From the impact of these three dimensions on ultra-peers, we could see that for this paradigm, the scalability is fundamentally a resource issue, we can determine it by whether each ultra-peer has enough resources to handle new nodes and their consequent queries and workload or not. It is faced with two principle issues:

— The first one consists of the excessive inter ultra-peers communications that consume bandwidth and introduce latency. This issue may be caused by poor routing algorithms and a poor grouping of nodes to construct the ultra-peers clusters. We refer to it by inter ultra-peers scalability. However, it can be resolved by efficient mechanisms of ultra-peers design that could be based on several strategies like locality preserving [She+12], Hilbert space [Moo+01], Semantic Web [Che+15] etc. and with the help of good routing algorithms. Many works have addressed this problem [Che+15; Kha+15; Llo+14; Bru+12; Pad+10; Jea+08; Nej+03a; Row+01; Sto+01];

— The second issue consists of the high density of nodes belonging to each ultra-peer and their engendered workload that produce insufficient local resources and may create a bottleneck on the ultra-peer; it is resumed on the local ultra-peer communication. Conversely, much of research in the field of ultra-peers has been directed to the technological issues mentioned in the previous point. Little attention is paid to the operational aspects of such systems, like [Li+08]. Hence, this problem remains the major drawback of the ultra-peers architectures.

In this section, we aim to study the behaviour of the workload on the ultra-peers networks and to propose solutions to guarantee their scalability. Therefore, we inspire from the dynamism of ecosystems that are considered as entities subject to disturbances and that are in the process of recovering from some past disturbance, when they are disposed to some sort of perturbation, they respond by moving away from their initial states. Despite the disturbance, these entities have the tendency to remain close to their equilibrium state [Cha+02]. We design and implement a dynamic scalability-aware model for ultra-peers networks where each ultra-peer can regulate and maintain itself. An ultra-peer represents an entity of the system, in which; its main workload come as requests from its environment [Rag+95]. Hence, it is subject to a change of the incoming requests' rate from its composing nodes and from the other entities. Once an ultra-peer encounters a bottleneck, following the proposed algorithms, it will be able to move from this state to another steady state.

For this end, we propose a process of prediction and decision as represented in Figure 5.2, in which the core component is the combination of the neural networks ANN, and the queueing theory [Tak62], specifically the Gelenbe [Gel75] blocking probability formula. This process integrates the Gelenbe formula with ANN, in which at each prediction cycle, the process is able to predict the future workload and to decide whether it will be manageable by the the ultra-peer or not. This will allow to each ultra-peer to inspect its future state and to maintain and regulate it conveniently. If the ultra-peer expects to receive an unmanageable workload causing a bottlenecks problems, it redirects an amount of the next incoming workload to another connecting node that functions as a recovery-peer. This strategy enables the ultra-peer to momentarily alleviate the workload on each ultra-peer, otherwise, the ultra-peer should apply the split process to divide its cluster and to create another new ultra-peer cluster. This strategy makes the proposed model very scalable and permits to add new ultra-peers' clusters to the system easily. The suggested solutions in this work are effective when the number of nodes increases that are suitable for both, very dynamic or more stable environments. In addition, they address the administrative information requirements when the system is growing, all with the compliance of the three dimension of a scalable system.

5.3 Workload Prediction based on Neural Network

This section highlights the process of prediction and decision making based on neural networks and queueing theory. This process is able to decide whether the future amount of workload will be a bottleneck on the ultra-peer or not. We will start by introducing the queueing theory and neural network models used in our work.

5.3.1 Queueing Theory

To study and model the ultra-peers networks, we use the queueing theory [Tak62]. Queueing theory has been successfully utilised for modelling and analysis of several real problems; like traffic engineering, customers in shops or banks, computer jobs waiting to receive a CPU time and so on. Similar to the work presented by [Ram+10] that evaluates the peer-to-peer file transfer latency in decentralised P2P systems and models each peer as an $M/G/1/K$ processor sharing queue to evaluate the peer processing delay. In the current study, we model each ultra-peer by an $M/G/1/K$ queue which in Kendall's notation means the arrival rate is Poisson, the service rate is general, one single server, while a system has a finite waiting queue capacity.

Even if the system can be seen as $G/G/1/K$ model because the ultra-peers networks are dynamic systems where nodes act independently, send several types of queries at unknown rates, and join and leave the system continuously. Yet, the rate of arriving messages from the interconnected ultra-peers is unknown too. In addition, the rate at which each ultra-peer can meet these requests and messages for processing is unknown since the requests service times can have any arbitrary distribution. But because the incoming requests rate to the ultra-peer will be the aggregation of the rate of all the requests coming from a large number of the independent connecting nodes, the arrival rate to the ultra-peer will be a Poisson as stated in [Alm14; Gel+10; Gel90]. The superposition of a large number of independent arrivals processes is approximately Poisson even if the individual components are not. This explains why Poisson arrival processes are frequently observed in practice.

The ultra-peer will suffer from a bottleneck if the amount of the incoming requests surpass its queue length, so if the ultra-peer can measure the blocking on its queue, it will be able to control the bottleneck. Numerous approximations for the blocking probability for $M/G/1/K$ systems exist. One survey article [Spr+91] that analyses in details five different approximations formulas, concludes that the formula by Gelenbe is the most accurate, robust and efficient for the majority of the cases tested. The same conclusion was obtained by [Mac04]. For this reason, we will use the Gelenbe formula to measure the blocking on the queue of each ultra-peer.

The Gelenbe Formula for $M/G/1/K$ is based on approximating the discrete queueing process as a continuous diffusion process. The blocking probability from Gelenbe equation 5.1 with squared arrival and service process coefficient of variation is given by the following equation [Gel75] where a^2 and s^2 are respectively the squared coefficient of variations of the arrival and service processes:

$$P_k = \frac{\lambda(\mu - \lambda)e^{-2((\mu-\lambda)(k-1)(\lambda a^2+\mu s^2))}}{\mu^2 - \lambda^2 e^{-2((\mu-\lambda)(k-1)(\lambda a^2+\mu s^2))}} \tag{5.1}$$

Where: a^2 is the squared coefficient of variation of the arrival process.
λ^2 is the external Poisson arrival rate.
μ^2 is the mean service rate.
k is the buffer capacity at the ultra-peer, including those in service.
P_k is the blocking probability of a finite queue of size K.
s^2 is the squared coefficient of variation of the service process.

Taking both the squared coefficients of variation of the inter-arrival time and the service time equal to 1, $a^2 = s^2 = 1$, equation 5.1 results in a Markovian single queues, $M/M/1/K$. Gelenbe's expression is also accurate for Markovian system, while it is not accurate for deterministic service time systems $M/D/1/K$, As noticed by [Smi+05; Spr+91].

5.3.2 Neural Network

In spite of the previous formula that simplifies the original model, rates of incoming requests in such dynamic system are still an important issue that is very hard to anticipate its values. For these reasons, this theoretical formula cannot be applied alone; instead, it has to be accompanied with other sophisticated techniques that would be capable of approximating the true value of the incoming rates of packets to the ultra-peer in question.

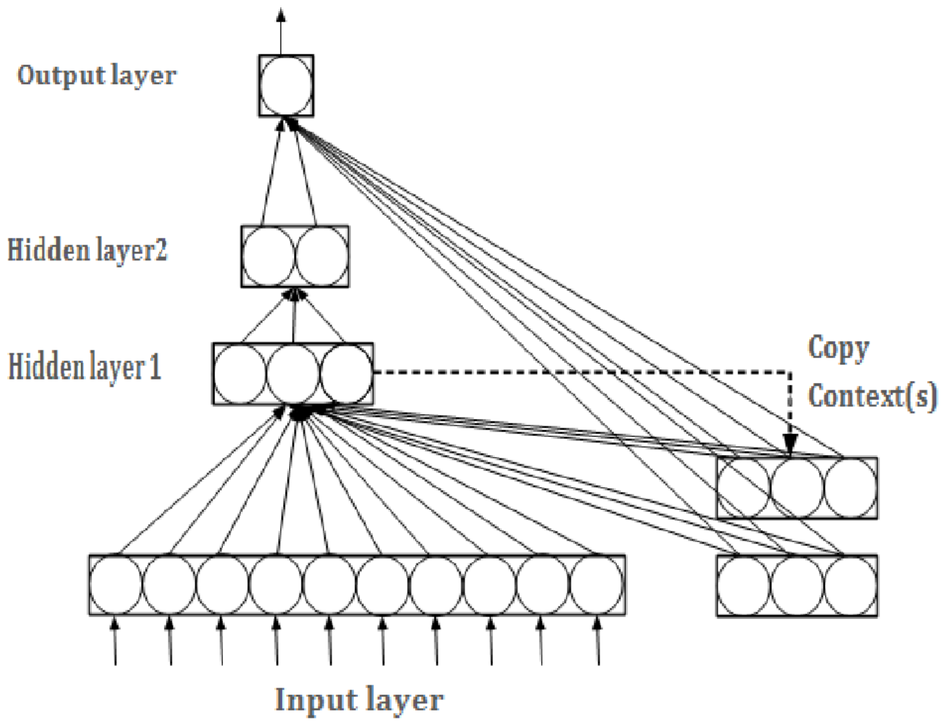


FIGURE 5.1 – The Multi-Context Recurrent Neural Network.

To this end, we call for an experimental study, neural networks. Because whatever the real model followed by the ultra-peer, a neural network will be able to model it in a fuzzy way. We use this network to predict the amount of the next workload, and this will represent the arrival rate for the previous formula.

Artificial Neural Networks (ANN) have been successfully applied to the prediction problems because of their ability to map, in a fuzzy way, inputs to outputs. They provide several distinguishing features which make them valuable and attractive for forecasting tasks. These networks are very suitable for problems that need knowledge which are difficult to specify but there are sufficient amount of observations. ANNs have the ability to learn from the observations data, and to infer the unseen part of the population. As forecasting is performed via prediction of future behaviour (the unseen part) from observations of past behaviour, it is an ideal application area for neural networks [Zha+98], and it is the perfect tool for our purpose too, to predict the future amount of workload from observations of the past behaviour of the ultra-peer.

In this investigation, we use a specific type of ANNs, The Multi-Context Recurrent Network ($MCRN$) [Hua+06], because it is characterised by the fact that the multi-context layers are directly linked to the output layer, like it is displayed in Figure 5.1; leading to a more complex connectivity between layers; but, it reduces the dependency of the network output on the hidden layer and speeding up the training [Hua+06].

5.3.3 Process of Prediction/ Decision based on Neural Networks and Gelenbe's Formula

To predict the future workload and to study the behaviour of the ultra-peer, we suggest the process of prediction/decision as shown in Figure 5.2. The core component of this process is the ANN and the Gelenbe blocking formula. This process bridges the Gelenbe formula with the ANN where at each prediction cycle, the process is able to predict the future workload and to decide whether it will be a bottleneck on the ultra-peer or not.

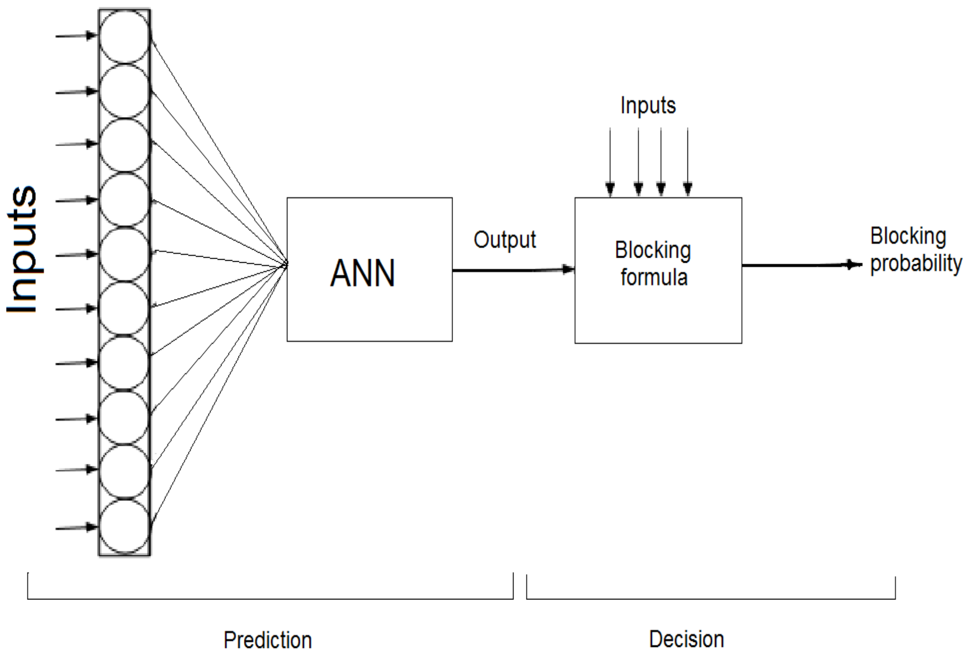


FIGURE 5.2 – The prediction / decision system scheme.

For this end, we have implemented the *MCRN* neural network (Figure 5.1), to first, predict the future amount of the workload based on several parameters represented by the ANN input layer and which contains 10 inputs; including:

— Status of days: beginning of the week, working days and weekends;
— Status of hours: morning, night, peak hours, afternoon;
— Number of nodes connecting to a ultra-peer;
— Number of the inter-connected ultra-peers;
— Past workload.

Then, the predicted workload which will be estimated in number of packets; will be considered as the arrival rate for the Gelenbe formula to estimate the blocking value; hence, to make a decision if it causes a bottleneck or not. For this purpose, we have slightly adapted the original formula 5.1, because we take into account number of packets rather than number of requests, so the $(k-1)$ will be replaced by $(k-n-al)$, see equation 5.2.

$$P_k = \frac{\lambda(\mu-\lambda)e^{-2((\mu-\lambda)(k-n-al)(\lambda a^2+\mu s^2))}}{\mu^2-\lambda^2 e^{-2((\mu-\lambda)(k-n-al)(\lambda a^2+\mu s^2))}}, \tag{5.2}$$

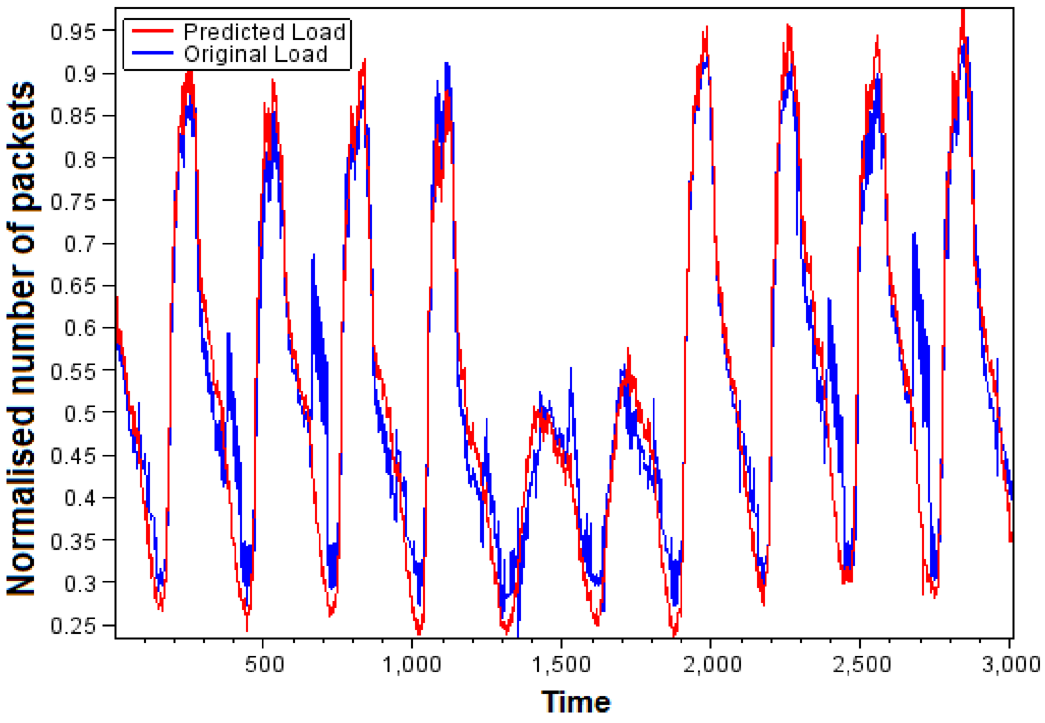


FIGURE 5.3 – original and predicted workload.

In which:
λ: Will be the predicted workload, in number of packets.
μ: The mean number of processed packets.
al: The actual load, it represents packets actually in the queue of the ultra-peer.
n: Number of packets of requests actually in processing.
k: The queue length of each ultra-peer.
P_k: The blocking probability, out put of the activation function, it will take values between 0 and 1.

The optimum network parameters such as a structure, number of neuron's in hidden layers, transfer functions, have been varied to find the best network parameters. The back propagation algorithm is used to learn the network where the performances of training and validation are evaluated by computing the Mean Square Error (MSE). Also, the cross-validation is used to test and check the predicting quality.

Figure 5.3 displays the predicted workload using the MCRN network. The data of the original workload are generated by simulations, and are collected with a time step of 5 seconds.

5.4 The process of State Control

In this section, we present a solution that allows each ultra-peer to control its state and to resolve its scalability problems by proposing a set of actions and an algorithm that combines all the above-mentioned theories. Prior to this, we will highlight again the problem of scalability from another point of view.

It is obvious that the number of nodes connecting to the ultra-peer strongly affects its performance, especially its individual load and individual processing, leading to a scalability problem and a bottleneck situation. Because nodes are the most important source of incoming traffic, it is very important to keep the number of nodes reasonable to the capacity of the ultra-peer, and the latter must strike balance between its local efficiency and scalability and the global scalability. A big number of nodes will cause a heavy workload which will limit the local scalability and will be a bottleneck. However, a small number of nodes will degrade the efficiency and the performance of a subsystem represented by the ultra-peer cluster. Since nodes are a source of resources too, a small number of nodes mean a small set of resources which decreases the local success rate in one hand. And on the other hand, by fault of its internal efficiency poverty, the ultra-peer will be more dependent on its interconnected ultra-peers in order to satisfy its local queries. Therefore, the communication between ultra-peers will increase and will cause a network congestion that will limit the inter scalability.

We propose a solution based on P_k which represents the probability of a bottleneck on the ultra-peer which is obtained by the process of prediction and decision described above, and by the actual status of the ultra-peer. Algorithm 4, explains how the ultra-peer takes decisions about its status according to P_k, it will first take a decision regarding the amount of the future workload, then to the number of nodes forming the cluster.

5.4.1 The state control algorithm

The Algorithm 4 uses the following parameters:

— Checklist: represents a list of adjacent ultra-peers which are interconnected with the ultra-peer in question, and it is the routing table between the interconnected ultra-peers;

— Critical: represents the situation where the state of the ultra-peer is critical where its local scalability is under question;

— $Safety$: represents the situation where the ultra-peer is in a safety state;

— T: refers to a time during which the system starts and remains on a critical situation;

— σ: represents a time threshold during which the critical situation persists;

— γ: represents a time threshold during which the ultra-peer redirects the workload to its recovery-peer;

— ϵ: represents the threshold for the blocking probability;

— *State*: it can take as a value "*critical*" if the system is in a critical state or "$safety$" if the system is in a safety state;

— *allowedRate*: a threshold for the allowed rate of blocking packets compared to the total incoming requests during the critical situation.

Algorithm 4 CheckState (ultra-peer X)

```
1: if $P_k \leq \epsilon$ then
2:    $T \leftarrow 0$
3:    $State \leftarrow safety$
4: else {*the beginning of a critical situation*}
5:    Calculate time $T$ during which the system remains in this situation.
6:    $State \leftarrow critical$
7:    if $T \prec \sigma$ then
8:       /*may be it is a temporary situation, a peak load period*/
9:       Delay any new demand of join.
10:      Temporary suspension of receiving requests from the inter-connected
         ultra-peers.
11:   else {*the critical situation persists*}
12:      if $BlockingRate \prec allowedRate$ then
13:         Extend the previous state for another $\sigma$ of time.
14:      else {* the blocking rate exceeds the allowed rate of blocking*}
15:         Redirect a workload to the recovery-peer.
16:         if $T \geq \gamma$ and $State = critical$ then
17:            Split the ultra-peer's cluster.
18:         end if
19:      end if
20:   end if
21: end if
```

In order to scale well with the growth of the network, each ultra-peer has to control its internal state in which the state of the ultra-peer depends on P_k. In this study, we refer by blocked packets to all the lost packets.

The blocking packets are generally a sign of congestion and bottlenecks. A considerable value of blocking packets indicates a bottleneck on that ultra-peer. An ultra-peer, following the Algorithm 4, continually estimates its P_k, according to its value where the ultra-peer may change its state from safety to be critical. Meaning that the blocking probability becomes very high and that the ultra-peer will be a bottleneck. In this situation, the first decision taken by the ultra-peer is to delay the joining of any new nodes and to temporarily suspend any solicitation from its interconnected ultra-peers. This temporary suspension may have an impact on the system efficiency; but, it stays a temporal impact, minor, neglected and can not surpass the effect if the whole ultra-peer cluster will break-down because of the bottleneck.

After that, the ultra-peer has to wait for an acceptable period of time σ to check if it can manage this situation in order to relieve the pressure on it, hoping to be a temporary situation causing by a peak load period. Meanwhile, it calculates its rate of blocking packets during this period, and checks if it exceeds or not the acceptable threshold *allowedRate*. According to the blocking rate, it decides whether to extend this situation for another period of time. This means that the rate of blocking is neglected and that the first decision helps the ultra-peer to manage the situation; otherwise, it calls for the second decision with a regard to the rescue action.

The rescue consists of redirecting an amount of the workload to a recovery-peer, which will help the ultra-peer in the processing of all the incoming requests. The redirection of the workload will remain for a period of time equal to γ, during which, if the ultra-peer recovers its $safety$ state, the rescue process will be stopped. Otherwise, it means that the ultra-peer is no more capable of serving all its interconnected nodes, therefore, the ultra-peer must apply the last decision, regarding the split process to divide its cluster and to create another cluster, then it starts a new life cycle.

5.4.2 The split process

The split process is responsible for the creation of a new ultra-peer cluster from the actual one, and its integration within the network with the other interconnected ultra-peers. This process permits to support the horizontal scale of the environment. It has to guarantee the following heuristics:

— The recovery-peer will be the first ultra-peer for the new cluster, and will have the same characteristics and Checklist of the old one. The checklist must be updated to inform all the interconnected ultra-peers about the new cluster;

— Each ultra-peer will need to designate its recovery-peer;

— The partition of the nodes must be conformed to the criterion used for clustering. For example; if the grouping process was based on semantic and similarities of interests or resources between the ultra-peer and nodes, the partitioning should follow the same strategy. In this work, and for a matter of simplicity, we do split each group by the half. As we have already said, each ultra-peer may follow a strategy to decide which peer has to quit the actual group and to join the new one, like the locality preserving;

— For the migration from one group to the other, the ultra-peer of the old group communicates peers that they have to leave the group and ask them to establish the connection with the new ultra-peer.

5.5 Experimental Results

This section presents the simulation environment and parameters used to evaluate the proposed strategies. Then it presents a study of the complexity of the proposed algorithms. Without loss of generality, we mention that the ultra-peers topology used in this work follows the ordinary topology described above where each cluster is managed by an ultra-peer and consisted of a set of nodes that send queries to a ultra-peer. Ultra-peers are interconnected to each other to form a network of ultra-peers.

5.5.1 Performances Evaluation and Simulation Parameters

To evaluate the performance of the proposed approach, we first conduct extensive computer simulations using a synthetic heavy workload, then we examine again the performances under a workload built from a real access traces of ClarkNet [1].

For both workloads, the performance evaluations will focus on two performances metrics:

— Number of nodes actually connected to the ultra-peer;
— The amount of blocked packets.

Simulation parameters used in this part are detailed in the table 5.1.

In this experiment, we consider the number of received packets when estimating the workload. In addition, some values of parameters of simulations like σ and γ are arbitrary set because they have an evident effect on the performances, such as; if σ that represents the time threshold during which the critical situation persists, has a high value so the ultra-peer will wait for more time before it begins the redirection of workload, hence the number of blocked packets will increase too. Other parameters, like the rate of prediction, the squared coefficients of variation and especially the critical one ϵ have been selected after a set of experiments. We have chosen the values of ϵ according to the obtained value of the blocking probability and the real number of blocked packets; we have selected the three most representative and accurate values to be set as thresholds for blocking. Where the squared coefficient of variation for the arrival process is set to $a^2 = 1$ because the system is an $M/G/1/K$ and the superposition of arrivals is Poisson, and the squared coefficient of variation for the service process s^2 was selected in the same way like ϵ. It is worth to notice that the investigation of the impact of the squared coefficients of variations on the blocking probability is out of the scope of this papers, and it was already studied by [Smi+05].

5.5.1.1 Synthetic workload

To evaluate the performance of the proposed approach, we conducted extensive computer simulations, we have undergone the ultra-peer under a very heavy workload to show its behaviour.

Figure 5.4, and Figure 5.5 depicts respectively, the effect of the rate of prediction on the number of blocked packets, and on the mean number of nodes connecting to a ultra-peer, during one week. We see that the mean number of nodes is almost the same for all experiments, where it reaches its maximum value for a time interval equal to 5, 30 and 60 minutes. On the other hand, we observe that number of blocked packets differs from experiment to the other, it reaches its lowest value for a time interval equal to 5 minutes then it starts to increase proportionally with the time interval. From these experiments, we conclude that the interval time of prediction has an effect on the prediction consistency, if the interval is very short, like in case of 1 minute, there will be an excessive predictions

1. http://ita.ee.lbl.gov/html/contrib/ClarkNet-HTTP.html

Simulation parameters	**Value**
Number of nodes per ultra-peer (UP) **P**	300
Bandwidth size between a node and its UP	Random number $\in [2, 6]$ Mbps
Bandwidth between two interconnected UPs	Random number $\in [4, 6]$ Mbps
Latency	Random number $\in [20, 40]$ ms
Number of resources/files per node	1
Number of types of resources/files per cluster	5-10
The transmission time depends on the size of the request, the bandwidth and the latency	$Trans\_Time =$ $Latency + (Packets/bandwidth)$
Prediction process	Every 5 minutes
Request arrivals rate	Non-homogeneous Poisson process with rates changing every hour
Service rates	normal distribution with $\mu \in [0.005, 0.1]$ and standard deviation $= 0.5$
Requests size	1 to 64 packets
Queue lengths K	1000 packets
$\epsilon \in (0, 1)$	$\epsilon \in (0.01, 0.03, 0.05)$
σ, a time threshold during which the critical situation persists	1 hour
γ	4 hours
allowedRate	1%

Table 5.1 – Simulation environment parameters for workload prediction and decision.

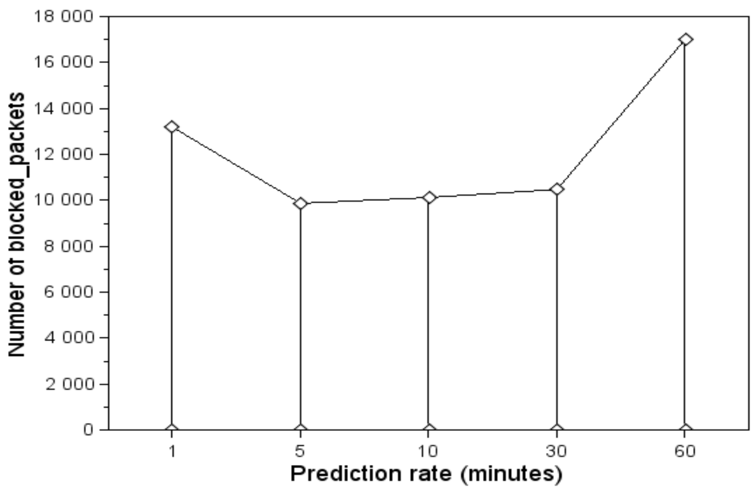


Figure 5.4 – Effect of prediction rate on number of blocked packets.

where there is no significant change in the state of a ultra-peer, and if it is long, the ultra-peer may enter into a critical situation for a long time before detecting it, which explains the high number of blocked packets for long-term predictions.

For the rest of the experiments, the measures will be taken at the interval of 5 minutes, for the five working days (excluding week-ends) in order to reduce simulation time.

Figure 5.6 depicts the change in the number of nodes over the time, for

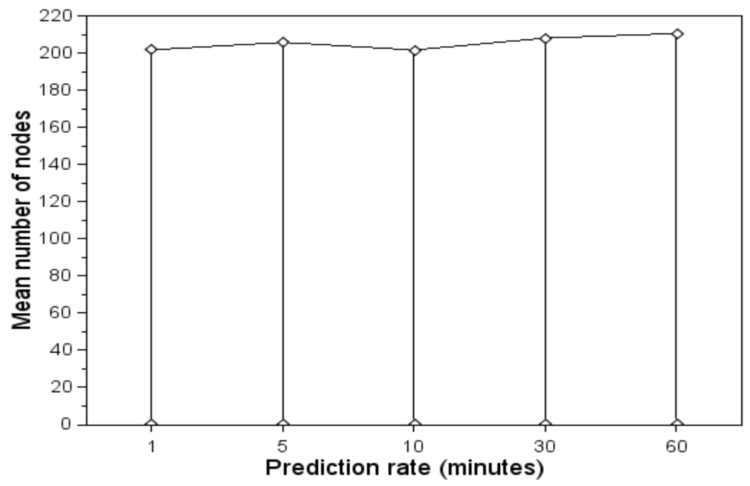


FIGURE 5.5 – Effect of prediction rate on number of nodes.

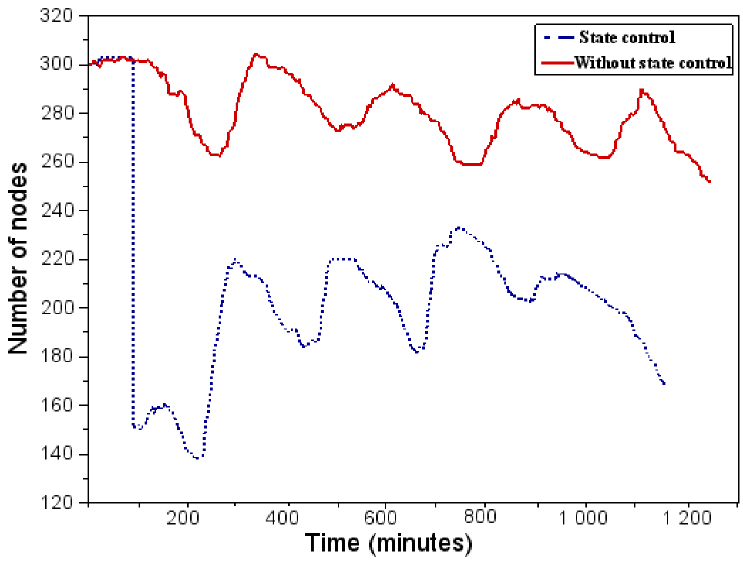


FIGURE 5.6 – The change in number of nodes in the proposed scheme and an ordinary scheme.

two models of ultra-peers. The first one follows a general scenario of ultra-peers while the other uses our proposed state control strategy. At the beginning, the two models start with 300 nodes in each; for the first ultra-peer, the number of nodes presents a small change over time; it's averaged between 260 and 320, then it dropped when time coincides with the beginning of the week-end where it is expected that the number of connecting nodes will decrease.

In contrast, in the proposed scheme which employs an intelligent bottleneck prediction mechanism, the number of nodes decreased by half for the first time, which means that the ultra-peer has detected that the actual number of nodes causes a bottleneck problem, after that the changing rate of the number of nodes remains almost steady, averaged between 180 and 230 over periods of days and nights. Then the number of nodes decreases with the beginning of the week-end.

Figure 5.7 indicates the change in the number of blocked packets over time, (a) for the scheme without state control and (b) a scheme with state control.

The two models start with 300 nodes. We can see that the number of blocked packets is extremely high for the model (a), which is a sign of a serious bottleneck; it increases over 2000 blocked packets in the busy hours and decreases to 100 blocked packets where it is expected to have low incoming requests. On the other hand, in the proposed model (b), there are some blocked packets at the first time, then it is dropped to zero; this means that the ultra-peer detects a bottleneck and subsequently it resolves it. We can see too that there are other few blocked packets, which are neglected and do not exceed a 200 blocked packets because it occurs during the peak load periods over the week, and because our proposed algorithm allows for a rate of 1% of blocked packets to occur.

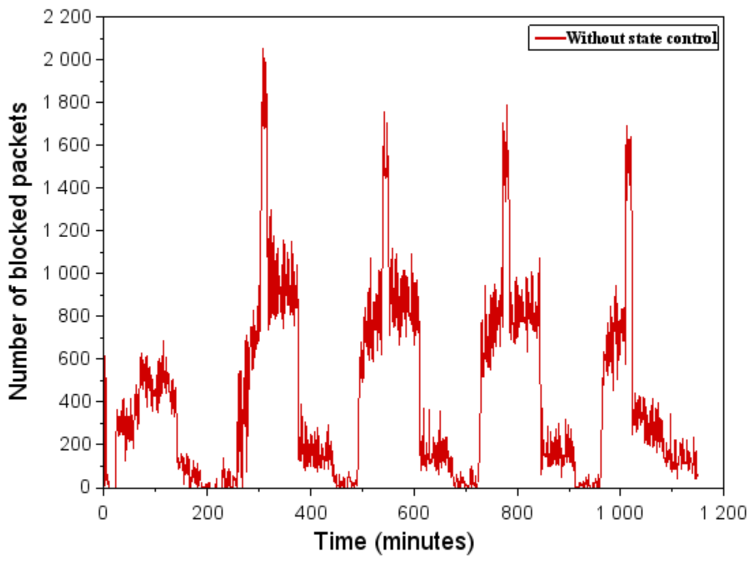
We can confirm that the proposed scheme is able to scale very well with the workload changes, by dynamically controlling the incoming requests based on bottleneck anticipation to avoid it in advance, where this is difficult for an ordinary ultra-peer.

Figure 5.8 illustrates the effect of the blocking probability threshold on the number of nodes constructing the ultra-peer of the proposed model. At the beginning, the three models present almost the same behaviour; number of nodes dropped from 300 to 150. This means that the three models have detected a congestion, and they apply a split process as a last resort, after delaying all the inter-ultra-peers communications, and after the redirection of an amount of the workload to the recovery node. But, we can see that the split happened at different time interval, for models with $\epsilon = 0.01$ and $\epsilon = 0.03$, they apply the split at the point near to 100, where for the model of $\epsilon = 0.05$, it uses a split at the point 200. After that the number of nodes decreases a bit for all models because it coincides with the end of the day, then it increases and decreases according to a period of the day.

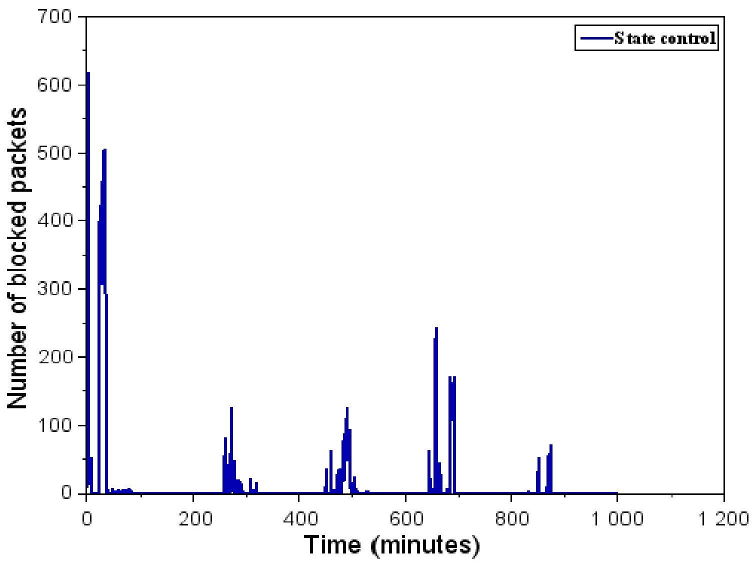
Contrary to the two other models, the model with $\epsilon = 0.01$ has applied the split process for only one time, this means that it was able to avoid bottlenecks without appeal to the split process again, and only with the temporary suspension of its inter-ultra-peers communications and the workload redirection to its recovery-peer. However, models with $\epsilon = 0.3$ and $\epsilon = 0.05$ have split their clusters again, at points 950 and 750 respectively, this means that there were existed other periods of bottlenecks, and they were not able to get out of the bottlenecks situations without appealing to the split process again.

This implies that the considerably higher values of the blocking threshold have an impact on the performances, and only the first model has successfully surpassed these periods without re-splitting its cluster again, by using only the delay and the redirection actions. Which means that the higher value of the threshold does not allow the other models to detect the congestion at a time.

By projecting Figure 5.9 which shows the effect of the blocking threshold on the number of blocked packets, on Figure 5.8 we observe the presence of blocked packets for the three models at the first time, as a result of a bottleneck caused by the high number of node and their consequent heavy workloads, for this reason ultra-peers split their clusters. We can see too that the number of blocked packets of the third model where $\epsilon = 0.05$ still considerably high compared to the two other models. Meanwhile, there are just a few blocked packets for the model with $\epsilon = 0.01$ which occurs during the busy hours, knowing that, this model applies



(a) The ordinary scheme without state control.



(b) The proposed scheme with state control.

FIGURE 5.7 – The change in number of blocked packets in the proposed scheme and an ordinary scheme.

only one split, and the number of its nodes is higher than the number of nodes of the two other models.

Figure 5.10 and Figure 5.11 present the relation between the number of nodes, and number blocked packets for respectively $\epsilon = 0.01$ and $\epsilon = 0.05$. From both figures, we see that the number of blocked packets is directly proportional to the number of nodes, and it is expected to increase during the peak load periods. When the number of nodes is equal to 300, we can observe that the number of blocked packets is significantly high, which a sign of a critical situation. This situation remains for almost 7 hours for the model with $\epsilon = 0.01$, and 12 hours for the other, before that the ultra-peer decides to split the cluster. Because, each

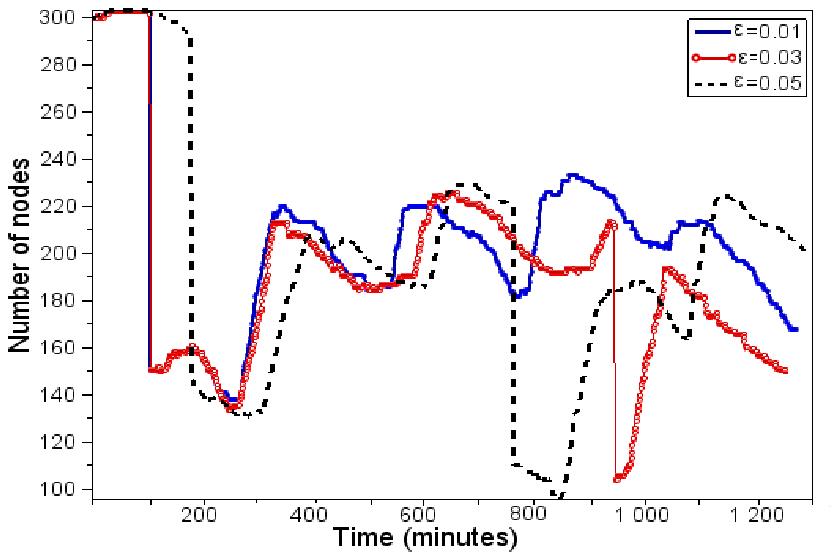


FIGURE 5.8 – Effect of the blocking threshold on the number of nodes.

ultra-peer try first to handle its critical situation with the delay and the recovery actions, hoping to be just a temporary and manageable situation.

For the figure 5.10 there are only a few blocked packets, happening at the peak workload hours, and due to the allowed blocking rate "1%". In addition, at the point 400, the mean number of nodes in this model is significantly higher than the model $\epsilon = 0.05$, so the number of requests to the ultra-peer will be higher too. Where in Figure 5.11, there are blocked packets in each busy hour; this means that the blocking probability threshold value does not allow the ultra-peer to control its state properly.

This proof that the strategy of state control works very skillfully, and that the ultra-peer can balance between its capacity and the number of nodes connecting to it, to scale well and to guarantee a good quality of its services.

5.5.1.2 Realistic workload

In this section, we evaluate our proposed solution using a realistic workload created from traces of real HTTP requests to a large web server system ClarkNet. ClarkNet is a commercial Internet service provider in the Baltimore (Washington D.C.) region. Its log contains 3328587 requests collected over a period of two weeks between August and September 1995, with an average request per day equal to 237756. Traces are in form of access logs consist of records collected on instant basis, each request contains a set of fields like the host making the request, the timestamps in the format of (DAY MON DD HH:MM:SS YYYY), etc. In this experiment, we make use of logs of only the first week, collected from (00 : 00 : 00) to (23 : 59 : 59) over a period of one week (28-08-1995 to 03-09-1995). Figure 5.12 shows the requests rates and the number of nodes (clients) connected to the server.

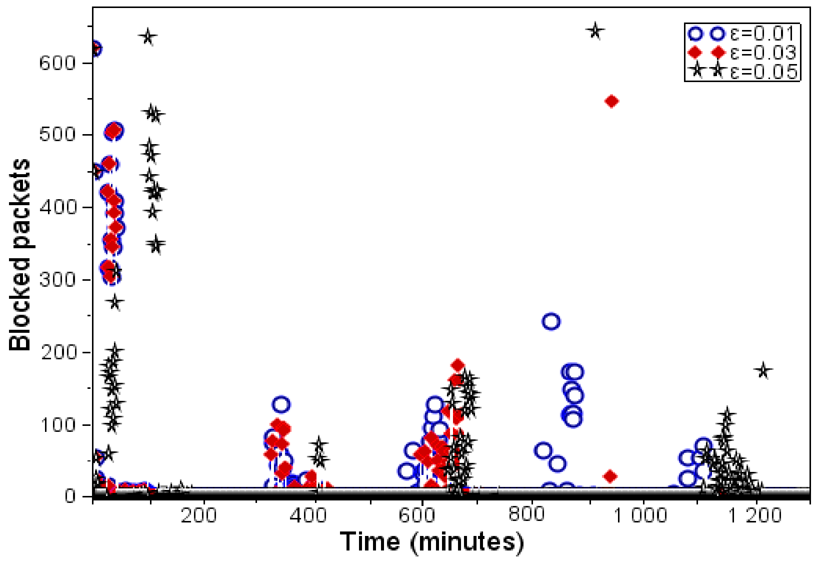


FIGURE 5.9 – Effect of the blocking threshold on the number of blocked packets.

The realistic workload was built by fitting the distribution of requests inter-arrivals rate using *StatGraphics*[2] tool, according to the trace, requests arrivals rate change every hour and they best fitted by the exponential distribution. Figure 5.13 demonstrates the fitting distribution.

For these cases of experiments, we set the queue size to 3000 packets and the blocking threshold to 0.05, 0.1 and 0.2. In order to test the effectiveness of our proposed solution, we activate the Check-State process after 12 hours of running, then we start recording and taking measures every 5 minutes because according to the benchmark, the ultra-peer starts with a small number of connecting nodes which will not cause any bottlenecks. This delay will allow us to get a more important number of connected nodes, and to test their impacts on the ultra-peer. Otherwise, they will be rejected by the Check-State process.

The experimentation shows a great similarity of the behaviour of the ultra-peer against the heavy workload, with results obtained by the synthetic workload. Figure 5.14 shows the effect of the blocking probability threshold on the number of nodes connecting to the ultra-peer. The number of nodes dropped from near to 1300 to 650 for the three models, this means that each of them has detected a congestion, then it has redirected an amount of the workload to the recovery node, then it has applied the split process. we can see that models with $\epsilon = 0.05$ and $\epsilon = 0.1$ have applied the split process only for one time, where the model with $\epsilon = 0.2$ has invoked the split process for another time at the point 950. for the rest, the number of nodes decreases for all models because it coincides with the end of the day, then it increases and decreases according to the period of the day.

Figure 5.15 which illustrates the effect of the blocking threshold on the number of blocked packets. At the first time, we can see the existence of a high number of blocked packets for the three models, because of the high number of nodes and their engendered workloads. Once the ultra-peers had split their

2. http://www.statgraphics.com/

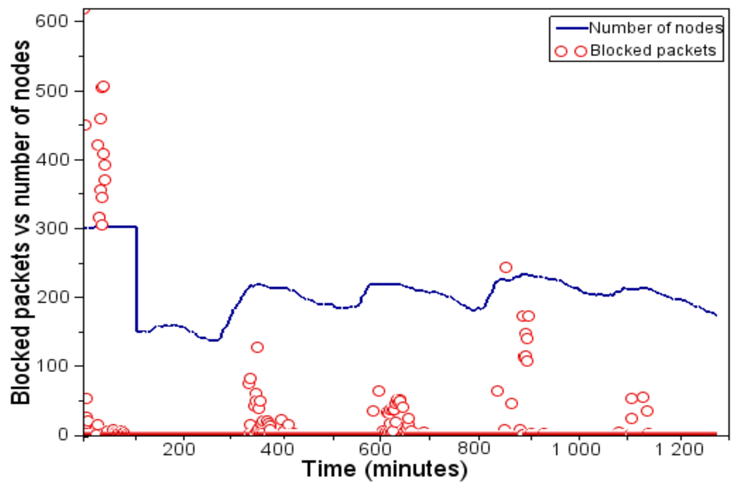


FIGURE 5.10 – Relation between number of nodes and blocked packets for $\epsilon = 0.01$.

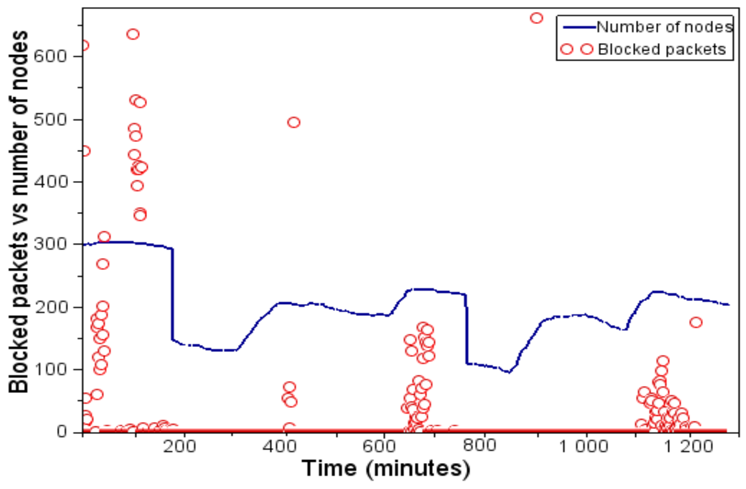
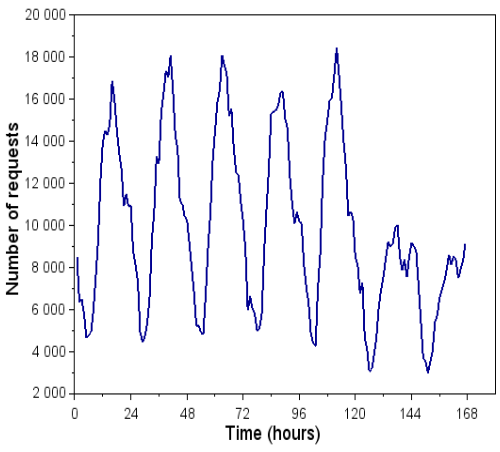


FIGURE 5.11 – Relation between number of nodes and blocked packets for $\epsilon = 0.05$.

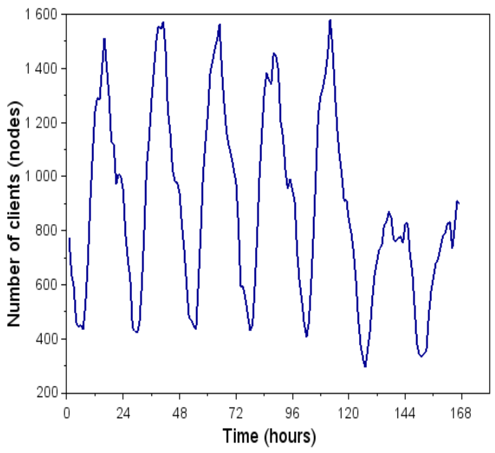
clusters the amount of the blocked packets was diminished. For the rest of the time, there were blocked packets at each peak workload period which is acceptable and because our algorithm allows for a blocking rate of "1%" to happen. On the other hand, the model with $\epsilon = 0.2$ has recorded the highest rate of blocked packets over all time of experiments, because the high value of the blocking threshold does not allow to the ultra-peer to effectively control its state, leading to another split process to avoid the bottleneck and the high number of blocked packets.

This implies that the considerably higher values of the blocking threshold have an impact on the performances.

Figure 5.16 and Figure 5.17 depict the relation between the number of nodes, and number blocked packets for respectively $\epsilon = 0.05$ and $\epsilon = 0.2$. From the two models, we see that the number of blocked packets is directly proportional to the number of nodes, and it increases during the peak load periods. When the number of nodes is near to 1300, we can observe that the number of blocked packets is significantly high, which is a sign of a critical situation, this situation has

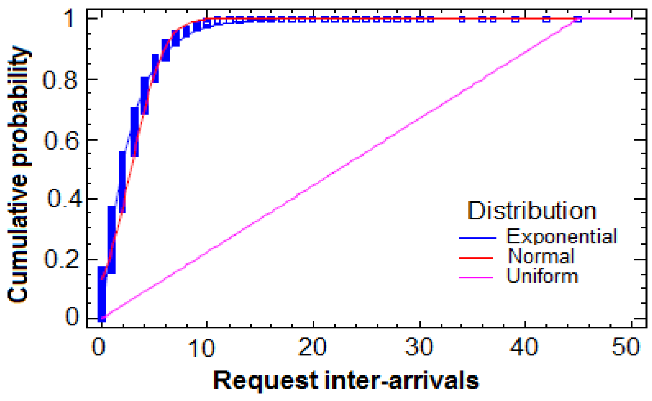


(a) Requests arrivals rate.

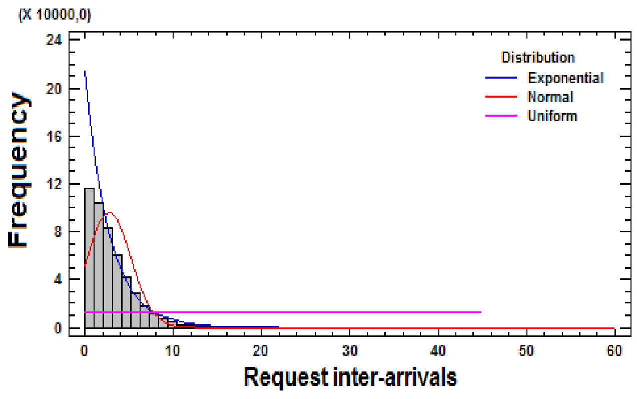


(b) Number of clients

FIGURE 5.12 – Excerpt from real workload traces.



(a) The Quantile plot of request inter-arrivals.



(b) The histogram of requests inter-arrivals.

FIGURE 5.13 – The fitting distribution.

remained for a while in which each ultra-peer has tried to manage the situation by delaying any new addition of new nodes first, then the invocation of the recovery action, hoping to be just a temporary and manageable situation. Because the

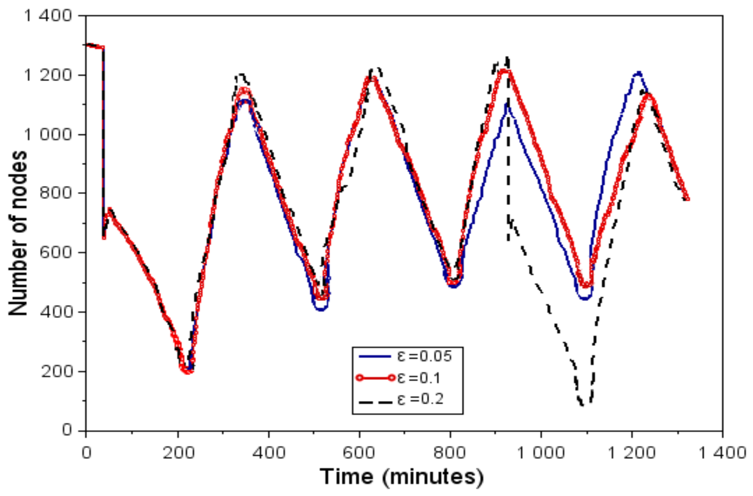


FIGURE 5.14 – Effect of the blocking threshold on the number of nodes.

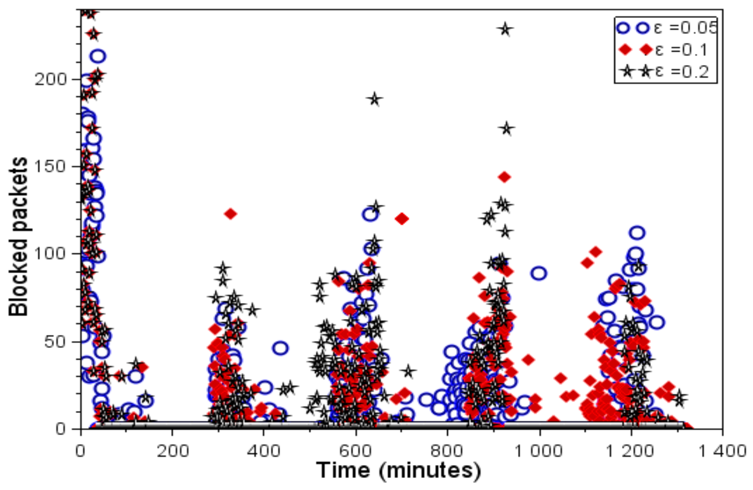


FIGURE 5.15 – Effect of the blocking threshold on the number of blocked packets.

critical situation was persisted and exceeded the time thresholds, each ultra-peer has applied the split process.

5.5.2 Complexity Analysis

In this section, we will study the complexity of the proposed algorithm regarding the communication cost and the generated messages. First, we will re-position again our proposed solution within the ultra-peers maintaining's cycle. Each ultra-peer has a maintaining cycle consists of several processes, to keep it update and to insure all its functionalities. Among others, the election process which consists of checking periodically if the current ultra-peer is always the node with the highest reputation and capacities, and to select a new ultra-peer and its recovery-peer. Our proposed algorithm, especially in its part of the split process, will be a complementary process and will not in any sort, replace the election process. At the same time, the election process can not replace the split algorithm,

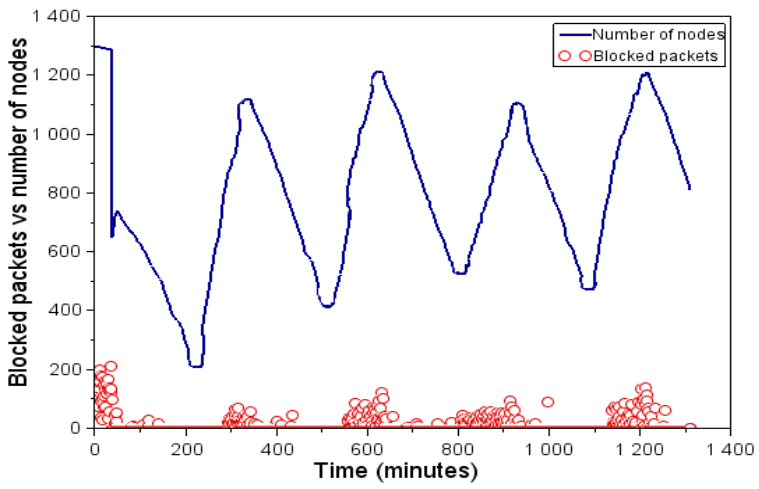


FIGURE 5.16 – Relation between number of nodes and blocked packets for $\epsilon = 0.05$.

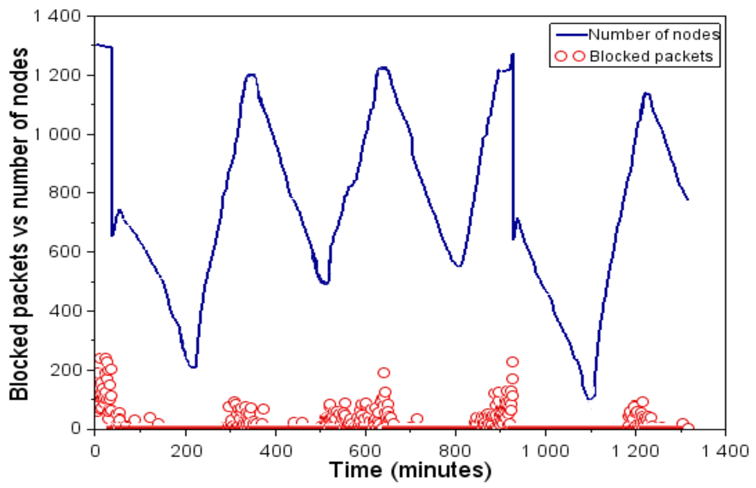


FIGURE 5.17 – Relation between number of nodes and blocked packets for $\epsilon = 0.2$.

because even if it can select a new powerful ultra-peer when the actual one suffers from a serious bottleneck, the limitations are: the possibility of not finding a stronger node to succeed the actual ultra-peer, which will be able to handle the bottleneck situations. In addition, the election process could take a considerably high time before that all peers agree on the new ultra-peer. Nevertheless, we will examine the complexity of both, the election and the split algorithms.

5.5.2.1 Election complexity

The process of election is divided into two phases, the first phase is to elect the best ultra-peer, and the second one is to inform the interconnected ultra-peers about the newly elected ultra-peer. If we assume that the second phase is not a mandatory, and there will be no conflicts and all the cluster will agree on the elected ultra-peer form the first time, so, under an extreme environment where all ultra-peers were interconnected, like in Gnutella, in an overlay of N ultra-peers, and for a cluster of size m, the complexity of the election of the ultra-peer is $O(m^2)$.

5.5.2.2 Split complexity

The split process consists of creating a new ultra-peer cluster from an existing one. For the first time, the recovery-peer will be the first ultra-peer of the new cluster, after that, the cluster starts its life cycle and can design a new ultra-peer using the election process at any time. The new cluster will have the same Checklist as its old one. The complexity of split process consists of the complexity of creating a new cluster and the complexity of updating the Checklist. To create a new cluster, p nodes, where $p < m$, have to leave the old cluster towards the new one, which has a complexity of $O(p)$ to leave the old cluster and another $O(p)$ to join the new one. Hence, the complexity of creating a new cluster is $O(2p)$.

The complexity of updating the Checklist, in the worst case is $O(N)$ in unstructured interconnection where all the ultra-peers are interconnected, or it will be $O(log(N))$ in a structured one like in Chord or Tapestry. Consequently, the overall split complexity is $O(2p + log(N))$.

5.5.2.3 Discussions

— Case 1: We have conducted five days of simulations, during which, the ultra-peer cluster encountered several critical situations, represented by the number of blocked packets, while it has applied a split process only one time, then it has found its steady state. Therefore, for this period, the complexity of our algorithm is $F1 = O(2p + log(N)) + O(T * (p \pm \delta)^2)$, where T is the periodic rate in which the new ultra-peer cluster applies the election process, and δ is the development of number of nodes over time.
Supposing now, that under the same ultra-peer network conditions and arrival/departure rates, at each critical situation the ultra-peer cluster elects a new ultra-peer (if it exists), capable of handling the critical situations. The complexity will be $F2$= $O((C + T') * (m \pm \delta)^2)$, where C is a non-null number of the potential critical situations, T' is the periodic rate of applying the election process.
For both scenarios and under the same conditions, $T' >= T$ and $((m \pm \delta)^2) >= ((p \pm \delta)^2)$, consequently, $F2 >= F1$.

— Case 2:
Now, let's consider a different scenario, in which the complexity of the election process would be equal to $O(m)$ as in a ring algorithm, and that the election process happens only each critical situation. For our proposed approach, before splitting the cluster, the ultra-peer cluster has to select another peer to be the ultra-peer for the new cluster, in this case, the election has a complexity of $O(m)$, and the complexity of the split process will be: $O(2p)$+ $O(m)$+ $Olog(N))$ = $O(2p + m + log(N))$.
Regarding the simulation results, the process with check state has encountered only one critical situation and has applied one split, thus: $F1$=$O(2p + m + log(N))$ And, the process without check state, during

the same period, has exercised several critical situations C, thus: $F2= O((C) * (m \pm \delta))$
and in this scenario, despite our simulation results that show the existence of several critical situations for the process without check state, if we consider that there will be only 3 critical situations, $C = 3$, then $F2 > F1$.

5.6 Summary

The scalability for the ultra-peer paradigm is crucial. It is the ability to add users and components without significantly degrading the performance of systems that are expected to grow and to accommodate a continually changing in their dimensions.

In this chapter, and inspired from ecosystems, we presented a scalability-aware and a self-stabilisation approach for ultra-peers networks to recover their correct state based on workload prediction. The proposed scheme allowed the ultra-peer to scale very well by dynamically control its incoming requests and to anticipate the bottleneck. We have used a neural network and queueing theory to predict the subsequent workload on the ultra-peer to decide whether the next amount of incoming requests will be safety or it will create bottlenecks situations. Then, we have suggested strategies to avoid bottlenecks by first delaying new nodes additions and temporary suspension of the inter-ultra-peers communications, then by the redirection of an amount of the workload to the recovery-peer, and as a last resort, we have proposed a split process which allowed the creation of a new ultra-peer cluster and its integration into the network.

We have evaluated the performance of our system using extensive simulation experiments, under a very heavy synthetic workload, as well as under a heavy workload built from real access traces. The results of which confirmed the effectiveness of the design on scalability, and that the ultra-peer has successfully supervised its state to scale with the growth of the network size.The next chapter will present an economic-based resource allocation.

Chapter 6

A Distributed and Economic Approach for Resource Allocation

6.1 Introduction

This chapter presents the third part of this thesis's contributions, which is a distributed and economic approach for resource allocation in P2P Grids. P2P Grid Computing differs from the other distributed computing like high-performance computing Grids and Clouds because of the non-dedication of its resources, but most importantly because of the nature of its resource contributors which are very often selfish, volatile and resources are connected and disconnected spontaneously, in addition to the impact of the free-riders (the behaviour of its resource users who benefit from the available resources but do not share or contribute by theirs). For these systems, resource allocation is a very complex task because it is hard to guarantee that users jobs can be accomplished, neither the system can not grow to a larger scale. Designing an economics-based approach for resource allocation which goes in the direction of utility computing, gives more incentives to providers to reach jobs assigned to them and to offer more resources, which will enhance the efficiency of the system and ameliorate the QoS, as a consequence users will be more satisfied with the offered services and may be motivated to become providers too by selling their idle resources. Besides, this brings the opportunity to the system to grow and to bring super-computing capacities to the users [Gui09].

6.2 Problematic and motivations

Distributed computing systems including Grids, Clouds and P2P systems have become popular because of their ability to coordinate geographically dispersed computational resources, in favour of users to accomplish their works. Resource allocation is one of the core services in these computational environments, it aims at allocating users' tasks to the available resources based on some predefined rules of selection that ensure both the QoS to the user application according to its requirements and the resources usage policies [Qur+14; Ism07].

P2P Grids and contrasting to other computing environments are characterised by the contributing nature of its resources, by their volatility, and by the selfishness of its users and providers, in addition to the free-riders behaviour of its users. The economical use of resources and system based on market approaches for resources allocation are of primary interest in P2P Grids [Sch+09], it offers an incentive to resource providers to enhance the quality and the quantity of the shared resources and for contributing more to the system, where users have to pay on allocation basis and could motivate them to become providers too. However, several criteria made from the resource allocation a complex process. Often, users and resources providers have different and conflicting goals, constraints and objectives. Users look for resources that respond to their tasks requirements with the lowest possible prices and as fast as possible too; where resources providers would achieve a maximum benefit from the allocation process. In addition, resource providers are self-interests and each provider attempts to attract more users even if this will lead to an over-utilisation of its resources and deteriorating of the *QoS*.

On the other hand, as the number of users increases, their demands on resources with different requirements increase too, this is good for the system resource providers, but it makes the resource allocation more challenging because the environment is non-cooperative and users have to compete with other users to acquire a resource. From the resource provider side, when the number of requests increases, the number of possible ways to allocate the resource (with different but limited capacity) to users' tasks increases too, and it is considered as an NP-Complete problem [Kam+12; Bha+08]. Designing a market-based resource allocation that considers the above-mentioned challenges is fundamental to distributed computing systems like P2P Grids.

Bargaining [Gar+11] is a model for economy-based Grid resource allocation that aims at employing negotiations between users and providers through their brokers or agents to make a decision about the price of the required resource and who deserves it. In the negotiation process, participants exchange deals and proposals until an agreement is reached [Sie+97] or the negotiation deadline is attained. When a proposal is received, it is either accepted or rejected, if rejected, a counter-proposal is issued, making some concessions since the last proposal. Bargaining in a competitive environment like P2P Grid takes place between self-interested parties where their primary interest is to maximise their own utility, normally at the cost of the opponent's utility, but without deteriorating the benefit of any part. Through bargaining, resource providers are given the opportunity to maximise their return-on-investment and consumers to minimise the price they pay for utilising Grid resources [Gar+11; Sim+06; Buy+01b].

In this thesis, we mainly consider a multilateral bargaining strategy for optimising the users and providers utilities. A multilateral negotiation is where different resources' providers throughout their brokers negotiate with multiple trading partners(users). We also consider a non-cooperative environment composed of users and providers, each user has a set of independent tasks to be allocated to different resources at a specified time, where the resource provider disposes a set of computational resources with limited capacity that can be allocated to several users at the same time. A provider receives allocation proposals from several users at the same time and can only satisfy some of them, so instead of bargaining for each deal apart, because it may be a time-consuming and individual decision may not lead to the optimal utility, the resource provider first tries to optimise its utility to find the optimal set of deals, then it follows its negotiation strategy to decide if it accepts the actual deals or to make a counter-proposal and go to another round of negotiations.

Therefore, based on the same system overlay architecture seen in Chapter 4, we propose a distributed market-based resource allocation approach which tries to respond to previous problems. The system is composed of a set of interconnected federations, each federation is managed by a central manager to perform the allocation process. We describe the federation components (users and providers) by utility functions, and assign each component to a broker. The users' utilities to represent their resource demand and preferences, specifically on budget and time, and the resource providers' utilities to express their expected benefits from offering resources (within the limit of their resources capacities). Consequently, this market-based resource allocation would help to build a large-scale computational P2P Grids, by encouraging more resource providers and yielding a fair basis for access to resources, distributes decision-making process, and aids to minimise free-riders impacts; in addition, it enables both consumers and producers to maximise their utility [Buy+02].

The contribution of this study is composed of the following points:

— Modelling the resource allocation problem by utility functions;

— Designing a negotiation model for multilateral bargaining, where brokers are negotiating with multiple trading partners. Multilateral negotiation is more realistic in resource allocation process of computational P2P Grids where there are more than one provider that sell a special type of resource [An+16; Ada+14];

— In literature, two major strategies are proposed for multilateral bargaining: the broker proposes the same price for all trading partners at each negotiation round (makes the same concession for all) [An+16], or proposes different price for each one of its trading partner [Ada+14; Ada+13b]. The former strategy is easy to realise, the broker makes the same amount of concession for all the trading partners regardless their proposals, and the decision-making is usually left until that the deadline of the broker is closed. The latter strategy, in our regard, is more realistic since the broker treats each proposal apart after that it makes concessions, the decision process is more complicated and has to take into considerations other factors like the broker beliefs on the future of the market and its dynamicity. In our work, the broker will first optimise its utility function to select best candidates, then its makes concessions according to the best proposals and market situation. In other words, the bargaining strategy is conducted by best offers and demand and supply;

— We consider the dynamic of the market to model the concession process and the process of decision-making. At each round of the bargaining, the broker has to make a decision on whether to accept an offer or to reject it and go to the next round, but it is faced to uncertainties; it is unsure about the exact benefit it will derive from a particular strategy [Par+08]. In addition, it is unsure about the number of its trading partners in the next time periods that may have an impact on the intensity of demand, thence on prices. We use a stochastic model based on Markov chain to model this uncertainty and to construct a decision-making process for each type of brokers.

It is worth to mention that this thesis does not focus on scheduling algorithms and will not address the payment and currency issues.

6.3 Distributed P2P Grid Utility Maximisation

In this work, we propose a market-based resource allocation in distributed computing environment. We use utility functions to express the federation's components objectives, user's quality of service requirements and resource provider's benefit function.

The system has two objectives, each one is represented by its own utility:

— Maximise the utility of users: the allocation of user's job is based on the quality of service QoS specified by the user and its preferences, without loss of generality, we take into consideration two parameters, budget and deadline. Each user looks for the cheapest resource that can execute its job with respect to the deadline;

— Maximise the utility of resources providers': each resource has to maximise its profit from the allocation process.

6.3.1 The System Components

A resource allocation mechanism allows the mapping of different users' jobs to available and suitable resources obtained by the resource discovery process, which presents the set of potentially suitable resources in response to the users' requests requirements, then the allocation process under the status of the available resources and their allowed constraints and policies allocates the users' jobs to the resources.

We suppose that the system is composed of multiple users and multiple resource providers. Each user has a set of independent tasks with different sizes and resources requirements, these tasks can be run on resources belonging to different providers; and each resource gets a set of offers from several potential users, then it tries to select a set of tasks that maximise its profit through rounds of optimisations and negotiations with users.

The proposed model for resource allocation is described in Figure 6.1, it is composed of:

— Resource providers: components which contribute their resources and charge a user for requested services. They publish their resources into the resources pool of the resource manager. We can refer to it by seller too.

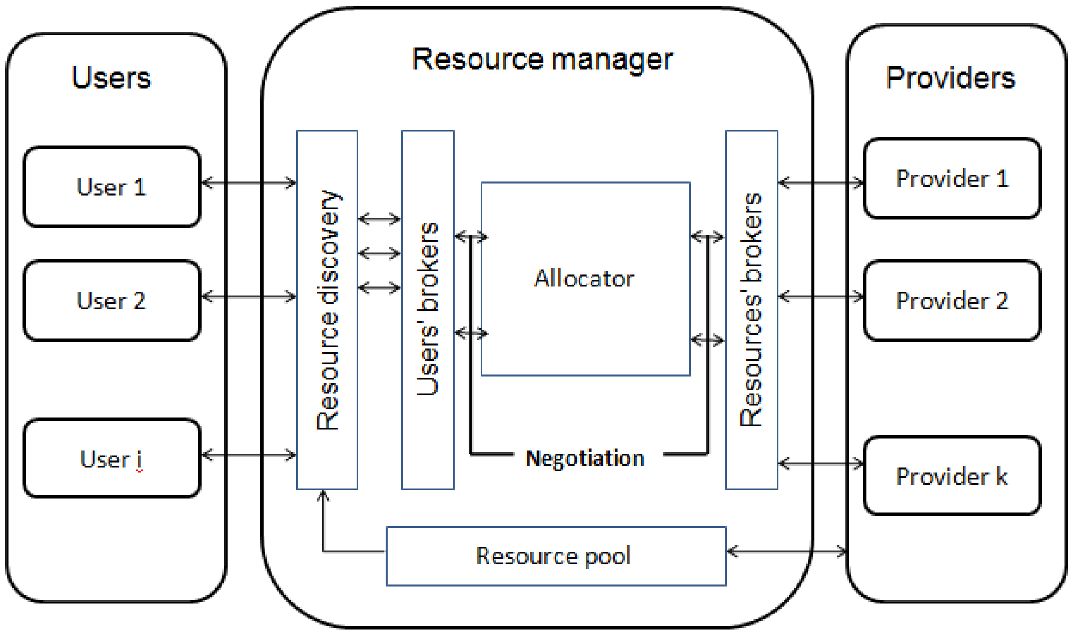


Figure 6.1 – The resource allocation scheme.

— Users: components which have a set of independents tasks, send requests to the resource manager to search and use the system resources, get offers from resource providers, and pay for the used resources. We refer to it also by consumer, client or buyer.

— Resource manager: this component has several tasks, it has a pool in which resource providers can publish their resources, and a resource discovery process which searches on the behalf of users for adequate resources that match their requests requirements among the available resources, and informs users about the appropriate resources. It assigns a broker to each user decides to start the allocation process, the user broker will negotiate with the broker of the available resource's provider according to the required specifications of the corresponding user.

— Allocator: usually the negotiation between users and providers does not need any supervision from a central component, and our proposed solution for resource allocation is distributed. However, in this environment characterised by the volatility and the free-riders, it's mandatory to assign the process of allocation to a more trusted node, thus the allocator will not intervene in the bargaining process but once there is an agreement between user and provider, it is the allocator responsibility to assure the transfer of task from user to provider and the results from the provider to user; it is also responsible for the transaction between them. If the payment is based on a real money, the user pays the required amount to the allocator with its submitted task, and once the allocator ensures a good reception of results by the user, it transfers the money to the provider. If the payment is based on other aspects like reputation score gain or virtual currency, the allocator assures at the end of the process to update the account of both user and provider.

6.3.2 The Basic Model

Let consider a system with a set of $k = 1..NR$ resources' providers and $i = 1..NU$ users. let C_k be the finite capacity of resource of provider k, and each resource has a unit price Pr_k, $C_k{=}V_k * NB_k$ where NB_k is the number of processing element in resource k and V_k the processing power of resource k, measured in terms of $MIPS$ (millions of instructions per second).

Each user has a job composed of a set of independent tasks, $S_i{=}[S_{i1},...,S_{iNJ}]$ is their lengths measured in MI (millions of instructions, $j = 1..NJ$ is the number of tasks submitted by user i.

x_{jk} is a non-negative quantity that a user i need of resource k to execute its task j, it represents the amount of resource k allocated to a task j.

The user utilities are to maximise its benefit from using a resource with respect to its budget and within a specified deadline (generally a starting time). $B_i{=}[b_1,...,b_{NJ}]$ is the budget specified by user i for all its tasks, and $D_i{=}[d_1,...,d_{NJ}]$ is the deadline specified by user i for all its tasks.

According to [Zot+99], users submit their jobs to the system, and they give only a rough estimation of the job running time that is a maximum job running time. In general, this estimated time is overestimated and very imprecise. Thus, in this work, we will not take the time aspect into consideration, and we will optimise the users' utilities regarding their budgets only.

b_{jk} is the amount to pay by user i for its task j to a resource k. $b_{jk}{=}x_{jk}pr_k$. hence, the cost for all user's i tasks on a resource k equals to:

$$\sum_{j=1}^{NJ} x_{jk}pr_k$$

The user's set of tasks can be executed on different resources, and his objective is to find the set of resources that maximise its benefit from paying an amount b_{jk} for using the selected resources k regardless of others. Thus, the benefit will be:

$$\sum_{j=1}^{NJ}\sum_{k=1}^{NR} b_{jk} * a_{jk}$$

a_{jk} is a matrix of affectations, $a_{jk} = 1$ if a task is affected to the resource, $a_{jk} = 0$ otherwise.

The user wants to complete its job with respect to its budget B_i,he wants to affect tasks to the best set of resources. The utility function of a user for all its tasks can be expressed as following:

$$U_{user} = \sum_{j=1}^{NJ}\sum_{k=1}^{NR} b_{jk} * a_{jk}$$

Then the optimisation problem for user i is:

$$MaxU_{user_i}(b_{jk}) \tag{6.1}$$

Subject to:

$$\mathrm{pr_k} > 0; \forall \mathrm{j}, \sum_{\mathrm{k=1}}^{\mathrm{NR}} \mathrm{b_{jk} a_{jk}} \ \leq \mathrm{B_i}; \forall \mathrm{k}, \sum_{\mathrm{j=1}}^{\mathrm{NJ}} \mathrm{a_{jk}} = 1$$

Suppose that resources providers know the amounts b_{jk} that the user i are willing to pay for its task j, and attempt to maximise their benefit, hence, each resource has a set of potential tasks $j = 1..NPJ$, and it tries to allocate those that maximise its profit.

For each resource from all $k = 1..NR$, the revenue that will be obtained by the resource k from the potential allocation of tasks NPJ is the utility function of a resource k:

$$U_{res} = \sum_{j=1}^{NPJ} b_{jk} x_{jk}$$

Then, the kth resource optimisation problem can be formulated as:

$$MaxU_{res_k}(x_{jk}) \tag{6.2}$$

Subject to:

$$\sum_{\mathrm{j=1}}^{\mathrm{NPJ}} \mathrm{x_{jk}} \ \leq \mathrm{C_k}$$
$$\mathrm{j} \ = 1..\mathrm{NPJ}; \ \mathrm{k} \ = 1..\mathrm{NR}$$

Users try to negotiate with resources that can match their jobs requirements and each resource negotiates with users of all its potential tasks, by exchanging the prices and the payments (proposal and counter-proposal).

6.4 The General Bargaining Mechanism

We consider the problem of allocating networked resources in dynamic and distributed computing environment, such as P2P Grid computing, to propose a distributed negotiation mechanism. Where the number of consumers and their requested resources is dynamic and the number of providers and their offered resources is dynamic too. Resource allocation in these environments, characterised by the selfishness of providers and consumers, is very challenging.

In this section, we will introduce the bargaining model used in this work, although with the algorithms to solve the resource allocation optimisation problem in P2P Grid described in the previous section.

6.4.1 The Bargaining model

This section presents the multilateral bargaining process that we will follow. It first describes the used terms and symbols, then it demonstrates the negotiation protocol, after that, it displays the proposed negotiation strategy model.

6.4.1.1 Terminology

In this part, we provide the terminology used throughout this chapter.

— Trading partner: for a client, trading partners are the set of providers' brokers. From the provider perspective, trading partners are the set of users' brokers;
— Competitor: for a client, trading partners are the other users' brokers. In other words, competitors are two brokers of the same type;
— Deadline: during the bargaining, negotiators make concessions, but they have to make a decision at the end of the time limit that called the deadline.
— Preferable price: for a provider, it indicates the price that he wants to reach, where for user it describes the price that he wishes to spend;
— Reserve Price: for a provider, it represents a threshold under it the resource can not be sold. For the user, it signifies the maximum price that he can pay for a resource above it the resource can not be bought.

6.4.1.2 Negotiation Protocol

Negotiation protocol specifies the mechanism and rules of negotiation. Similar to several works [Ada+14; Ada+13b; Ada+13a; An+10; Sim05], we consider the most common bargaining protocol, the Rubinstein's [Rub85; Rub82] sequential alternating offer protocol.

The Rubinstein alternating offer protocol [Rub85; Rub82] is the most widely used bargaining protocol in strategic bargaining games. Negotiation process of this protocol is as follows: the negotiators can take actions only at certain times in the set $Time = 1, 2, 3, ...t'$. In each period $t' \in Time$, one of the negotiators proposes a deal, and the other negotiator either accepts it or rejects it. If the offer is accepted, then the negotiation ends between this two parts, and the agreement is implemented. If the offer is rejected, then the process passes to period $t'+1$; and the other negotiator proposes another deal, usually known by counter-proposal, in its turn, the former negotiator may accept or reject. The negotiation process will go on in this way. The alternating-offers protocol captures the most important features of bargaining: bargaining consists of a sequence of offers and decisions to accept or reject these offers.

The original alternating-offers protocol is designed for the simple discrete time bilateral single-issue negotiation and the allowed actions include (offer and accept) [An+16; An11]. In the purpose of this study, the alternating-offers protocol will be extended to allow the manipulation of more complex negotiation situations like the multilateral negotiation.

In addition to the following assumptions which are given to specify the bargaining rules:

Symbol	**Definition**
UB_i	The broker of user i
PB_k	The broker of provider k
$SNeg_{kj}$	Set of negotiators for the task number j
$MNeg_{kj}$	The mean number of negotiators for the task number j (updated at each negotiation round)
STi	Set of tasks of user i
RP_j	Reserve price for task j
PP_j	Preferable price for task j
RP_k	Reserve price of provider k
PP_k	Preferable price of provider k
$BPPROP_k$	The provider k who makes the best proposal
$BUPROP_{ij}$	The users best bids set
$ENeg$	Expected number of trading partners for the next round
EX_{jk}	Expected demand for the next round
CP_k	Counter proposal for provider k
CP_{ij}	Counter proposal for user i for its task j
Acc_Bid_{kj}	Set of accepted bids from trading partners for task j
Acc_Prop_k	The accepted proposal from provider k
Acc_Prop_{ij}	The accepted proposal for task j
$SNPJ$	The provider set of potential users tasks
τ	The current trading time
α	The The negotiation deadline
β	Time preference

Table 6.1 – List of key notations.

— We consider one issue negotiation attribute; price of a good, and a multilateral bargaining without outside knowledge where brokers have incomplete information about each other like deadlines and reserve prices, and uncertain about the market dynamic like number of competitors, or the changing rate of their trading partners;
— The P2P Grid resource negotiation progresses in a series of rounds;
— Time is discrete and is indexed by $0, 1, 2, \ldots$ where ”0” designates the first round [Ada+14; Ada+13b; Sim+06];
— Negotiators have information only about the index of the time period;
— A negotiator proposes its most preferred deal initially [Sim+06].
— Coalition is not allowed [Ada+14; Ada+13a; Sim+06], the negotiators do not cooperate and make decisions independently;
— There is a finite number of negotiators.

6.4.1.3 Time Function

In Bargaining, the passage of time has an important impact on the decision of the bargainers and the amount of the concession that they will make [Rub85]. The deadline may put negotiators under pressure [Sim+06; Ken94; Rub85]. Sim in [Sim+06; Sim05] take into consideration the mentioned concept by introducing time discount factor in their proposed concession making strategies.

The effect of time discount factor in negotiator's bargaining power has been modelled via time-dependent function by [Sim05]. The time-dependent function is used to decide the amount of concession in the price of the resource.

The time-dependent function is $T(\tau, \alpha, \beta)$:

$$T(\tau, \alpha, \beta) = 1 - (\frac{\tau}{\alpha})^{\beta} \tag{6.3}$$

Where: ($\tau > 0$ and $\beta \geq 0$)

The time preference β reflects the negotiator preference about its eagerness for finishing the negotiation earlier [Far+98]. It means that the negotiator with different time preferences may adopt different concession rates with respect to time [Sim05]. For example, the negotiator may prefer to concede less rapidly in the early rounds of negotiation and more rapidly as its deadline approaches.

Although there are infinitely many strategies with respect to the remaining trading time, they can be classified into three major classes of concession making strategies as follows [Sim05]:

— Conservative: $1 < \beta < \infty$, an aggressive strategy, the negotiator makes smaller concession in early rounds and larger concession in later rounds;
— Linear: $\beta = 1$, a neutral strategy, the negotiator makes a constant rate of concession;
— Conciliatory: $0 < \beta < 1$, a defensive strategy, the negotiator makes larger concession in the early trading rounds and smaller concessions in the later rounds.

6.4.2 Bargaining Strategy

In this section, we will describe the bargaining strategy between a provider and a user, more precisely between their brokers PB and UB respectively.

The bargaining strategy for users and providers is affected by many factors; deadline, budget and reserve prices, demand and supply, competitions, in addition to uncertainties surrounding most of these factors. Developing a framework that covers all these factors is hyper-complicated. In this proposition, we try to connect those interdependent factors using a mixture between heuristics and existing algorithms to approximate brokers' decision-making.

In summary, Each provider wants to sell resources of limited capacity to one or several clients for a price. Each client has a set of independent tasks and wants to buy resources (from one or different providers) to accomplish his tasks.

All the brokers involved in the same bargaining enter the market at time 0, but new brokers can join the process later, before the deadline of the current bargaining.

Whereas consumers select resource providers that offer the lowest service costs and also meet their budget requirements, resource providers offer services to the resource consumer with the highest bid as long as the consumer's objectives can be met.

User and provider do not follow the same strategy, the user strategy for negotiation focuses on maximising its utility function constrained by its budget

6.1, thence the user looks for resources with the lowest possible price. Whereas the provider utility is to maximise its utility 6.2 constrained with the capacity of its resource. The provider looks for tasks with the highest possible price without violating the capacity constraint. Which means if there is a resource with a capacity $= cap$ and two tasks, $ts1(size1, price1)$ and $ts2(size2, price2)$ where $price1 < price2$ and $size1 < cap < size2$, the provider then chooses to run $ts1$.

More details on the bargaining strategy for both users and providers are provided in the following sections.

6.4.2.1 The User's Bargaining Strategy

The bargaining strategy for the user is summarised in the following algorithms 5, 6, 7 and 8.

By considering both, the remaining time, the best resources' providers proposal and market supply, the user's offering price for its trading partners is calculated in the following way:

$$\mathrm{CP_k} = \mathrm{Min} \begin{cases} RP_j - T * \frac{SNeg_{kj}}{MNeg_{kj}} * (BPPROP_k - PP_j) \\ BPPROP_k \end{cases} \tag{6.4}$$

The $\frac{SNeg_{kj}}{MNeg_{kj}}$ represents the rate of the supply. So that the user's concession will be calculated according to the demand-and-supply too, if the supply increases then the resources prices' decrease.

The UB strategy is to try to make agreements which can satisfy its resource requirements and optimise its utility eq: 6.1. It initialises the bargaining by selecting the set of its trading partners to whom it presents its first offer, usually its PP_j. At each negotiation round, UB either receives counter proposals or acceptation for its offers. For both cases, it has to evaluate the current situation according to its state following the algorithm 6. So that, if it is not its last round of negotiation, the UB optimises again its utility function to select the best bid, then it evaluates the received counter proposals using algorithm 7. Otherwise, if it is the last round of negotiation the UB must take a decision or it will fail to find an agreement, for this reason, it will accept the best-offered bid as long as it is lower than its reserve price.

Algorithm 5 User's Bargaining_Initialisation: $UB\_Initialisation()$

```
Require: initialisation of: Budget, deadline $\alpha$, $\beta$, $STi$;
Require: $\tau = 0$, $t$
1: for each task $\in STi$ do
2:     Set $RP_j$ and $PP_j$;
3:     Select providers and update $SNeg_{kj}$;
4:     Generate proposals for all selected providers.
5: end for
```

Since the elapsed time is not higher than a specified time threshold (which is set to divide the UB bargaining time into relaxed time and deliberated time),

Algorithm 6 User Bargaining_Strategy: $UB\_Strategy()$

```
1: Initialise_Bargaining $UB\_Initialisation()$;
2: while $(\tau < \alpha - 1)$ and $(STi \neq \phi)$ do
3:    for all received bids do
4:       Optimise the user utility using Eq:6.1 and select best bids $BPPROP_k$ for each task;
5:       $Evaluate\_Proposal()$;
6:    end for
7: end while
8: if $(\tau = \alpha - 1)$ and $(STi \neq \phi)$ then
9:    /* the last round of bargaining*/
10:   for all tasks $\in STi$ do
11:      Optimise the user utility and select best proposals $BPPROP_k$ which should be $< RP_j$;
12:      $Accept\_Proposal(BPPROP_k)$.
13:   end for
14: end if
```

the UB can reject offers and makes a new counter proposal using eq: 6.4. But if it arrives at the deliberated time and that it is not the last round of negotiations, the UB estimates the number of its trading partners for the next round of negotiation which corresponds to the future supply using the Markov chain proposed model. Accordingly, it decides whether to accept the offer or to generate a counter proposal.

Algorithm 7 User Proposal_Evaluation: $Evaluate\_Proposal()$

```
1: if $\tau \leq t$ then
2:    for each task $\in STi$ do
3:       Generate a Counter_Proposal according to $BPPROP_k$ and Eq: 6.4 and communicate all the concerning trading partners;
4:    end for
5: else {*it is not the last round but $\tau > t$ *}
6:    for all $BPPROP_k$ do
7:       { analyse only offers from the best proposals }
8:       Estimate the number of trading partners for each corresponding task for the next round: $ENeg$
9:       if $ENeg > 1$ then
10:         Generate the Counter_Proposal: $CP_k$
11:         if $CP_k = BPPROP_k$ then
12:            Accept_Proposal($BPPROP_k$)
13:         else
14:            Make a Counter_Proposal($CP_k$) for the provider $k$
15:         end if
16:      end if
17:   end for
18: end if
```

If the UB gets acceptations on its proposals, it refers to the algorithm 8 to evaluate the accepted offers. If the number of the received accepted proposals equals to the number of its trading partners, then the UB must choose one provider and validate the agreement with it (algorithm 11) because once it refuses to validate an accepted offer, the negotiation between the two trading partners will be cancelled (algorithm 10). Else, if the passed time is lower than the time threshold, UB does not have to validate the acceptation because it makes the same proposal to all of its trading partners, thence there is more time to reach better opportunities. But if the time is passed, the UB have to evaluate the risk of not approving the acceptation.

Algorithm 8 User's Accepted_Proposal: $Acc\_Prop_k()$

```
1: if $SNeg_{kj} = 1$ then
2:     /*there is only one trading partner*/
3:     $Validate\_Proposal()$
4: else
5:     if $Acc\_Bid_{kj} = SNeg_{kj}$ then
6:         /*all trading partners accept the user proposal*/
7:         Select a provider and validate the operation.
8:         Cancel the bargaining with the remaining providers.
9:     else
10:        if $\tau < t$ then
11:            Refuse the proposal $Acc\_Prop_k$ and cancel the bargaining;
12:        else {*evaluate the risk of not validating the current accepted proposal*}
13:            $Risk\_Eval()$
14:        end if
15:    end if
16: end if
```

The risk evaluation (see algorithm 9) is used when the time of negotiation attains the deliberated phase and UB needs to decide if it has to approve an accepted proposal or not. It aims at estimating the future resources' supply using the Markov chain process and generating of the corresponding counter proposal. If the UB expects a reasonable supply and that the generated counter proposal is better than the current proposal, the UB then has to refuse the proposal and cancel the bargaining with the corresponding provider, otherwise, it must validate the proposal.

6.4.2.2 The Provider's Bargaining Strategy

The bargaining strategy for providers is much similar to users' strategy. It is summarised by algorithms 12, 13, 14 and 15. Their key differences are that the provider follows the market demand to update its price. Providers which receive fewer demands may decrease their prices if they want to earn more allocations, consequently, providers who receive lots of demand can increase their prices to make a maximum profit. Each provider through its broker PB receives several proposals from several potential clients, at this moment it has to solve the

Algorithm 9 User's Risk Evaluation: $Risk\_Eval()$

```
1: Estimate the number of trading partner for the next round: $ENeg$.
2: Generate the Counter_Proposal: $CP_k$ according to Eq: 6.4;
3: if $ENeg > 1$ then
4:     if $CP_k >= Acc\_Prop_k$ then
5:         Validate_Proposal()
6:     else
7:         Refuse the proposal $Acc\_Prop_k$ and cancel the bargaining Cancel();
8:     end if
9: else
10:    Validate_Proposal()
11: end if
```

Algorithm 10 Cancel the Bargaining: $Cancel()$

```
1: Remove the corresponding provider from $SNeg_{kj}$
2: or remove the corresponding user's task from the potential client set $SNPJ$;
3: Cancel the bargaining with the corresponding trading partner.
```

Algorithm 11 Proposal Validation: $Validate\_Proposal()$

```
1: Update the user budget and remove the task from $STi$
2: or update the provider resource capacity $C_k$;
3: Validate the acceptation of the proposal with the corresponding trading
   partner.
```

combinatorial allocation problem which is NP-complete (a knapsack problem), using the dynamic programming.(Chapter 2).

In the same way as for users, the provider update price formula used to generate proposals and counter proposals considers different parameters, the remaining time, the best users' proposal and market demand. The provider's offering price for its trading partners is calculated in the following way:

$$\mathrm{CP_k} = \mathrm{Max}\begin{cases} RP_k + T * DS_{jk} * (PP_k - BUPROP_{ij}) \\ BUPROP_{ij} \end{cases} \tag{6.5}$$

$$\mathrm{DS_{jk}} = \begin{cases} \frac{|X_{jk} - C_k|}{C_k} & if |X_{jk} - C_k| \neq 0 \\ 1 & otherwise \end{cases}$$

The DS_{jk} represents the rate of the demand. So that the provider's concession will be calculated according to the demand-and-supply; if the demand increases then prices of resources increase too.

Algorithm 12 Provider's Bargaining_Strategy: $PB\_Strategy()$

```
Require: initialisation of: available resource capacity $C_k$, deadline $\alpha$, $\beta$ , $t$ , $RP_k$
    and $PP_k$;
Require: $\tau = 0$.
 1: while $(\tau < \alpha - 1)$ and $( C_k \neq 0)$ do
 2:   for all received bids do
 3:     if $\sum(X_{jk}) > C_k$ then
 4:       Optimise the provider utility using Eq:6.2 and return best bids
          $BUPROP_{ij}$;
 5:     end if
 6:     $Evaluate\_Proposal()$;
 7:   end for
 8: end while
 9: if $(\tau = \alpha - 1)$ and $( C_k \neq 0)$ then
10:   /* the last round of bargaining*/
11:   for all received bids do
12:     if $\sum(X_{jk}) > C_k$ then
13:       Optimise the provider utility and return best proposals $BUPROP_{ij}$
          which should be $< RP_k$;
14:     end if
15:     $Accept\_Proposal(BUPROP_{ij})$.
16:   end for
17: end if
```

Algorithm 13 Provider's Proposal_Evaluation: $Evaluate\_Proposal()$

```
 1: if $\tau \leq t$ then
 2:   for each task $\in SNPJ$ do
 3:     Generate a Counter_Proposal according to $BUPROP_{ij}$ and Eq: 6.5 and
        communicate new price to all the concerning trading partners;
 4:   end for
 5: else {it is not the last round but $\tau > t$ }
 6:   for all $bestoffers$ do
 7:     /* analyse only selected offers returned by the optimisation of Eq:6.2*/
 8:     Estimate the number of trading partners and the quantity of their re-
        quested resources for the next round: $EX_{jk}$.
 9:     if $(EX_{jk}) > C_k$ then
10:       Generate the Counter_Proposal: $CP_{ij}$
11:       if $CP_{ij} <= BUPROP_{ij}$ then
12:         Accept_Proposal($BUPROP_{ij}$)
13:       else
14:         Make a Counter_Proposal($CP_{ij}$);
15:       end if
16:     end if
17:   end for
18: end if
```

Algorithm 14 Provider's Accepted_Proposal: $Acc\_Prop_{ij}()$

```
1: if $\sum(X_{jk}) < C_k$ then
2:     /* low demand rate */
3:     $Validate\_Proposal(Acc\_Prop_{ij})$
4: else
5:     if $\tau < t$ then
6:         Refuse the proposal $Acc\_Prop_{ij}$ and cancel the bargaining for this task $j$;
7:     else {*evaluate the risk of not validating the current accepted proposal*}
8:         $Risk\_Eval()$
9:     end if
10: end if
```

Algorithm 15 Provider's Risk Evaluation: $Risk\_Eval()$

```
1: Estimate the number of trading partners and the quantity of their requested resources for the next round: $EX_{jk}$.
2: Generate the Counter_Proposal: $CP_{ij}$ according to Eq: 6.5;
3: if $(EX_{jk}) > C_k$ then
4:     if $CP_{ij} = Acc\_Prop_{ij}$ then
5:         Validate_Proposal()
6:     else
7:         Refuse the proposal $Acc\_Prop_{ij}$ and cancel the bargaining Cancel() for this task;
8:     end if
9: else
10:     Validate_Proposal()
11: end if
```

6.5 The Markov Chain Model

For the prediction of the expected number of trading partners and the expected demand of resources in next negotiation round, we inspire several aspects from the model proposed by [An+08]. The original model *MCDM* (Markov Chain based Decision Making) is proposed for decision-making of negotiation agents that they can use it to determine when to complete negotiation, it allows the computing of the expected utility if they decide to complete negotiation at the next round, then they compare the expected utility with the actual one, upon which they make decisions. The MCDM strategy is based on heuristics. It models the dynamics of negotiation stochastically using a Markov Chain *MC*, assuming that the negotiation behaves as a random process. The MC model captures the variables that influence the agent's utility values and the uncertainties associated with them. The MC model takes those variables into account in the MC states and the transition probabilities. The MCDM is designed from the perspectives of agents of type 'buyer' only and does not take market competition into account.

The model used in this work is quite simple and correspond to any ordinary Markov chain, but it captures other features not considered in MCDM, first, each broker type has its special MC. The UB used the MC to estimate the number of its trading partners in the next round, where the PB used the MC to measure the expected demand on resources in the next round. Second, because we assume no knowledge about the trading partners and competitors, we do not model the deadlines and reserve prices, but we focus on the market dynamic and we model our bargaining model based on the demand and supply. From the user perspective, the high number of trading partners means that the supply is high, and from the provider perspective the high quantity of requested resources (represented by the size of tasks) means that the demand is high.

So that, we model both types of MCs to estimate the number of trading partners in the future negotiation. For this end, the MC needs to know the information about the rate of which new partners enter to the negotiation and existing partners quit the negotiation, we assume that this information follows some probability distribution (Poisson for example).

For example, if we consider that the UB knows its actual status of negotiation at instant t' and uses the MC to expect its state at $t'+1$. Therefore, from its initial state, at each negotiation round, it can determine the set of its possible states.

We will illustrate the proposed MC for UB through an example:

Let the initial state be the state while building the MC for getting the expected number of negotiators is the set of providers the UB negotiates at round t'.

At round $t'+1$, there will be two actions that can occur, action1 is that new providers enter the negotiation and action2 existing providers leave the negotiation.

If we make the same assumption like in the MCDM model, assuming that at most one trading partner leaves negotiation and at most, one new trading partner enters negotiation at each negotiating round.

Since the UB wants to estimate its $ENeg$, following the two possible actions the $ENeg$ can take one of the three possible values:

— State1: if only one new trading partner enters the negotiation which means that the action1 occurs: $ENeg = SNeg_{jk} + 1$;
— State2: if only one trading partner leaves, which means the action2 occurs: $ENeg = SNeg_{jk} - 1$;
— State3: if a new negotiator arrives and an existing negotiator quits, this means that both actions occur at the same time: $ENeg = SNeg_{jk} + 1 - 1$. Or if no new partner arrives and no existing one leaves which means no action occurs: $ENeg = SNeg_{jk}$. For this two cases the state of the negotiation does not change, therefore we model these two cases by the same state.

So, from the actual state, the negotiation can go to any of the following three possible states. These states are described in table 6.2. In our model, we do not pay attention to the identity of the leaving partner, because the proposed strategy makes the same concession to all the trading partners, what is really matter is the number of trading partners.

	State1	**State2**	**State3**
State1	P_{11}	P_{12}	P_{13}
State2	P_{21}	P_{22}	P_{23}
State3	P_{31}	P_{32}	P_{33}

TABLE 6.2 – State transition for users

The MC of PB is modelled in the same way as the $UB's$ MC, it tries to expect the number of negotiators at the next negotiation round. We associate at each resulting state a probability distribution function (for example uniform) to give a value to the task size that will be brought or left with the corresponding negotiator, we refer to it by expected task size EJS. Example, if the expected state reveals that a new negotiator will enter to the negotiation, then this negotiator will bring a task with a specified size given by a probability distribution function.

The motivation behind the estimation of $ENeg$ rather than EX_{jk} is for simplicity. Example: if we expect the arrival of 10 new customers to the negotiation (knowing that this fact is a state in the transition matrix), then the expected number is 10. Where if we use the tasks sizes, it will be very hard to estimate the size of the 10 tasks, because one task may have several possibilities for its size, even if we limit the task sizes. Another possibility is to model the PB MC states by the quality of the expected demand, for example by low demand and high demand comparing to the provider resource capacity C_k. However, the proposed concession strategy requires an explicit amount of the demand (the task size) and not its quality.

The PB wants to estimate its EX_{jk} through the estimation of $ENeg$, it follows the two possible actions, the EX_{jk} can take one of the four possible values:

— State1: if only one new trading partner enter the negotiation which means that the action1 occurs: $EX_{jk} = X_{jk} + EJS$;
— State2: if only one trading partner leave which means the action2 occurs: $EX_{jk} = X_{jk} - EJS$;
— State3: if a new negotiator arrives and an existing negotiator quits, this means that both actions occurs at the same time:$EX_{jk} = X_{jk} + EJS1 - EJS2$ such that $EJS1 \neq EJS2$
— State4: if no new partner arrives and no existing partner leaves which means no action occurs: $EX_{jk} = X_{jk}$

So, from the actual state, the negotiation can go to any of the following four possible states. These states are described in table 6.3.

	State1	**State2**	**State3**	**State4**
State1	P_{11}	P_{12}	P_{13}	P_{14}
State2	P_{21}	P_{22}	P_{23}	P_{24}
State3	P_{31}	P_{32}	P_{33}	P_{34}
State4	P_{41}	P_{42}	P_{43}	P_{44}

TABLE 6.3 – State transition for providers

6.6 Early Experiments and Preliminary Results

Although we have been working towards the evaluation of the performance of the proposed approach, it is still work-in-progress. Hence, in this section, we present our first experiments and evaluations that we undertook using the same simulator presented in the previous chapters.

6.6.1 Simulation Parameters Settings

This simulation of the multilateral negotiation is carried out according to the following input parameters:

We remind that resources are shared among tasks. We consider only one type of resource, the processor that is characterised by its capacity expressed in millions of instructions per second $MIPS$. The resource cost can be defined as the processing cost per MIPS. The capacities of the resources are chosen uniformly in the interval $[50, 500]$, the reserve price of the resource is set from 10 to 20 (monetary unit) and the preferable price is set from 50 to 100.

We assume that each user can submit tasks to a set of providers (not all of them). Jobs (users) arrive at each site according to a homogeneous Poisson process, and the number of tasks per job varies from 1 to 5. To take into account the wide dispersion in tasks sizes, the sizes of the task is taken randomly from the uniform distribution in the interval $[10, 100]$. The budget of the user depends on its tasks sizes, it is equal to the $(job\_size * reserve\_price)$, its reserve price is set from 50 to 100, and the preferable price from 5 to 20.

For this part of simulation, we consider only a linear time preference for negotiation $\beta = 1$, the deadline α for both user and provider is selected from the interval of $[10, 20]$ ($deadline1$) for the first set of simulations and from $[30, 40]$ for the second one ($deadline2$). We set $t = \alpha/2$.

6.6.2 Simulation Results

Figure 6.2 presents the users' success rate according to the resource supply for two types of deadlines above mentioned. The success rate is the number of the successful tasks that found a consensus per the total number of generated tasks into the negotiation. For each experiment, we try to maintain a regular number of existing providers during the whole process of bargaining. Providers enter and leave the negotiation process at each negotiation round, but their average number is kept almost steady. For both types of deadlines, the success rate increases with the number of resources. At the beginning, there were fewer available resources in the system, and hence, it would be more difficult for users to successfully negotiate for resources and find agreements. It can be seen too, that success rate when negotiators use a more long deadline is higher than those who use a short one, this is because with a long deadline negotiators are under less pressure and with the passage of time new providers can arrive and bring more resources, so they were faced with more opportunities to find agreement.

Figure 6.3 depicts the number of reached agreements by providers according to the demand for two the types of deadlines. As in the previous experiment,

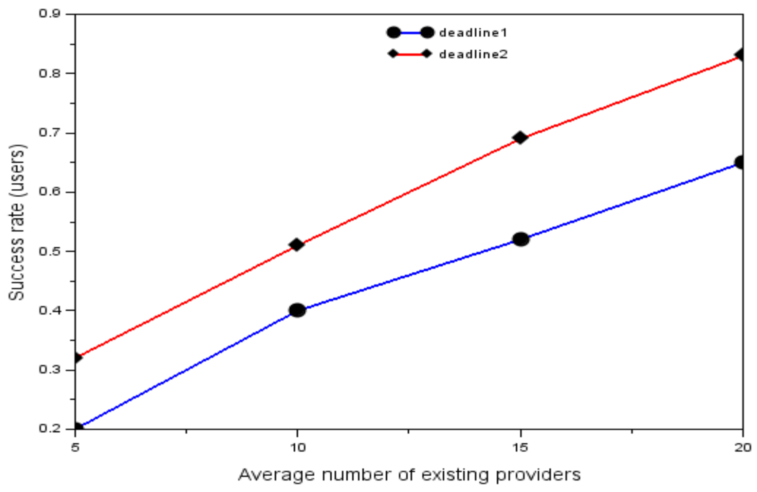


FIGURE 6.2 – Users' success rate and supply dynamic.

in this one, we try also to maintain a regular number of existing users during the whole process of bargaining. Users enter and leave the negotiation process at each negotiation round, but their average number is kept almost steady. We observe that, for example, for the first measurement $(20, \approx (150, 210))$ the average number of users is 20 (remind that a user generates at most 5 tasks), the number of the agreement reached surpasses 100. This is because of: first the market is dynamic and at each round new users enter the bargaining. Second, the few number of users mean a low demand, thence providers have adapted their prices with this situation and made interesting concessions to sell their resources and reach agreements. Although, the number of reached agreement increases with the number of users and it can be seen that it is higher for negotiators that use long deadline.

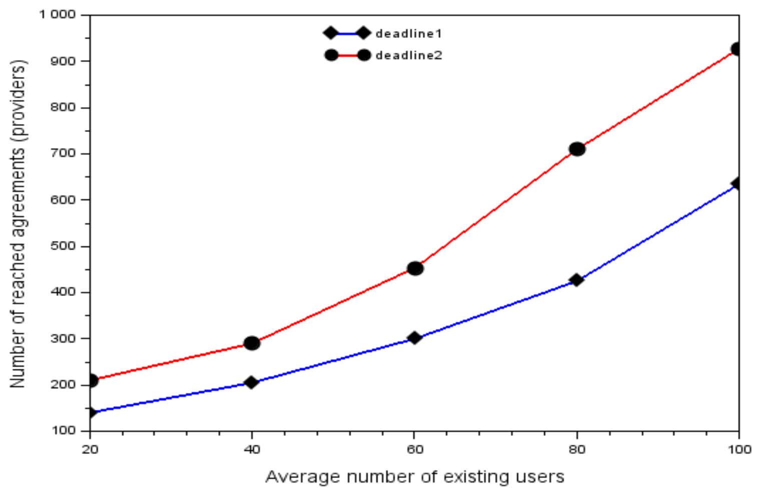


FIGURE 6.3 – Providers' reached agreements and demand dynamic.

6.7 Summary

This chapter presented an economic and distributed resource allocation for P2P Grid Computing. The economic aspect for resource allocation in this environment that is characterised by the volatility and the selfishness of its components is very important because it gives incentive to these components to contribute more resources to the system. We have first modelled the system and its components by utility functions. Then, we have designed the negotiation model for multilateral bargaining, where brokers are bargaining with multiple trading partners through alternative rounds of negotiations. For this end, we have proposed several algorithms to conduct the bargaining strategy. As well, we have proposed a stochastic model based on Markov chain model to capture the dynamic and the uncertainties imposed by the environment, to help the negotiator to take the right decision at each negotiation round.

Yet, some shortcomings remain. In the proposed approach, we did not consider the task execution starting time and its deadline. In addition, we dealt only with a single type of resource. However, two or more different types of resources are simultaneously required in order to execute a task. Therefore, the suggested mechanism can be extended to consider resource combinations and optimisation.

Our preliminary results showed that the proposed multilateral bargaining model has yielded promising results. However, an extensive amount of stochastic simulations that combine the different used parameters and comparisons with existing approaches is needed to evaluate all the features presented by the current work.

Chapter 7

Conclusions, Perspectives and Future Insights

“Anybody who has been seriously engaged in scientific work of any kind realises that over the entrance to the gates of the temple of science are written the words: 'You must have faith.' ”

—— Max Planck

7.1 Conclusions

This thesis dealt with some of the main issues related to resource management in environments that are characterised by communities composed of dynamic and volatile participants, who share their heterogeneous resources and require a mechanism to increase their efficiency and improve the scalability when the community grows. The work described in this thesis made a number of interesting contributions. It aimed to study three main aspects related to the resource management in these environments where the P2P Grids was taken into consideration because they are a good representative of these environments.

The thesis had first taken into account the system architecture and the resource discovery process (RD) in which a combination of the ultra-peer techniques with Semantic Web technologies via a SKOS lightweight ontology to build a three layered architecture for P2P Grid RD were proposed. Nodes were clustered according to their domain of interest to form groups called federations. A process for constructing federations and the layered overlay network was suggested, in addition to a mechanism of query routing to target federations in an efficient and a scalable way.

Second, the thesis proposed a scalability-aware approach for ultra-peers like networks, where each ultra-peer was able to prevent the presence of bottleneck and to regulate and maintain it to stay in a steady state. It used neural networks in conjunction with queueing theory to create a process of prediction and decision to estimate the future workload for each ultra-peer, and to take decisions on whether the next period will become a bottleneck situation or not. It provided solutions in case the ultra-peer has expected to receive an unmanageable workload, which could cause bottlenecks problems. The proposed strategy enabled the ultra-peer to scale with the growth of the network size, in addition, it permitted to add new ultra-peers' clusters to the system easily.

Lastly, the thesis presented a distributed and economic approach for resource allocation. It utilised the multilateral bargaining where several users and providers were allowed to bargain concurrently with each other in a dynamic and complex negotiation environments and with incomplete information about the market. It modelled the system components (user and provider) by utility functions where each one had different objectives and wanted to maximise its profit through a series of negotiations with its trading partners. Different algorithms were proposed to realise the objectives of this proposition. The suggested mechanism of negotiation allowed the system components to achieve their goals.

To validate the effectiveness of the proposed approaches, we created a simulator and implemented the different algorithms. We conducted a series of experiments to study the performances of our system: response time, communication overhead and efficiency of the semantic routing algorithm, scalability and the state control.

Results demonstrated that the use of Semantic Web and the created SKOS ontology permitted to the routing algorithm to efficiently localise the resource.

Experiments through the synthetic and realistic workload showed the ability of our system to scale and to keep its steady state under severe circumstances.

Finally, the distributed resource allocation using bargaining and the optimisation allowed to all parts of the system to maximise their utilities.

7.2 Perspectives

Our first ongoing work is to ameliorate the current implementation of the resource allocation mechanism, to cover all its proposed features, and to evaluate its performances by comparison with other existing works. In addition, in the proposed work, the multilateral bargaining model used for resource allocation considered only one single attribute negotiation which was the price. It will be more interesting to consider other attributes such as time and other QoS.

Also, modelling the brokers (for both users and providers) as agents using the multi-agents system aspects will ameliorate their reaction by allowing the design of their behaviour in a way that matches the user/provider policy and to adapt to new conditions of the market.

Another important challenge is to consider other economic models in order to support other kinds of services and resources like games and soft-wares.

Finally, the reconsideration of the recommended resource allocation to allow it to be adaptable to other distributed environments like Clouds.

7.3 Insight into the future of P2P Grid Computing

In the era of Cloud Computing, what is the future for P2P Grids (or in general P2P Desktop Grid Computing (*P2PDGC*))?

The answer to this question is related to the position that can take the P2PDGCs, between death, survive, reborn or evolution.

P2PDGCs will die if we see them as concurrent to the Cloud Computing, their QoS is much lower than those offered by Clouds. Hence, big companies offering Cloud services will dominate the market.

P2PDGCs will survive if we see them as an alternative solution for the distributed computing and the resource sharing, or that they can cooperate with Cloud Computing:

— Cooperation between these two paradigms can be achieved in a way that Cloud can allocate resources from a P2PDGC when it is needed and vice versa. The collaboration can be achieved too if the two systems work together. If the user needs to run applications in a bag of tasks fashion, it will be oriented to P2PDGC; and if he needs to analyse data or he has a massively parallel application with communication it will be redirected to the Cloud.

— As an alternative solution; from the fact that each one of the systems has its own community of users and can attract special clients. The Cloud is well known to use the pay-per-use, and users have to pay for services. This could be attractive for clients who have money and want to pay for good services. On the other hand, users of P2PDGC can use services for lower prices than Clouds, or can pay through virtual currency like "bitcoin", or can gain reputations and exchange services. With the increasing of number of personal computers and their huge aggregated capacity, users may prefer to pay for services that may be lower or not than those offered by the Cloud, in that they can earn money by becoming providers too and execute applications for other users of the system rather than using Cloud without getting anything in return but the needed service. In accordance with, it is for the benefit of users to use the P2PDGC and that the system remains operational.

P2PDGC could reborn, and appear as a new paradigm. If we observe the history of the information technology, we perceive its evolution from centralised systems through the mainframes (where the centralised computer does everything, it is connected to terminals which do not make any computation). Then, it comes the apparition of personal computers and mini-mainframes (servers) leading to the creation of distributed systems and the client-server version (both parts store data and make computation). After that, the Internet and HTTP raise resulted in the creation of more distributed systems like P2P and Grids. Now, it is the time of Clouds (the data centres and VM do everything from data storage to intensive computation where the users do not make any computation but they pay for services), which are an evolved version of mainframes and the centralised systems. We can imagine that the next step is a new distributed systems composed of several Clouds, owned by several organisations and companies connected through the Internet in a P2P fashion. This could be called as P2P-Clouds.

P2PDGS may evolve, and lead to the emergence of a more powerful system than what it is now. Usually, P2PDGC is suitable for bag of tasks applications that do not need communication between them, due to its nature based on volunteer computers connected through low bandwidth and high latency network. According to Trieflinger [Tri13], the Nielsen's Law [Nie98] that claims that the bandwidth available to end users' grows by 50% per year– slower by 10% than the number of transistors of a processor according to Moore's Law [Moo75]. However, the broad adoption of optical fibre network technology will probably overthrow this relation as stated by the Butter's Law [Teh00]. This latter states that the bandwidth of an optical fibre is doubling every nine months. Thus, bandwidth-related inefficiencies of applications with lower compute intensity are likely to become less dominant. In addition, the wireless networking has reached a high degree of maturity and result to a more available bandwidth. These two facts will offer to the P2PDGC the ability to run new kind of applications that need a large amount of data to be exchanged. So That, if we assume the laws of Moor and Butter, we will see more powerful desktop computers and faster networks, which may allow to the P2PDGC to execute applications that today are executable on supercomputers only.

References

[Ada+13a] S. Adabi, A. Movaghar, A.M. Rahmani, and H. Beigy. "Market-based grid resource allocation using new negotiation model". In: *Journal of Network and Computer Applications* 36 (2013), pp. 543–565.

[Ada+13b] S. Adabi, A. Movaghar, A.M. Rahmani, and H. Beigy. "Negotiation strategies considering market, time and behavior functions for resource allocation in computational grid". In: *The Journal of SuperComputing* 66.3 (Dec. 2013), pp. 1350–1389.

[Ada+14] S. Adabi, A. Movaghar, A.M. Rahmani, H. Beigy, and H.D. Tabrizi. "A new fuzzy negotiation protocol for grid resource allocation". In: *Journal of Network and Computer Applications* 37 (2014), pp. 89–126.

[ALI+12] H. ALI and M. Ahmed. "HPRDG: A scalable framework hypercube-P2P-based for resource discovery in computational Grid". In: *Proceeding of the 22nd Conference on computer theory and applications (ICCTA'2012).* Alexandria, Egypt: IEEE, 2012, pp. 2–8.

[Ali+12] A. Ali-Eldin, J. Tordsson, and E. Elmroth. "An adaptive hybrid elasticity controller for cloud infrastructures". In: *In IEEE Network Operations and Management Symposium.* IEEE, 2012, pp. 204–212.

[Alm14] A.L. Almudevar. *Approximate Iterative Algorithms.* CRC Press, Inc. Boca Raton, FL, USA, 2014. ISBN: 0415621542 9780415621540.

[An+08] B. An, K.M. Sim, L.G. Tang, C.Y. Miao, Z.Q. Shen, and D.J. Cheng. "Negotiation Agents' Decision Making Using Markov Chains". In: *Rational, Robust, and Secure Negotiations in Multi-Agent Systems.* Ed. by T. Ito, H. Hattori, M. Zhang, and T. Matsuo. Vol. 89. Studies in Computational Intelligence. Springer, 2008, pp. 3–23.

[An+10] B. An, V. Lesser, D. Irwin, and M. Zink. "Automated negotiation with decommitment for dynamic resource allocation in cloud computing". In: *Proceedings of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010).* Toronto, Canada, May 2010, pp. 981–988.

[An+16] B. An, N. Gatti, and V. Lesser. "Alternating-offers bargaining in one-to-many and many-to-many settings". In: *Annals of Mathematics and Artificial Intelligence* 77.1 (2016), pp. 67–103.

[An11] Bo An. "Automated Negotiation for Complex Multi-Agent Resource Allocation". PhD dissertation. University of Massachusetts - Amherst, USA, 2011.

[And+02] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer. "Setihome: an experiment in public-resource computing". In: *Communications of the ACM* 45.11 (2002), pp. 56–61.

[And+04] S. Androutsellis-Theotokis and D. Spinellis. "A survey of peer-to-peer content distribution technologies". In: *ACM Computing Surveys* 36 (2004), pp. 335–371.

[And+06] P.A. Anderson, C. Christensen, and B. Allen. "Grid resource management—designing a runtime system for volunteer computing". In: *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing (SC'06)*. ACM Press, New York, NY, USA, 2006.

[And04] David Anderson. "BOINC: a system for public-resource computing and storage". In: *Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing GC'04*. Pittsburgh, USA: ACM Press, New York, NY, USA, 2004, pp. 365–372.

[Ang+10] C. Anglano, M. Canonico, and M. Guazzone. "The ShareGrid Peer-to-Peer Desktop Grid: Infrastructure, Applications, and Performance Evaluation". In: *Journal of Grid Computing* 8.4 (Dec. 2010), pp. 543–570.

[Ara+16] H. Arabnejad, P. Jamshidi, G. Estrada, N. El-Ioini, and C. Pahl. "An Auto-Scaling Cloud Controller Using Fuzzy Q-Learning - Implementation in OpenStack". In: *In book of Service-Oriented and Cloud Computing, the 5th IFIP WG 2.14 European Conference, ESOCC 2016, Vienna, Austria* 9846 (2016), pp. 152–167.

[Ard+12] D. Ardagnaa, S. Casolari, M. Colajanni, and B. Panicucci. "Dual time-scale distributed capacity allocation and load redirect algorithms for cloud systems". In: *Journal of Parallel and Distributed Computing* 72.6 (2012), pp. 796–808.

[Ass+07] M. D. Assunçao, W. Streitberger, T. Eymann, and R. Buyya. "Enabling the simulation of service oriented computing and provisioning policies for autonomic utility grids". In: *In Proceedings of the 4th International Workshop on Grid Economics and Business (GECON'07)*. Rennes, France, 2007, pp. 136–149.

[Bal+08] Z. Balaton et al. "EDGeS: The Common Boundary Between Service And Desktop Grids". In: *Grid Computing: Achievements and Prospects*. Ed. by S.Gorlatch, P.Fragopoulou, and T. Priol. Springer, 2008, pp. 37–48.

[Bas+75] F. Baskett, K. Chandy, R. Muntz, and F. Palacios. "Open, closed, and mixed networks of queues with different classes of customers". In: *Journal of the ACM* 22.2 (1975), pp. 248–260.

[Bec+08a] S. Bechhofer, Y. Yesilada, R. Stevens, S. Jupp, and B. Horan. "Using ontologies and vocabularies for dynamic linking". In: *Internet Computing* 12.3 (2008), pp. 32–39.

[Bec+08b] D. Beckett and T. Berners-Lee. *Turtle-Terse RDF triple language*. 2008. URL: http://www.w3.org/TeamSubmission/turtle/.

[Ber+01] T. Berners-Lee, J. Hendler, and O. Lassila. “The Semantic Web”. In: *Scientific American* (2001).

[Bha+03] R. Bhagwan, S. Savage, and G. Voelker. “Understanding availability”. In: *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS ’03).* Berkeley, CA, USA, Feb. 2003, pp. 256–267.

[Bha+08] S.M.S. Bhanu and N.P. Gopalan. “A hyper-heuristic approach for efficient resource scheduling in grid”. In: *International Journal of Computers, Communications & Control* 3.3 (2008), pp. 249–258.

[Bia+10] D. Bianchini, V. DeAntonellis, and M. Melchiori. “P2P-SDSD: On-the-Fly Service-Based Collaboration in Distributed Systems”. In: *International Journal of Metadata, Semantics and Ontologies* 5.3 (2010), pp. 222–237.

[Bod01] M. Boden. “A guide to recurrent neural networks and backpropagation, The DALLAS project. Report from the NUTEK supported project AIS-8, SICS.Holst: Application of Data Analysis with Learning Systems”. In: (2001).

[Bor97] Willem Nico Borst. “Construction of engineering ontologies for knowledge sharing and reuse”. PhD dissertation. University of Twente, Enschede, Netherlands, 1997.

[Bru+12] R. Brunner, A.C. Caminero, F.O. Rana, F. Freitag, and L. Navarro. “Network-aware summarisation for resource discovery in P2P-content networks”. In: *Future Generation Computer Systems* 28 (2012), pp. 563–572.

[Buy+00] R. Buyya, D. Abramson, and J. Giddy. “NimrodG: An Architecture for a Resource Management and Scheduling System in a Global Computational Grid”. In: *In Proceedings of the HPC ASIA’2000, the 4th International Conference on High Performance Computing in Asia-Pacific Region, Beijing, China, IEEE Computer Society Press.* May 2000.

[Buy+01a] R. Buyya, D. Abramson, and J. Giddy. “An Economy Grid Architecture for Service-Oriented Grid Computing”. In: *In the 10th IEEE International Heterogeneous Computing Workshop (HCW’01), with (IPDPS’01).* California, USA, Apr. 2001.

[Buy+01b] R. Buyya, H. Stockinger, J. Giddy, and D. Abramson. “Economic Models for Management of Resources in Grid Computing”. In: *CoRR* (Dec. 2001).

[Buy+01c] R. Buyya and S. Vazhkudai. “Compute power market: towards a market-oriented grid”. In: *Proceedings of the 1st International Symposium on Cluster Computing and the Grid, (CCGRID ’01).* Brisbane, Qld: IEEE Computer Society, May 2001, pp. 574–581.

[Buy+02] R. Buyya, D. Abramson, J. Giddy, and H. Stockinger. “Economic models for resource management and scheduling in grid computing”. In: *Concurrency and Computation: Practice and Experience* 14.13 (2002), pp. 1507–1542.

[Buy+05] R. Buyya, D. Abramson, and S. Venugopal. “The Grid Economy”. In: *Proceedings of the IEEE on Grid Computing.* Vol. 93. 9. IEEE, Feb. 2005, pp. 698–714.

[Cer+12] C. Cerin and G. Fedak. *Desktop Grid Computing.* Ed. by C. Cerin and G. Fedak. Chapman & Hall/CRC Numerical Analysis and Scientific Computing Series. Chapman and Hall/CRC, 2012. ISBN: 9781439862148.

[Cha+02] F.S. Chapin, P.A. Matso, and H.A. Mooney. *Principles of Terrestrial Ecosystem Ecology.* Springer-Verlag New York, 2002.

[Cha+03] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker. “Making gnutella-like P2P systems scalable”. In: *In Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications SIGCOMM.* Karlsruhe, Germany: IEEE, 2003, pp. 407–418.

[Cha+04] A. Chakravarti, G. Baumgartner, and M. Lauria. “Application specific scheduling for the organic grid”. In: *Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing (GRID’04).* IEEE Computer Society, 2004, pp. 146–155.

[Cha+05] A. Chakravarti, G. Baumgartner, and M. Lauria. “The organic grid: self organising computation on a peer-to-peer network”. In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 35.3 (May 2005), pp. 373–384.

[Cha+10] R.S. Chang and M.S. Hu. “A resource discovery tree using bitmap for grids”. In: *Future Generation Computer Systems* 26.1 (2010), pp. 29–37.

[Che+11a] N. Chergui and S. Chikhi. “A Scalable Hybrid Architecture based-SKOS Ontology for Resource Discovery in Grid”. In: *Proceeding of the International Conference on Information and Communication Systems (ICICS’11).* Irbid, Jordan, May 2011, pp. 113–118.

[Che+11b] N. Chergui and S. Chikhi. “Toward an Efficient and Scalable Architecture Based on SKOS Ontology for Resource Discovery in Grid”. In: Guelma, Algeria, Apr. 2011, pp. 229–240.

[Che+11c] N. Chergui, S. Chikhi, and M.T. Kechadi. “An Architecture based-SKOS Ontology for, Scalable, Efficient and Fault Tolerant Grid Resource Discovery”. In: Hamammet, Tunisia: IEEE, Dec. 2011, pp. 78–83.

[Che+15] N. Chergui, S. Chikhi, and M.T. Kechadi. “Semantic grid resource discovery based on SKOS ontology. (In press) Accepted in 2015”. In: *International Journal of Grid and Utility Computing. Inderscience* (2015).

[Che+17] N. Chergui, M.T. Kechadi, and S. Chikhi. “Scalability-aware mechanism based on workload prediction in ultra-peer networks(In press)”. In: *Peer-to-Peer Networking and Applications. Springer* (2017).

[Chi+03] A. Chien, B. Calder, S. Elbert, and K. Bhatia. “Entropia: Architecture and Performance of an Enterprise Desktop Grid System”. In: *Journal of Parallel and Distributed Computing* 63.5 (May 2003), pp. 597–610.

[Chi+04] A. Chien, B. Calder, S. Elbert, and K. Bhatia. “Resource management in the entropia system”. In: *Grid resource management*. Ed. by J. Nabrzyski, J.M. Schopf, and J. Weglarz. Kluwer Academic Publishers Norwell, MA, USA, 2004, pp. 431–450.

[Chi03] Andrew Chien. “Architecture of a commercial enterprise desktop grid: the entropia system”. In: *Grid Computing: Making the Global Infrastructure a Reality*. Ed. by F. Berman, G. Fox, and T. Hey. John Wiley & Sons, 2003. Chap. 12, pp. 337–350.

[Cho+01] E.K. P. Chong and S.H. Zak. *An Introduction to Optimization. 2nd edition*. A Wiley-lnterscience Publication JOHN WILEY & SONS, INC., 2001.

[Cho+07] S. Choi, H. Kim, E. Byun, M. Baik, S. Kim, C. Park, and C. Hwang. “Characterizing and classifying desktop grid”. In: *7th IEEE International Symposium on Cluster Computing and the Grid. CCGRID’07*. IEEE, 2007, pp. 743–748.

[Cho+08] S. Choi, R. Buyya, H. Kim, E. Byun, M. Baik, J. Gil, and C. Park. *A taxonomy of desktop grids and its mapping to state-of-the art systems*. Tech. rep. Grid Computing and Distributed Systems Laboratory, University of Melbourne, Australia, Feb. 2008.

[Chu+08] L. Chunlin and L. Layuan. “A new optimal approach for multiple optimisation objectives grid resource allocation and scheduling”. In: *International Journal of Systems Science* 39.12 (2008), pp. 1127–1138.

[Chu+09] L. Chunlin, Z. J. Xiu, and L. Layuan. “Resource scheduling with conflicting objectives in grid environments: Model and evaluation”. In: *Journal of Network and Computer Applications* 30 (2009), pp. 760–769.

[Chu+12] L. Chunlin and L. Layuan. “An efficient resource allocation for maximizing benefit of users and resource providers in ad hoc grid environment”. In: *Information Systems Frontiers* 14 (2012), pp. 987–998.

[Cir+06] W. Cirne, F. Brasileiro, N. Andrade, L. Costa, A. Andrade, R. Novaes, and M. Mowbray. “Labs of the World, Unitë. In: *Journal of Grid Computing* 4.3 (2006), pp. 225–246.

[Con08] Nicolae-Zoran Constantinescu-Fulop. “A Desktop Grid Computing Approach for Scientific Computing and Visualisation”. PhD dissertation. Norwegian University of Science and Technology, 2008.

[Con11] Hannah Constantin. “Markov Chains and Queueing Theory”. In: 2011.

[Cor+06] O. Corcho, P. Alper, L. Kotsiopoulos, P. Missier, and C. Goblel. “An overview of S-OGSA: A reference semantic grid architecture”. In: *Journal of Web Semantics* 4.1 (2006), pp. 102–115.

[Den+10] L. Denoeud-Belgacem, E. Gourdin, R. Krishnaswamy, and A. Ouorou. “Combinatorial auctions for exchanging resources over a grid network”. In: *Journal of Annals of Telecommunications* 65 (2010), pp. 689–704.

[Dep+14] W. Depoorter, K. Vanmechelen, and J. Broeckhove. “Advance reservation, co-allocation and pricing of network and computational resources in grids”. In: *Future Generation Computer Systems* 41 (2014), pp. 1–15.

[Din+00] R. Dingledine, M. J. Freedman, and D. Molnar. “The Free Haven project: Distributed anonymous storage service”. In: *International workshop on Designing privacy enhancing technologies: design issues in anonymity and un-observability.* Springer-Verlag, July 2000, pp. 67–95.

[Ebe+05] J. Eberspcher and R. Schollmeier. “First and Second Generation of Peer-to-Peer Systems”. In: *Peer-to-Peer Systems and Applications.* Ed. by R. Steinmetz and K.Wehrle. Vol. 3485. Lecture Notes in Computer Science. Springer, 2005. Chap. 5, pp. 35–56.

[EMI] EMI. *European Middleware Initiative EMI.* URL: http://www.eu-emi.eu/middleware (visited on 06/20/2016).

[Far+98] P. Faratin, C. Sierra, and N.R. Jennings. “Negotiation decision functions for autonomous agents”. In: *nternational Journal of Robotics and Autonomous Systems* 24.3-4 (1998), pp. 159–182.

[Fed+01] G. Fedak, C. Germain, V. Neri, and F. Cappello. “XtremWeb: a generic global computing system”. In: *Proceedings of 1st IEEE/ACM International Symposium on Cluster Computing and the Grid.* IEEE Press, 2001, pp. 582–587.

[Fed+08] G. Fedak et al. “Edges: a bridge between desktop grids and service grids”. In: *Proceedings of the the 3rd ChinaGrid Annual Conference (chinagrid ’08), CHINAGRID.* Washington, DC, USA: IEEE, 2008, pp. 3–9.

[Fed10] G. Fedak. “Recent advances and research challenges in desktop grid and volunteer computing”. In: *Grids, P2P and Services Computing.* Ed. by F. Desprez, V. Getov T. Priol, and R. Yahyapour. Springer US, 2010, pp. 171–185.

[Fed15] Gilles Fedak. “Contributions to Desktop Grid Computing From High Throughput Computing to Data-Intensive Sciences on Hybrid Distributed Computing Infrastructures”. Ability to conduct researches. University of Lyon, 2015.

[Fel06] C. Fellbaum. “WordNet(s), http://wordnet.princton.edu/”. In: *Encyclopedia of Language and Linguistics* 13 (2006), pp. 665–670.

[Fer13] Stefano Ferretti. “Gossiping for Resource Discovering: an Analysis based on Complex Network Theory”. In: *Future Generation Computer Systems (FGCS)* 29.6 (2013), pp. 1631–1644.

[Fos+01] I. Foster, C. Kesselman, and S. Tuecke. “The anatomy of the grid: Enabling scalable virtual organisations”. In: *International Journal of High Performance Computing Applications* 15.3 (Aug. 2001), pp. 200–222.

[Fos+02] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. *The physiology of the grid: An open grid services architecture for distributed systems integration*. Tech. rep. 2002. URL: `http://toolkit.globus.org/alliance/publications/papers/ogsa.pdf` (visited on 06/07/2016).

[Fos+05] I. Foster et al. *The open grid services architecture OGSA*. Tech. rep. 2005. URL: `https://www.ogf.org/documents/GFD.30.pdf` (visited on 06/07/2016).

[Fre+02] J. Michael Freedman and R. Vingralek. “Efficient Peer-To-Peer Lookup Based on a Distributed Trie”. In: *Proceeding of the 1st International Workshop on Peer-to-Peer Systems (IPTPS’02), MIT*. Mar. 2002.

[Gar+10a] P. Garbacki, D.H.J. Epema, and M. van Steen. “The Design and Evaluation of a Self-Organizing Superpeer Network”. In: *IEEE Transactions on Computers* 59.3 (2010), pp. 317–331.

[Gar+10b] S. K. Garg, R. Buyya, and H. J. Siegel. “Time and cost trade-off management for scheduling parallel applications on Utility Grids”. In: *Future Generation Computer Systems* 26 (2010), pp. 1344–1355.

[Gar+11] S. K. Garg and R. Buyya. “Market-Oriented Resource Management and Scheduling: A Taxonomy and Survey”. In: *Cooperative Networking*. Ed. by M. S. Obaidat and S. Misra. John Wiley & Sons, 2011. Chap. 14, pp. 277–305.

[Gel+10] E. Gelenbe and I. Mitrani. *Analysis and Synthesis of Computer Systems*. Imperial College Press London, UK, ISBN:1848163959 9781848163959, 2010.

[Gel75] E. Gelenbe. “On approximate computer system models”. In: *Journal of the ACM* 22.2 (1975), pp. 261–269.

[Gel90] E. Gelenbe. “Performance Analysis of the Connection Machine”. In: *ACM SIGMETRICS Performance Evaluation Review* 18.1 (1990), pp. 183–191.

[gLi] gLite. *gLite - Lightweight Middleware for Grid Computing*. URL: `http://grid-deployment.web.cern.ch/grid-deployment/glite-web/` (visited on 06/02/2016).

[Gom+04] A. Gomez-Perez, M. Fernandez-Lopez, and O. Corcho. “Ontological engineering with examples from the areas of knowledge management”. In: *Ontological Engineering*. Advanced Information and Knowledge Processing. Springer, 2004.

[Gri+09] C.M. Grinstead and J.L. Snell. *Introduction to Probability.* 2009. Chap. 11, pp. 405–470.

[Gru93] T.G. Gruber. “A Translation Approach to Portable Ontology Specifications”. In: *Knowledge Acquisition* 5.2 (1993), pp. 199–220.

[Gua+09] N. Guarino, D. Oberle, and S. Staab. “What Is an Ontology?” In: *Handbook on Ontologies. Part of the series International Handbooks on Information Systems.* Ed. by S. Staab and R. Studer. Springer, May 2009, pp. 1–17.

[Gui09] Xavier Vilajosana Guillen. “Evaluating The Performance of Mluti-Objective Particle Swarm Optimisation Algorithms”. PhD dissertation. Universitat Oberta de Catalunya, Spain, 2009.

[Hag+14] S. Haghtalabi, R. Javidan, and A. Harounabadi. “A New Resource Allocation Model for Grid Networks based on Bargaining in a Competitive Market”. In: *Journal of Soft Computing and Applications* (2014), pp. 1–17.

[Haq+14] A. Haque, S.M. Alhashmi, and R. Parthiban. “Identifying and Modeling the Strengths and Weaknesses of Major Economic Models in Grid Resource Management”. In: *Journal Grid Computing* 12.2 (June 2014), pp. 285–302.

[Haq+15] A. Haque, S.M. Alhashmi, and R. Parthiban. “An optimization-based adaptive resource management framework for economic Grids: A switching mechanism”. In: *Future Generation Computer Systems* 47 (2015), pp. 48–59.

[Har+04] S. Hariri and M. Parashar. *Tools and Environments for Parallel and Distributed Computing.* John Wiley & Sons, 2004. ISBN: 9780471474845.

[Hay94] Simon Haykin. *Neural Networks: A Comprehensive Foundation.* 1994. ISBN: 0023527617.

[Hod00] Gail Hodge. *Systems of Knowledge Organization for Digital Libraries: Beyond Traditional Authority Files. Council on Library and Information Resources.* Tech. rep. 2000. URL: `https://www.clir.org/pubs/reports/pub91/pub91.pdf` (visited on 09/07/2016).

[Hor+03] I. Horrocks, P.F. Patel-Schneider, and F. Van-Harmelen. “From SHIQ and RDF to OWL: The making of a web ontology language”. In: *Journal of Web Semantics* 1.1 (2003), pp. 7–26.

[Hor+04] M. Horridge, H. Knublauch, R. Rector, R. Stevens, and C. Wroe. *A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools Edition 1.0.* 2004.

[Hri+05] M. Hristakeva and D. Shrestha. “Different Approaches to Solve the 0/1 Knapsack Problem”. In: *In Proceedings of the 38th Midwest Instruction and Computing Symposium.* Washington, DC, USA, Apr. 2005.

[Hua+05] B.Q. Huang and M.T. Kechadi. "A recurrent neural network recogniser for online recognition of handwritten symbols". In: *Proceedings of the 7th International Conference on Enterprise Information Systems(ICEIS)'05.* Miami, USA, May 2005, pp. 27–34.

[Hua+06] B.Q. Huang, T. Rashid, and M.T. Kechadi. "Multi-Context Recurrent Neural Network for Time Series Applications". In: *International Journal of Computational Intelligence* 3.1 (2006), pp. 45–54.

[Iam+01] A. Iamnitchi and I.T. Foster. "On fully decentralised resource discovery in grid environments". In: *Proceeding of the 2nd International Workshop on Grid Computing.* London, UK: Springer-Verlag, 2001, pp. 51–62.

[Iam+02] A. Iamnitchi, M. Ripeanu, and I.T. Foster. "Locating data in (small-world) peer-to peer scientific collaborations". In: *Proceeding of the International Workshop on Peer-to-Peer Systems (IPTPS'01).* Honolulu, Hawaii, USA: Springer-Verlag, 2002, pp. 232–241.

[Iam+04] A. Iamnitchi and I. T. Foster. "A Peer-to-Peer Approach to Resource Location in Grid Environments". In: *Grid resource management: state of the art and future trends.* Ed. by J. Nabrzynski, J. M. Schopf, and J. Weglarz. Kluwer Academic Publishers Norwell, MA, USA, 2004, pp. 413–429.

[Iqb+11] W. Iqbal, M.N. Dailey, D. Carrera, and P. Janecek. "Adaptive Resource Provisioning for Read Intensive Multi-tier Applications in the Cloud". In: *Future Generation Computer Systems* 27.6 (2011), pp. 871–879.

[Ism07] L. Ismail. "Dynamic resource allocation mechanisms for grid computing environment". In: *Proceedings of 3rd IEEE International Conference on Testbeds and Research Infrastructure for the Development of Networks and Communities.* IEEE, 2007, pp. 1–5.

[Jav+15] S. Javanmardia, M. Shojafar, S. Shariatmadari, and S.S. Ahrabi. "FR TRUST: A Fuzzy Reputation Based Model for Trust Management in Semantic P2P Grids". In: *International Journal of Grid and Utility Computing* 6.1 (2015), pp. 57–66.

[Jea+08] E. Jeanvoine and C. Morin. "RW-OGS: An optimized random walk protocol for resource discovery in large scale dynamic grids". In: *the 9th IEEE/ACM International Conference on Grid Computing.* Washington, DC, USA: IEEE, 2008, pp. 168–175.

[Jes+06] G.P. Jesi, A. Montresor, and O. Babaoglu O. "Proximity-aware superpeer overlay topologies". In: *Proceeding of the 2nd IEEE Workshop on Self-Managed Networks, Systems, and Services.* Dublin, Ireland: IEEE, June 2006, pp. 43–57.

[Jup+09] S. Jupp, S.S.Bechhofer, and R. Stevens. "A flexible API and editor for SKOS". In: *Proceeding of the 6th European Semantic Web Conference, (ESWC) In book of the Semantic Web: Research and Applications.* Heraklion, Crete, Greece: Spriger, 2009, pp. 506–520.

[Kal+03] Y. Kalfoglou and M. Schorlemmer. “Ontology mapping: the state of the art”. In: *The Knowledge Engineering Review* 18.1 (2003), pp. 1–31.

[Kal+04] L. Kale, S. Kumar, M. Potnuru, J. DeSouza, and S. Bandhakavi. “Faucets: efficient resource allocation on the computational grid”. In: *In proceeding of the International Conference on Parallel Processing (ICPP’04)*. Vol. 1. Aug. 2004, pp. 396–405.

[Kam+12] G.K. Kamalam and V.M. Bhaskaran. “New enhanced heuristic min-mean scheduling algorithm for scheduling metatasks on heterogeneous grid environment”. In: *European Journal of Scientific Research* 70.3 (2012), pp. 423–430.

[Kav13] Hanna Kavalionak. “Autonomous Resource Management for Cloud-Assisted Peer-To-Peer Based Services”. PhD dissertation. Universita degli Studi di Trento, Italy, 2013.

[Ken53] David George Kendall. “Stochastic Processes Occurring in the Theory of Queues and their Analysis by the Method of the Embedded Markov Chain”. In: *The Annals of Mathematical Statistics* 24.3 (1953), pp. 338–354.

[Ken94] Gavin Kennedy. *NField Guide to Negotiation*. Cambridge, MA: Harvard Business School Press, 1994.

[Ker+91] G. Kersten, W. Michalowski, S. Szpakowicz, and Z. Koperczak. “Restructurable representations of negotiation”. In: *Journal Management Science* 37.10 (Oct. 1991), pp. 1269–1290.

[Kha+15] A.M. Khan, U.C. Buyuksahin, and F. Freitag. “Incentive-based resource assignment and regulation for collaborative cloud services in community networks”. In: *Journal of Computer and System Sciences* 81.8 (2015), pp. 1479–1495.

[Kim+07] J.S. Kim, B. Nam, M.A. Marsh, P.J. Keleher, B. Bhattacharjee, D. Richardson, D. Wellnitz, and A. Sussman. *Creating a robust desktop grid using peer-to-peer services*. Long Beach, CA, Mar. 2007.

[Kle+05] M. Kleis, E.K. Lua, and X.M. Zhou. “Hierarchical peer-to-peer networks using lightweight SuperPeer topologies”. In: *Proceeding of 10th IEEE Symposium on Computers and Communications (ISCC’05)*. Murcia, Spain: IEEE, June 2005, pp. 143–148.

[Kle75] L. Kleinrock. *Queueing systems*. Vol. 1. Wiley-Interscience, 1975. ISBN: 0471491101.

[Kol+11] J. Kolodziej and F. Xhafa. “Modern approaches to modeling user requirements on resource and task allocation in hierarchical computational Grids”. In: *International Journal of Applied Mathematics and Computer Science* 21.2 (2011), pp. 243–257.

[Kon+04] D. Kondo, M. Taufer, C. L. Brooks, H. Casanova, and A. A. Chien. “Characterizing and evaluating desktop grids: An empirical study”. In: *Proceeding of 18th International Parallel and Distributed Processing Symposium*. Sante Fe, New Mexico: IEEE, Apr. 2004.

[Kon+15] Y. Kong, M. Zhang, D. Ye, and X. Luo. “A Negotiation Method for Task Allocation with Time Constraints in Open Grid Environments”. In: *Next Frontier in Agent-based Complex Automated Negotiation, Volume 596 of the series Studies in Computational Intelligence.* Ed. by K. Fujita, T. Ito, M. Zhang, and V. Robu. Springer, 2015. Chap. 2, pp. 19–36.

[Ksh+11] A. D. Kshemkalyani and M. Singhal. *Distributed Computing: Principles, Algorithms, and Systems.* Cambridge University Press, 2011. ISBN: 0521876346.

[Lai+15] Y. Lai, Y. Chen, Z. Liu, Z. Yang, and X. Li. “On monitoring and predicting mobile network traffic abnormality”. In: *Simulation Modelling Practice and Theory, Special Issue on Resource Management in Mobile Clouds* 50 (2015), pp. 176–188.

[Law+03] C. Law and K. Siu. “Distributed construction of random expander networks”. In: *22nd Annual Joint Conference of the IEEE Computer and Communications (INFOCOM ’03).* Vol. 3. IEEE, Mar. 2003, pp. 2133–2143.

[Li+08] Y.M. Li, Y. Tan, and Y.P. Zhou. “Analysis of Scale Effects in Peer-to-Peer Networks”. In: *IEEE/ACM Transactions on Networking* 16.3 (2008), pp. 590–602.

[Li+10] J.S. Li and C.H. Chao. “An Efficient Superpeer Overlay Construction and Broadcasting Scheme Based on Perfect Difference Graph”. In: *IEEE Transactions on Parallel and Distributed Systems* 21.5 (2010), pp. 594–606.

[Li+11] Z. Li, R. Wang, and Z. Han. “Traffic Prediction-based Routing Algorithm over Structured P2P Networks”. In: *Journal of China Universities of Posts and Telecommunications* 18.1 (2011), pp. 23–27.

[Li+14] M. Li, N. Xiong, B. Yang, Z. Li, J. H. Park, and C. Lee. “Posted price model based on GRS and its optimization for improving grid resource sharing efficiency”. In: *Telecommunication Systems* 55 (2014), pp. 71–79.

[Li10] Juan Li. “Grid resource discovery based on semantically linked virtual organisations”. In: *Future Generation Computer Systems (FGCS)* 26.3 (2010), pp. 361–373.

[Lia+04] J. Liang, R. Kumar, and K.W. Ross. “The Kazaa Overlay: A measurement study”. In: *Proceedings of the IEEE Annual Computer Communications Workshop.* Florida, USA, 2004.

[Liu+13] M. Liu, H. Erkki, and M. Ylianttila. “An efficient selection algorithm for building a super-peer overlay”. In: *Journal of Internet Services and Applications* 4.4 (2013), pp. 292–308.

[Liu14] Meirong Liu. “Efficient Super-Peer Based Coordinated Service Provision”. PhD dissertation. UNIVERSITY OF OULU, Finland, 2014.

[Llo+14] J. Lloret, M. Garcia, J. Tomas, and J.J.P.C. Rodrigues. “Architecture and protocol for inter cloud communication”. In: *Information Sciences* 258 (2014), pp. 434–451.

[Loo+09] B.T. Loo, T. Condie, M. Garofalakis, D.E. Gay, J. M. Hellerstein, P. Maniatis, R. Ramakrishnan, T. Roscoe, and I. Stoica. “Declarative Networking”. In: *Communications of the ACM* 52.11 (2009), pp. 87–95.

[Los+04] A. Loser, F. Naumann, W. Siberski, W. Nejdl, and U. Thaden. “Semantic overlay clusters within super-peer networks”. In: *Databases, Information Systems, and Peer-to-Peer Computing. In Proceeding of 1st International Workshop on Databases, Information Systems, and Peer-to-Peer Computing (DBISP2P).* Toronto, Canada: Springer, 2004, pp. 33–47.

[Lua+05] E.K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim. “A survey of peer-to-peer content distribution technologies”. In: *IEEE Communications Surveys and Tutorials* 7.2 (2005), pp. 72–93.

[Mac04] J. MacGregor-Smith. “Optimal Design and Performance Modelling of M/G/1/k Queueing Systems”. In: *Mathematical and Computer Modelling* 39.9-10 (2004), pp. 1049–1081.

[Mah+10] A.N. Mahmood, C. Leckie, J. Hu, Z. Tari, and M. Atiquzzaman. “Network traffic analysis and scada security”. In: *Handbook of Information and Communication Security* (2010), pp. 383–405.

[Mas+05] C. Mastroianni, D. Talia, and O. Verta. “A super-peer model for resource discovery services in large-scale Grids”. In: *Future Generation Computer Systems (FGCS)* 21.8 (2005), pp. 1235–1248.

[Mes+16] V.R. Messias, J.C. Estrella, R. Ehlers, M.J. Santana, R.C. Santana, and S. Reiff-Marganiec. “Combining Time Series Prediction Models Using Genetic Algorithm to Auto-scaling Web Applications Hosted in the Cloud Infrastructure”. In: *Neural Computing and Applications* 27.8 (2016), pp. 2383–2406.

[Mil+03] D.S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. *The open grid services architecture OGSA*. Tech. rep. 2003. URL: http://www.hpl.hp.com/techreports/2002/HPL-2002-57R1.pdf (visited on 06/07/2016).

[Mil+09] A. Miles and S. Bechhofer. *SKOS simple knowledge organisation system reference*. 2009. URL: http://www.w3.org/TR/skos-reference. (visited on 12/15/2013).

[Miz03] Riichiro Mizoguchi. “Tutorial on Ontological Engineering Part 1: Introduction to Ontological Engineering”. In: *New Generation Computing* 21 (2003), pp. 365–384.

[Mod+11] G. Di Modica, O. Tomarchio, and L. Vita. “Resource and Service Discovery in SOAS: A P2P Oriented Semantic Approach”. In: *International Journal of Applied Mathematics and Computer Science* 21.2 (2011), pp. 285–294.

[Mon+03] A. Montresor, H. Meling, and Z. Babaoglu. “Messor: load-balancing through a swarm of autonomous agents”. In: *Agents and Peer-to-Peer Computing*. Ed. by G. Moro and M. Koubarakis. Vol. 2530. Lecture Notes in Computer Science. Springer, Berlin, 2003, pp. 125–137.

[Mon04] A. Montresor. “A robust protocol for building superpeer overlay topologies”. In: *Proceeding of the 4th International Conference on Peer-to-Peer Computing (P2P’04)*. IEEE, Aug. 2004, pp. 202–209.

[Moo+01] B. Moon, H.V. Jagadish, C. Faloutsos, and J.H. Saltz. “Analysis of the Clustering Properties of the Hilbert Space-Filling Curve”. In: *IEEE Transactions on Knowledge and Data Engineering* 13.1 (2001), pp. 124–141.

[Moo75] G.E. Moore. “Progress in digital integrated electronics”. In: *In Proceedings of the IEEE International Electron Devices Meeting (IEDM’75)*. Washington, DC, USA, Dec. 1975, pp. 11–13.

[Naj+15] H.S. Najafi, J. Pourqasem, and S.A. Edalatpanah. “The Use of Super Node to Process Query in Peer-to-Peer Networks”. In: *Digital Technologies* 1.1 (2015), pp. 28–32.

[Nav+14] N.J. Navimipour, A.M. Rahmani, A.H. Navin, and M.H. Zadeh. “Resource discovery mechanisms in grid systems: A survey”. In: *Journal of Network and Computer Applications* 41 (2014), pp. 389–410.

[Naz+03] A. Nazareno, W. Cirne, F. Brasileiro, and P. Roisenberg. “OurGrid: An approach to easily assemble Grids with equitable resource sharing”. In: *Proceedings of the 9th Workshop on Job Scheduling Strategies for Parallel Processing (JSSPP’03)*. Seattle,WA, USA, 2003, pp. 61–86.

[Naz+07] A. Nazareno, F.V. Brasileiro, W. Cirne, and M. Mowbray. “Automatic grid assembly by promoting collaboration in Peer-to-Peer grids”. In: *Parallel and Distributed Computing* 67.8 (2007), pp. 957–966.

[Nec+91] R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W.R. Swartout. “Enabling Technology for Knowledge Sharing”. In: *AI Magazine* 12.3 (1991), pp. 36–56.

[Nej+02] W. Nejdl, B. Wolf, C. Qu, S. Decker, A. Naeve, M. Nilsson, M. Palmer, and T. Risch. “EDUTELLA: A P2P Networking Infrastructure Based on RDF”. In: *Proceeding of the 11th international conference on World Wide Web*. ACM, May 2002, pp. 604–615.

[Nej+03a] W. Nejdl, M. Wolpers, W. Siberski, C. Schmitz, M. Schlosse, I.Brunkhorst, and A.Loser. “Super-peer-based routing and clustering strategies for RDF-based peer-to-peer networks”. In: *Proceedings of the 12th International Conference on World Wide Web WWW*. 2003, pp. 536–543.

[Nej+03b] W. Nejdl, M. Wolpers, W. Siberski, C. Schmitz, M.T. Schlosser, I. Brunkhorst, and A. Loser. “Super-peer-based routing and clustering strategies for RDF-based peer-to-peer networks”. In: *Proceeding of the 12th International World Wide Web Conference (WWW’03)*. ACM, 2003, pp. 536–543.

[Neu94] B.C. Neuman. “Scale in Distributed Systems”. In: *Readings in Distributed Computing Systems* (1994), pp. 463–489.

[Nez+16] A. Nezarat and G. Dastghaibyfard. “A game theoretical model for profit maximization resource allocation in cloud environment with budget and deadline constraints”. In: *Journal of Supercomputing* 72.12 (Dec. 2016), pp. 4737–4770.

[Nie98] Jakob Nielsen. *Nielsen’s law of Internet bandwidth*. Apr. 1998. URL: http://www.useit.com/alertbox/980405.html (visited on 12/12/2016).

[Our] OurGrid. *OurGrid*. URL: http://www.ourgrid.org/overview.php (visited on 08/03/2016).

[Pad+10] A. Padmanabhan, S. Ghosh, and S. Wang. “A Self-Organised grouping SOG framework for efficient grid resource discovery”. In: *Journal of Grid Computing* 8.3 (2010), pp. 365–389.

[Par+08] P. Parpas and B. Rustem. “A Pricing Mechanism for Resource Management in Grid Computing”. In: *Computational Economics* 31.4 (2008), pp. 381–395.

[Par+13] D. Park, J. Yu, J. Park, and M.S. Kim. “NetCube: a comprehensive network traffic analysis model based on multidimensional OLAP data cube”. In: *International Journal of Network Management* 23.2 (2013), pp. 101–118.

[Pip10] Gregor Pipan. “Use of the TRIPOD overlay network for resource discovery”. In: *Future Generation Computer Systems (FGCS)* 26 (2010), pp. 1257–1270.

[Pir+12] G. Pirro, D. Talia, and P. Trunfio. “A DHT-based semantic overlay network for service discovery”. In: *Future Generation Computer Systems (FGCS)* 28.4 (2012), pp. 689–707.

[Pou+06] B. Pourebrahimi, K. Bertels, G. M. Kandru, and S. Vassiliadis. “Market-based Resource Allocation in Grids”. In: *Proceedings of the Second IEEE International Conference on e- Science and Grid Computing*. Dec. 2006.

[Pyu+04] Y.J. Pyun and D. S. Reeves. “Constructing a balanced, (log(N)/loglog(N))-diameter super-peer topology for scalable P2P systems”. In: *Proceedings of the 4th International Conference on Peer-to-Peer Computing*. Zurich, Switzerland: IEEE, Aug. 2004, pp. 210–218.

[QAD] QADPZ. *Quite Advanced Distributed Parallel Zystem*. URL: http://qadpz.sourceforge.net/ (visited on 08/03/2016).

[Qia+07] B. Qiao, G. Wang, and K. Xie. “A taxonomy-based approach for constructing semantics based super-peer networks”. In: *Proceeding of International Workshops on Advances in Web and Network Technologies, and Information Management.* Springer, 2007, pp. 122–134.

[Qur+14] M.B. Qureshi et al. “Survey on Grid Resource Allocation Mechanisms”. In: *Grid Computing* 12.2 (2014), pp. 399–441.

[Rag+95] S.V. Raghavan, P.J. Joseph, and G. Haring. “Workload models for multiwindow distributed environments”. In: *Book of Quantitative Evaluation of Computing and Communication Systems* 977 (1995), pp. 314–326.

[Ram+10] K.K. Ramachandran and B.Sikdar. “A queuing model for evaluating the transfer latency of P2P systems”. In: *IEEE Transactions on Parallel and Distributed Systems* 21.3 (2010), pp. 367–378.

[Ram+98] R. Raman, M. Livny, and M. Solomon. “Matchmaking: distributed resource management for high throughput computing, in Proceeding of the 7th IEEE (HPDC)”. In: Washington DC, USA: IEEE, 1998, pp. 140–146.

[Rao09] Singiresu S. Rao. *Engineering Optimization, Theory and Practice. 4th edition.* JOHN WILEY & SONS Inc., Hoboken, New Jersey, 2009.

[Rat+01] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. “A Scalable Content-Addressable Network”. In: *Proceedings of the 2001 ACM SIGCOMM.* San Diego, California USA, Aug. 2001, pp. 161–172.

[Reg+00] O. Regev and N. Nisana. “The POPCORN market. Online markets for computational resources”. In: *Decision Support Systems* 28.1-2 (2000), pp. 177–189.

[Rip01] M. Ripeanu. “Peer-to-peer architecture case study: Gnutella network”. In: *Proceeding of 1st International Conference on Peer-to-Peer Computing.* Vol. 3. IEEE, Aug. 2001, pp. 99–100.

[Rou+05] D.D. Roure, N.R. Jennings, and N.R. Shadbot. “The Semantic Grid: Past, Present”. In: *Proceedings of the IEEE.* Vol. 93. 3. IEEE, Mar. 2005, pp. 669–681.

[Row+01] A. Rowstron and P. Druschel. “Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems”. In: *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware).* Vol. 3. Springer-Verlag, Nov. 2001, pp. 329–350.

[Rub82] Ariel Rubinstein. “Perfect equilibrium in a bargaining model”. In: *Econometrica* 50.1 (Jan. 1982), pp. 97–109.

[Rub85] Ariel Rubinstein. “A bargaining model under incomplete information about time preferences”. In: *Econometrica* 53.5 (1985), pp. 1151–1172.

[Sah07] G.K. Saha. “Web Ontology Language (OWL) and Semantic Web”. In: *Ubiquity* 8 (2007), pp. 1–24.

[SAR+02] S. SAROIU, P.K. GUMMADI, and S.D. GRIBBLE. “A measurement study of Napster and Gnutella as examples of peer-to-peer file sharing systems”. In: *ACM SIGCOMM Computer Communication Review* 32.1 (2002), pp. 82–82.

[Sch+09] S. Schulz, W. Blochinger, and H. Hannak. “Capability-Aware Information Aggregation in Peer-to-Peer Grids Methods, Architecture, and Implementation”. In: *Grid Computing* 7.2 (2009), pp. 135–167.

[Sch01] Rdiger Schollmeier. “A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications”. In: *In Proceedings of the IEEE International Conference on Peer-to-Peer Computing (P2P2001).* Aug. 2001, pp. 101–102.

[Sen+04] S. Sen and J. Wang. “Analyzing Peer-To-Peer Traffic Across Large Networks”. In: *IEEE/ACM Transactions on Networking* 12.2 (2004), pp. 219–232.

[Ser09] Richard Serfozo. “Markov chain”. In: *Basics of Applied Stochastic Processes, Probability and its Applications.* Springer, 2009. Chap. 1.

[She+12] H. Shen and K. Hwang. “Locality-Preserving Clustering and Discovery of Resources in Wide-Area Distributed Computational Grids”. In: *IEEE Transactions on Computers* 61.4 (2012), pp. 458–473.

[Sie+97] C. Sierra, N.R. Jennings, P.Noriega, and S. Parsons. “A framework for argumentation based negotiation”. In: *In Intelligent Agents IV: the 4th International Workshop on Agent Theories Architectures and Languages. Springer.* Dec. 1997, pp. 177–192.

[Sim+06] K.M. Sim and K.F. Ng. “A Relaxed-Criteria Bargaining Protocol for Grid Resource Management”. In: *In Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid Workshops (CCGRIDW’06).* Dec. 2006.

[Sim05] Kwang Mong Sim. “Equilibria, Prudent Compromises, and the Waiting Game”. In: *IEEE Transactions On Systems, Managements and Cybernetics* 35.4 (2005), pp. 712–724.

[SIM06] KWANG MONG SIM. “Relaxed-criteria G-negotiation for Grid Resource Co-allocation”. In: *ACM SIGecom Exchanges, Vol. 6, No. 2.* Dec. 2006, pp. 37–46.

[Sim10] Kwang-Mong Sim. “Grid Resource Negotiation: Survey and New Directions”. In: *IEEE Transactions On Systems, Management and Cybernetics-Part C: Applications and Reviews* 40.3 (May 2010), pp. 245–257.

[Smi+05] J. MacGregor Smith and FRB. Cruz. “The buffer allocation problem for general finite buffer queueing networks”. In: *IIE Transactions on Design and Manufacturing* 37.4 (2005), pp. 343–65.

[Smi80] R. Smith. “The Contract Net Protocol High-Level Communication and Control in a Distributed Problem Solver”. In: *IEEE Transaction on Computer* 4 (1980), pp. 1104–1113.

[Spr+91] M. Springer and P. Makens. “Queueing Models for performance analysis and Selection of Single Station Models”. In: *EJOR* 58 (1991), pp. 123–145.

[Ste+13] S. Steve-Harris and A. Seaborne. *SPARQL 1.1 Query Language*. 2013. URL: http://www.w3.org/TR/sparql11-query/.

[Sto+01] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, and H. Balakrishma. “CHORD: A scalable Peer to Peer Lookup Service for Internet Application”. In: *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM’01)*. San Diego, California USA: ACM, 2001, pp. 149–160.

[Stu+07] G. Stuer, K. Vanmechelen, and J. Broeckhove. “A commodity market algorithm for pricing substitutable Grid resources”. In: *Future Generation Computer Systems* 23.2 (2007), pp. 688–701.

[Stu+08] D. Stutzbach, R. Rejaie, and S. Sen. “Characterizing unstructured overlay topologies in modern P2P file-sharing systems”. In: *IEEE/ACM Transactions on Networking* 16.2 (2008), pp. 267–280.

[Stu+98] R. Studer, V. Benjamins, and D. Fensel. “Knowledge engineering: Principles and methods”. In: *Data and Knowledge Engineering - Special jubilee issue* 25.1-2 (1998), pp. 161–197.

[Swa+96] B. Swartout, R. Patil, K. Knight, and T. Russ. “Toward distributed use of large-scale ontologies”. In: *Proceedings of the 10th Workshop on Knowledge Acquisition for Knowledge-Based Systems*. Nov. 1996, pp. 138–148.

[Tak62] L. Takacs. *Introduction to the theory of queues*. Oxford University Press, New York, 1962.

[Tal+06] D. Talia, P. Trunfio, and J. Zeng. “Peer-to-peer models for resource discovery in large-scale grids: scalable architecture”. In: *High Performance Computing for Computational Science, VECPAR 2006, LNCS*. Vol. 4395. Springer-Verlag, 2006, pp. 66–78.

[Tan+02] T. Tannenbaum, D. Wright, K. Miller, and M. Livny. “Condor: a distributed job scheduler”. In: *Beowulf cluster computing with Linux*. MIT Press Cambridge, 2002. Chap. 12, pp. 307–350.

[Tan+05] S. Tangpongprasit, T. Katagiri, K. Kise, H. Honda, and T. Yuba. “A time-to-live based reservation algorithm on fully decentralised resource discovery in Grid Computing”. In: *Journal of Parallel Computing* 31.6 (2005), pp. 529–543.

[Tan+12] Y.H. Tan, K. Lü, and Y.P. Lin. “Organisation and management of shared documents in super-peer networks based semantic hierarchical cluster trees”. In: *Networking and Applications* 3.5 (2012), pp. 292–308.

[Teh00] Rich Tehrani. *Communications solutions publisher's outlook: As we may communicate*. Jan. 2000. URL: http://www.tmcnet.com/articles/comsol/0100/0100pubout.htm (visited on 12/12/2016).

[Ten+14] H.Y. Teng, C.N. Lin, and R.H. Hwang. "A self-similar super-peer overlay construction scheme for super large-scale P2P applications". In: *Information System Frontier* 16 (2014), pp. 45–58.

[Tor12] Javad Akbari Torkestani. "A distributed resource discovery algorithm for P2P grids". In: *Journal of Network and Computer Applications* 35 (2012), pp. 2028–2036.

[Tri13] Sven Trieflinger. "High Performance Peer-to-Peer Desktop Grid Computing Architecture, Methods, Applications". PhD dissertation. University of Stuttgart, Germany, 2013.

[Urg+08] B. Urgaonkar, P. Shenoy, A. Chandra, P. Goyal, and T. Wood. "Agile dynamic provisioning of multi-tier Internet applications". In: *ACM Transactions on Autonomous and Adaptive Systems* 3 (2008), pp. 1–39.

[Val+04] A. Valarakos, G. Paliouras, V. Karkaletsis, and G. Vouros. "Enhancing ontological knowledge through ontology population and enrichment". In: *Engineering Knowledge in the Age of the Semantic Web*. Ed. by E. Motta, N.R. Shadbolt, A. Stutt, and N. Gibbins. Vol. 3257. Springer, 2004. Chap. 32, pp. 144–156.

[Ven15] Neelanarayanan Venkataraman. "A Context-Aware Task scheduling in Ad-Hoc Desktop Grids". In: *Procedia Computer Science. 2nd International Symposium on Big Data and Cloud Computing (ISBCC'15). Elsevier*. Vol. 50. 2015, pp. 653–662.

[Ven16] Neelanarayanan Venkataraman. "A Survey on Desktop Grid Systems-Research Gap". In: *Proceedings of the 3rd International Symposium on Big Data and Cloud Computing Challenges (ISBCC'16)*. Ed. by V. Vijayakumar and V. Neelanarayanan. Vol. 49. Smart Innovation, Systems and Technologies. Springer, Feb. 2016, pp. 183–212.

[Vri+90] B.de Vries and J.C.Principe. "A theory for neural networks with time delays". In: *Conference on Advances in neural information processing systems (NIPS)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc, 1990, pp. 162–168.

[Wal+00] M. Waldman, A. D. Rubin, and L. F. Cranor. "Publius: A robust, tamper-evident, censorship-resistant web publishing system". In: *Proceedings of the 9th USENIX Security Symposium*. Vol. 9. San Diego, California USA, Aug. 2000.

[Wal+92] C.A. Waldspurger, T. Hogg, B.A. Huberman and J.O. Kephart, and W.S. Stornetta. "Spawn: A distributed computational economy". In: *IEEE Transactions on Software Engineering* 18.2 (Feb. 1992), pp. 103–117.

[Wol+01] R. Wolski, J.S. Plank, T. Bryan, and J. Brevik. "G-commerce: market formulations controlling resource allocation on the computational grid". In: *Proceedings of the 15th International Parallel and Distributed Processing Symposium. IPDPS'01. IEEE Computer Society.* Apr. 2001, pp. 747–772.

[Wol+03] R. Wolski, J. Brevik, J. S. Plank, and T. Bryan. "Grid resource allocation and control using computational economies". In: *Grid Computing: Making The Global Infrastructure a Reality.* Ed. by F. Berman, G. Fox, and A. Hey. John Wiley & Sons, 2003. Chap. 32.

[Xia+05a] L. Xiao, Y. Zhu, L. Ni, and Z. Xu. "GridIS: An Incentive-Based Grid Scheduling". In: *In Proceedings of 19th IEEE International Parallel and Distributed Processing Symposium.* Apr. 2005, 65b–65b.

[Xia+05b] L. Xiao, Z. Zhuang, and Y. Liu. "Dynamic Layer Management in Superpeer Architectures". In: *IEEE Transactions on Parallel and Distributed Systems* 16.11 (2005), pp. 1078–1091.

[Yan+03] B. Yang and H. Garcia.Molina. "Designing a super-peer network". In: CA, USA: IEEE, Mar. 2003, pp. 49–60.

[Yu+05] Y. Yu, S. Lee, and Z.L. Zhang. "Leopard: A locality aware peer-to-peer system with no hot spot". In: *NETWORKING 2005. Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications Systems. Proceeding of the 4th International IFIP-TC6 Networking Conference, Waterloo, Canada.* Ed. by R. Boutaba, K. Almeroth, R. Puigjaner, S. Shen, and J.P. Black. Vol. 3462. Lecture Notes in Computer Science. Springer, May 2005, pp. 27–39.

[Zha+01] Y. Zhang and N. Duffield. "On the constancy of internet path properties". In: *In Proceeding of the first ACM SIGCOMM Workshop on Internet Measurement IMW'01.* 2001, pp. 197–211.

[Zha+04] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz. "Tapestry: A resilient global-scale overlay for service deployment". In: *IEEE Journal on Selected Areas in Communications* 22.1 (Jan. 2004), pp. 41–53.

[Zha+05] C. Zhang, A. Krishnamurthy, and R. Y. Wang. "Brushwood: Distributed trees in peer-to-peer systems". In: *Proceeding of the 4th International Workshop on Peer-To-Peer Systems (IPTPS'05).* Springer-verlag, Feb. 2005, pp. 47–57.

[Zha+11] H. Zhao, X. Liu, and X. Li. "A taxonomy of P2P desktop grid paradigms". In: *Cluster Computing* 14 (2011), pp. 129–144.

[Zha+15] Y. Zhang, H. Huang, H. He, J. Teng, and Z. Wang. "Efficient distributed semantic based data and service unified discovery with one-dimensional semantic space". In: *Journal of Network and Computer Applications* 49 (2015), pp. 78–87.

[Zha+98] G. Zhang, B.E. Patuwo, and M.Y. Hu. “Forecasting with artificial neural networks: The state of the art”. In: *International Journal of Forecasting* 14.1 (1998), pp. 35–62.

[Zho+03] L. Zhong, D. Wen, Z.W. Ming, and Z. Peng. *Paradropper: a general-purpose global computing environment built on peer-to-peer overlay network*. May 2003.

[Zhu+05] Z. Zhuang, Y. Liu, and L. Xiao. “Dynamic Layer Management in Super-peer Architectures”. In: *Journal IEEE Transactions on Parallel and Distributed Systems* 16.11 (2005), pp. 1078–1091.

[Zol+09] S. Zols, Q. Hofstatter, Z. Despotovic, and W. Kellerer. “Achieving and maintaining Cost-Optimal Operation of a Hierarchical DHT System”. In: *Proceeding of the IEEE International Conference on communication (ICC)*. Germany: IEEE, June 2009, pp. 1–6.

[Zot+99] D. Zotkin and P.J. Keleher. “Job-Length Estimation and Performance in Backfilling Schedulers”. In: *Proceedings of the 8th IEEE International Symposium on High Performance Distributed Computing, HPDC ’99*. Perth, Malysia, Aug. 1999.

example.bib