Market-Oriented Resource Management and Scheduling: A Taxonomy and Survey

Saurabh Garg and Rajkumar Buyya

Cloud Computing and Distributed Systems (CLOUDS) Laboratory
Dept. of Computer Science and Software Engineering
The University of Melbourne, Australia
Email: {sgarg, raj}@csse.unimelb.edu.au

The shift of grids from providing compute power on sharing basis to commercial purposes, even though has not fully unfolded and still mostly limited to research, has led to various technical advancements paved a way to make utility grids a reality. Those advancements favor the application of market-based mechanisms for Grid systems by providing various pre-requisites on technical and economic sides. The creation of pervasive grid requires integration view of scalable system architecture, resource management and scheduling, and market models as shown in Figure 1.

This chapter summarizes the recent advances toward the vision of utility grids. First, it specifies all the requirements of a utility grid and presents an abstract model to conceptualize essential infrastructure needed to support this vision. Then, a taxonomy and survey of the current market-oriented and system-oriented schedulers is provided, examining the contribution and the outstanding issues of each system in terms of utility grid's requirements. This survey is intended to help researchers to make cooperative effort towards the goal of utility grids and provide insights for extending and reusing the existing grid middleware.

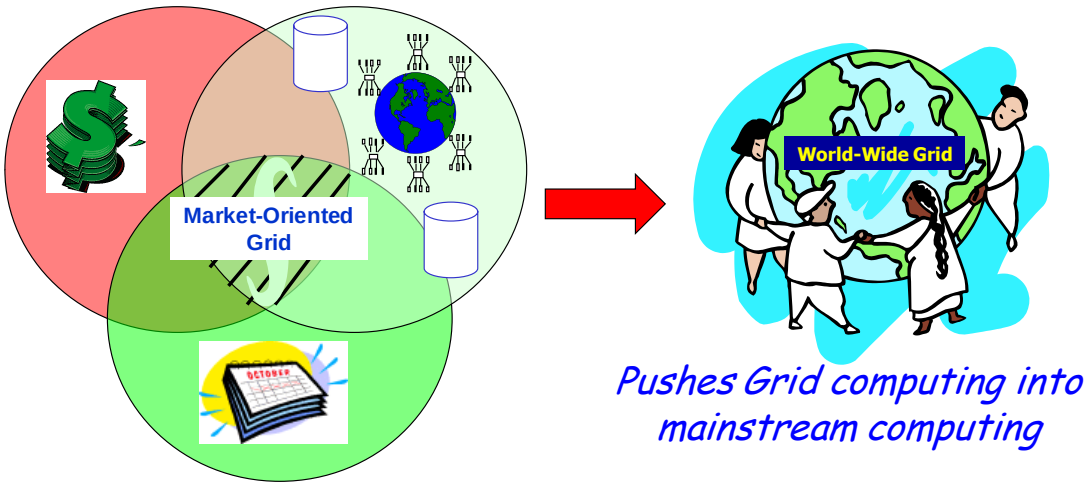


Figure 1: A view of market-oriented grid pushing grid into mainstream computing.

1.1 Overview of Utility Grids and Preliminaries

Utility grid imitates a market scenario consisting of the two key players i.e. Grid Service Consumers (GSCs) and Grid Service Providers (GSPs). Each of these players is generally self-interested and wanted to maximize their utility (see Figure 2). Consumers are users who have resource requirements to execute their applications. The resource requirement varies depending on the application model. For instance parallel applications demand multiple CPUs at the same time with equal configuration and network. The consumers are willing to compensate a provider for using its resources in the form of real money or barter. Providers, on the other hand, are the owner of resources (i.e. disk, CPU) which satisfy consumer needs. They can advertise their resources using other agents of the grid such as Grid Market Directories [68]. It is the responsibility of resource providers to ensure user's application gets executed according to service level agreement signed with consumer.

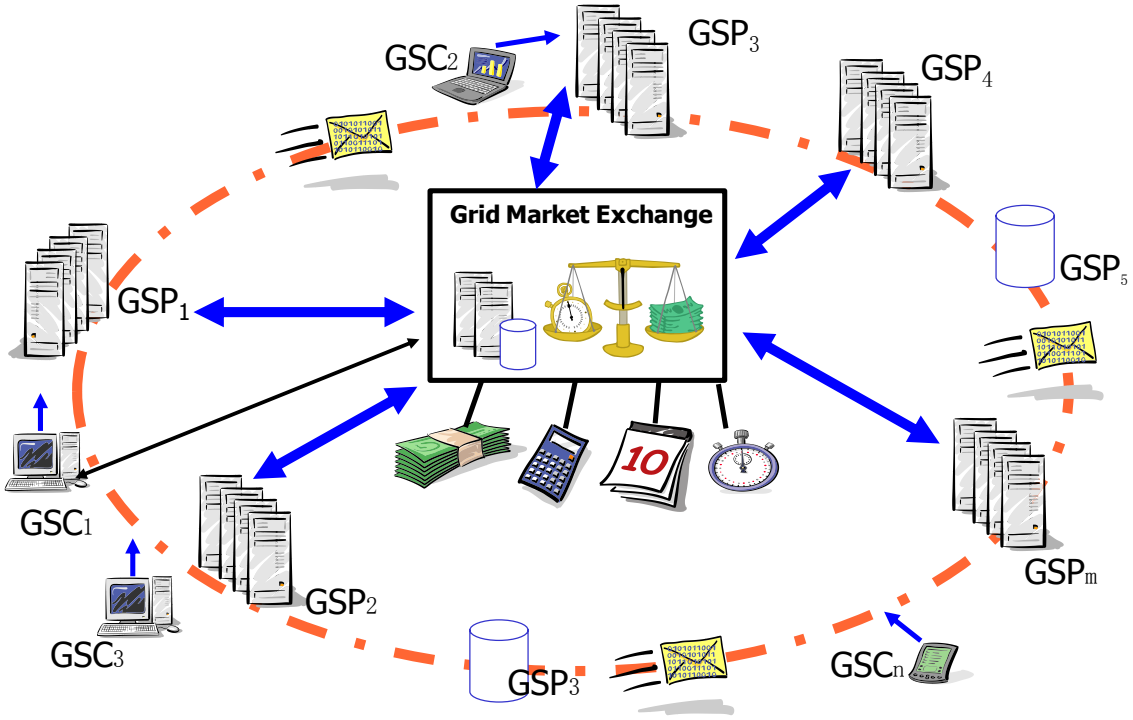


Figure 2: A grid market exchange managing self-interested entities (providers and consumers).

To ease and control the buying and selling process there are other players in the utility grid such as market place or exchange [43][46] which allows various consumers and providers to publish their requirements and goods (compute power or storage) respectively. This kind of market exchange can coordinate the users and lower down the delay in acquiring resources. Moreover, the market exchange can help in price control and reduces the chances of market being monopolized. The market exchange service provides a shared trading infrastructure designed to support different market-oriented systems. It provides transparent message routing among participants, authenticated messages and logging of messages for auditing. Brokers are another kind of middle agents on behalf of users which can do resource monitoring, resource discovery based on user Quality of Service (QoS) requirements, and job submission. The broker hides all

the complexity of grids from users. Similarly, there is also a need of legal support that can resolve various conflicts between providers and consumers, such as violation of Service level Agreement (SLA) [15]. Thus, the legal support can come from some authoritative agency such as country government. Each of the three main players i.e. consumer (or user agents such as broker), provider, market exchange has different requirements and goals. These requirements are discussed in detail and summarized in the next section.

1.2 Requirements

In this section, we discuss the main bottlenecks or infrastructural enhancements required for utility grids. In general, consumers and providers need mechanisms and tools that facilitate the description of their requirements and facilitate decision making to achieve their goals such as minimization of cost while meeting QoS requirements. For utility grid following requirements are essential:

1.2.1 Consumer Side Requirements

User-centric Brokers: These brokers are the user agents that discover and schedule jobs on to resources according to user's priorities and application QoS requirements such as budget, deadline, and number of CPU required [41][63]. These brokers hide heterogeneity and complexity of resources available in the grid. On behalf of users, the brokers provide functionalities such as application description, application submission and scheduling, resource discovery and matching, and job monitoring. The user broker can also do negotiation and bidding in an auction conducted by market exchange or providers for acquiring resources.

Bidding/Valuation Mechanism: In the utility grid a variety of market models can exist simultaneously such as commodity and auction market. To participate in both of the market model, users need to know the valuation of their application in the form of budget which estimates the user's requirements. For example in auction market, many users' bid to grab a resource, in such a requirement budget or valuation can help brokers to bid on behalf of users. In summary, consumers need a utility model to allow them to specify resource requirements and constraints.

Market-oriented Scheduling Mechanisms: In traditional grids, generally users want to schedule their applications on the resources which can provide the minimum response time and satisfy other QoS requirements in terms of memory and bandwidth. In the utility grids, one additional factor comes into picture i.e. cost which requires new mechanisms as user may relax its some of its other requirements to save on execution cost. Thus, one of the objectives of new scheduling mechanisms will be to execute the user application on the cheapest resource which can satisfy user's QoS requirements. These market-oriented mechanisms can vary depending on market model; user's objective (such as reduce time or cost) and application model (require multiple types of resources).

Allocation of Multiple Resources: Depending on the application model, consumer may want to run its application on multiple resources provided by more than one resource provider; for example, scheduling of a large parameter sweep across a number of providers, performing

distributed queries across multiple databases, or creating a distributed multi-site work flow. Thus, brokers should have capabilities to schedule applications and obtain resources from multiple resource sites.

Estimation of Resource Usage: In general, due to the heterogeneity of hardware and different input sizes, it is difficult to describe precisely execution time and requirement of an application which can vary drastically. In the traditional grid it is an important research problem of how to profile an application run time since it can affect not only the resource utilization but also cause delays for users. In the utility grid, this requirement becomes more critical as over estimation and under estimation of resource requirements can lead to tangible loss in the form of real money. Currently, the resource providers such as Amazon sell their compute resources in time blocks. In addition to that, if many users compete for the same resource, resource availability, depending on individual user's requirement, can vary from minutes to days. Thus, users must estimate their resource needs in advance. Thus, the profiling tools and mechanisms are required for efficient resource allocation in terms of utility.

1.2.2 Resource Provider Side Requirements

Resource Management Systems: These systems interact with underline hardware infrastructure and control the allocation of resources and job scheduling. In market-oriented system, the advance reservation function is required to identify and reserve resources in advance, and also to track the availability of resources that can be advertised by the provider. Thus, the reservation system should be able to provide the guaranteed resource usage time (based on SLA) and support the provider by estimating the future resource offers. Grid Middleware such as Globus has components such as advance reservation, but to support the market-oriented reservation and scheduling, they should be integrated with a module that supports various market-oriented scheduling mechanisms and models.

Pricing/Valuation Mechanism: In utility grids, resource provider's main objective is to maximize its profit not just the efficiency of the system, thus the mechanisms are required to set the resource price based on market supply and demand, and current level of resource utilization. These prices can be static or can vary dynamically based on resource demand. For example, academic user may require more resources and, thus willing to pay more due to conference deadline.

Admission Control and Negotiation Protocols: As stated before, in the market-oriented system, all participants are self-interested and want to maximize their utility. Thus, providers need to decide which user application they should accept or negotiate based on their profit. Since there may be chance of reservation cancellation by users, thus the mechanisms such as over provisioning of resources may be required by resource provider. SLA is also needed to be formulated once a user request is accepted for reservation. In addition, depending on how providers want to lease their resources, they may choose different market model for negotiation. For example, the simplest negotiation is required in Commodity model, while the bargaining model requires more intelligent negotiation protocol.

Commoditization of the Resources: Unlike many other markets, commoditization of the resources is one of the major difficult problems that complicate the reservation and allocation decisions. For instance, for a compute intensive application, it is meaningless to just allocate CPU without some memory. How much memory should be allocated when hardware infrastructure contain shared memory? Even for storage some small CPU cycle is required. Thus, the partitioning of resources by the provider is to be done that captures the hardware difficulties.

1.2.3 Market Exchange Requirements

An Information and Market Directory is required for advertising participants, available resources, auctions. It should support heterogeneous resources, as well as provide support for different resource specifications. This means that the market ought to offer functionalities for providing, for instance, both storage and computation with different qualities and sizes.

Support for Different Market Models: Multiple market models are needed to be designed and deployed as the resource providers and the consumers have different goals, objectives, strategies, and requirements that vary with time [43]. If there are multiple sellers for the same good, a double auction which aggregates supply and demand generally yields higher efficiency. If there is only one seller (e.g. in a differentiated service market for complex services), supporting single-sided auction protocols may be desirable. The negotiation protocol also depends on the user application. For example, in the case applications with soft deadlines, the large scheduling cycle helps in collecting more number of bids and offers for auction. This may lead to more efficient allocation than clearing continuously, since the allocation can be based on more resource information and has more degrees of freedom in optimizing efficiency (and/or other objectives). On the contrary, the users having urgent resource requirement may prefer an immediate allocation, thus commodity model will be better choice for negotiation. Consequently, market exchange must clearly support multiple negotiation protocols.

Reputation and Monitoring System: In general, it is assumed that after the scheduling mechanism has determined the allocation and resultant pricing, the market participants adhere to the market's decisions and promises. In reality, however, this does not happen due to several reasons such as untruthful behaviour of participants, failure while communicating the decision, and failure of resources. Consequently, there is a need for reputation mechanisms that prevent such circumstances by removing distrustful participants. However, there is strong need of monitoring systems that can detect any SLA violation during the execution. In the grids, the reason for a job failure or a corruption of results is hard to detect, since it can occur due to several reasons such as intentional misbehaviour of the resource provider, technical reasons which are neither controlled by the user nor the provider, and programming errors of the user. The monitoring systems should support reputation system for early detection of violations and responsible participant. An important challenge is thus to design such intelligent monitoring systems.

Banking system (Accounting, Billing, Payment mechanism): In the market exchanges, accounting system is necessary to record all the transaction between the resource providers and consumers. The accounting system especially records the resource usage and charges the consumer as per the usage agreement between consumer and provider. Meta-scheduling/Meta-

Brokering: The market exchange provides the services such as meta-scheduling of consumer applications on multiple resource providers in the case several consumers requires simultaneous access to resources. For instance, a large parameter sweep require resources across the number of providers, performing distributed queries across multiple databases, or creating a distributed multi-site work flow. Thus, meta-scheduling service does two task for their clients i.e. resource discovery and efficiently scheduling applications according to client's objectives. It can act as an auctioneer in case client wants to hold an auction.

Currency Management: For ensuring the fair and efficient sharing of resources and successful market, a well-defined currency system is essential. The two kinds of currencies models are proposed i.e. virtual and real currency. Both of these currency models have advantages and disadvantages based on managerial requirements. The virtual currency is generally deployed [52] due to its low risk and low stakes in case of mismanagement or abuse. However, virtual currency requires careful initial and ongoing management and lack flexibility. For buying and selling resources in real commercial environment, the use of real currency is preferred due to several reasons. The most important reason is that the real currency formats (e.g. USD, Euro, etc.) are universally recognised and are easily transferable and exchanged, and are managed outside the scope of a grid market exchange, by linked free markets and respective government policy.

Security: To avoid spamming, there should be a security system for user registration. All the services of the exchange must be accessed by authorized users.

1.3 Utility Grid Infrastructural Components

Based on the above requirements, in this section we discuss various infrastructure required for fully functional utility grid. Figure 1 outlines an abstract model for utility grid that identifies essential components. This model can be used to explore how existing grid middleware such user-centric brokers, meta-schedulers and resource management systems can be leveraged and extended to incorporate market-oriented mechanisms to support utility grid in practice.

The utility grid consists of multi-layer middleware for each participant: users (grid application, user level middleware), grid exchange, and provider (core middleware and Grid fabric).

The architecture of each of the component should be generic enough to accommodate different negotiation models for resource trading. Except grid exchanges and highlighted components, most of the components are also present in traditional grids.

The lowest layer is the **grid fabric** that consists of distributed resources such as computers, networks, storage devices, and scientific instruments. These computational resources are leased by providers, thus the resource usage is need to be monitored periodically to inform above layers about free resources which can be rented out. The resource managers in this layer have the responsibility of scheduling applications.

The **core middleware** offers the interface for negotiation with grid exchange and user-level middleware. It offers services such as remote process management, co-allocation of resources, storage access, information registration and discovery, security, and aspects of QoS such as

resource reservation and trading. These services hide the heterogeneity at the fabric level. The support for accounting, market model and pricing mechanisms is vital for provider to enable him to participate in the utility grid. The pricing mechanism decides how requests are charged. The pricing of resource usage by consumers can depend on several variables such as submission time (peak/off-peak), pricing rates (fixed/changing) or availability of resources (supply/demand). Pricing serves in the utility grid and serves as an efficient and cost-effective medium for resource sharing. The accounting mechanism maintains the actual usage of resources by applications so that the final cost can be computed and charged to the consumers. The market model defines the negotiation protocol that can be used to serve different resource requests depending on their effect on provider's utility.

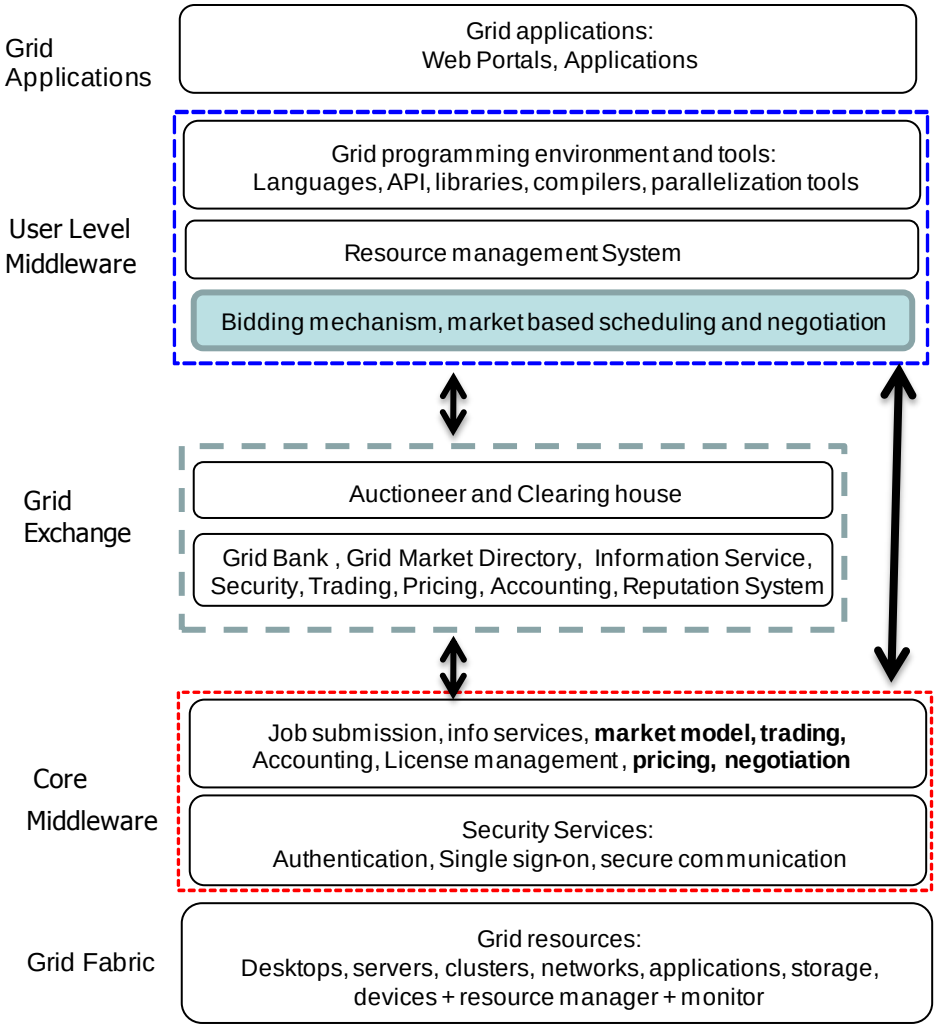


Figure 3. Utility Grid Components.

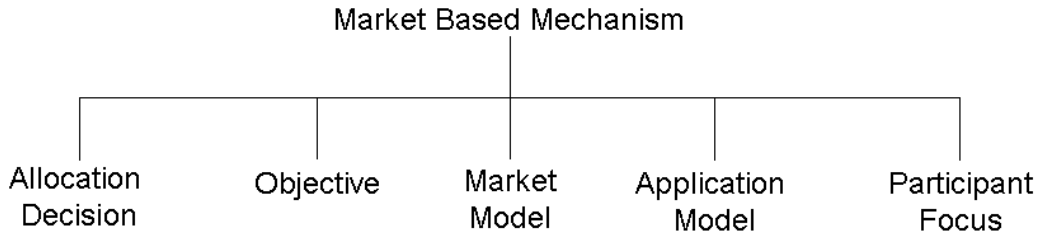
The **user-level grid middleware and applications** also need to be enhanced to satisfy the requirements discussed in the previous section. A new layer is needed to be build in the brokers to give users functionality of automatic bidding and negotiation. This layer also discovers resources based on user's requirements such as deadline and budget. Automated negotiation

capabilities are needed to be added to allow brokers to interact with grid exchange and provider's middleware, and form SLAs.

In traditional grids, the users and providers generally interact on one-to-one basis rather than using third party services. Similar to other markets, in utility grids, since negotiation with provider is more complex due to the involvement of money and flexible resource reservation, the third party services becomes essential. Thus, utility grids require **grid exchange** middleware which can act as buyer and seller on the behalf of users and resource providers respectively. They require having capability of auction and clearing house to match user resource demand to available resources. This middleware needs infrastructures such as meta-broker which does the matching of users and providers, Grid Market Directory (GMD) that allows resource advertisements, negotiation protocol, reputation system, security and price control. Meta-broker is the core of grid-exchange which act as an auctioneer or clearing house, and thus schedules user's application on the desired resources.

1.4 Taxonomy of Market-oriented Scheduling

There are several proposed taxonomies for scheduling in distributed and heterogeneous computing. However, none of these taxonomies focus on market-oriented scheduling mechanism in grids. Here, we present the taxonomy which emphasizes on the practical aspects of market-oriented scheduling in grids and its difficulties which are vital to achieve utility-based grid computing in practice. We can understand work in market-oriented scheduling from five major perspectives, namely market model, allocation decision, market objective, application model, and participants focus.



1.4.1 Market Model

The market model refers to the mechanism used for trading between consumers and providers. Any particular market model cannot solve all the special requirements of different participants. Having different characteristics each model is the most profitable to its participants depending on grid situation. For example, when the number of participants in terms of consumer and providers is almost same, the double auction is much better choice. Due to the differences in the applicability of auctions various authors has applied and analyzed the efficiency achieved [43]. Various market models that can be applied for market-oriented grid computing include the following:

Game Theory
If the grid participants only interact in the form of an allocation game with different payoffs as a result of specific actions employing various strategies, a game theoretical setting can be

assumed. This approach is generally used to ease the congestion of common resource or network which can lead to reduction in overall utility of the system. There are two types of solution approaches in this context:

1. To avoid excessive use of the resources one can use game with self-interested economic agents (non-cooperative games).
2. To achieve a load balancing effect the unselfish distribution of tasks over resources can be achieved by using cooperative game agents.

The cooperative and non-cooperative games for resource sharing and allocation often employ "Nash bargaining" approach, where bargainers negotiate for a fair "contract point" within feasible solution set [Nash 1950]. The use of games is not very common for resource allocation in market-oriented grid Computing. Feldman et al. [23] indicated in their analysis that the price-anticipating allocation scheme can result in higher efficiency and fairness at equilibrium. In their approach, resource price is estimated by total bids placed on a machine. Kwok et al. [37] proposed a Nash equilibrium-oriented allocation system with hierarchical organized game. They used reputation index instead of virtual budget or monetary unit for job acceptance decision.

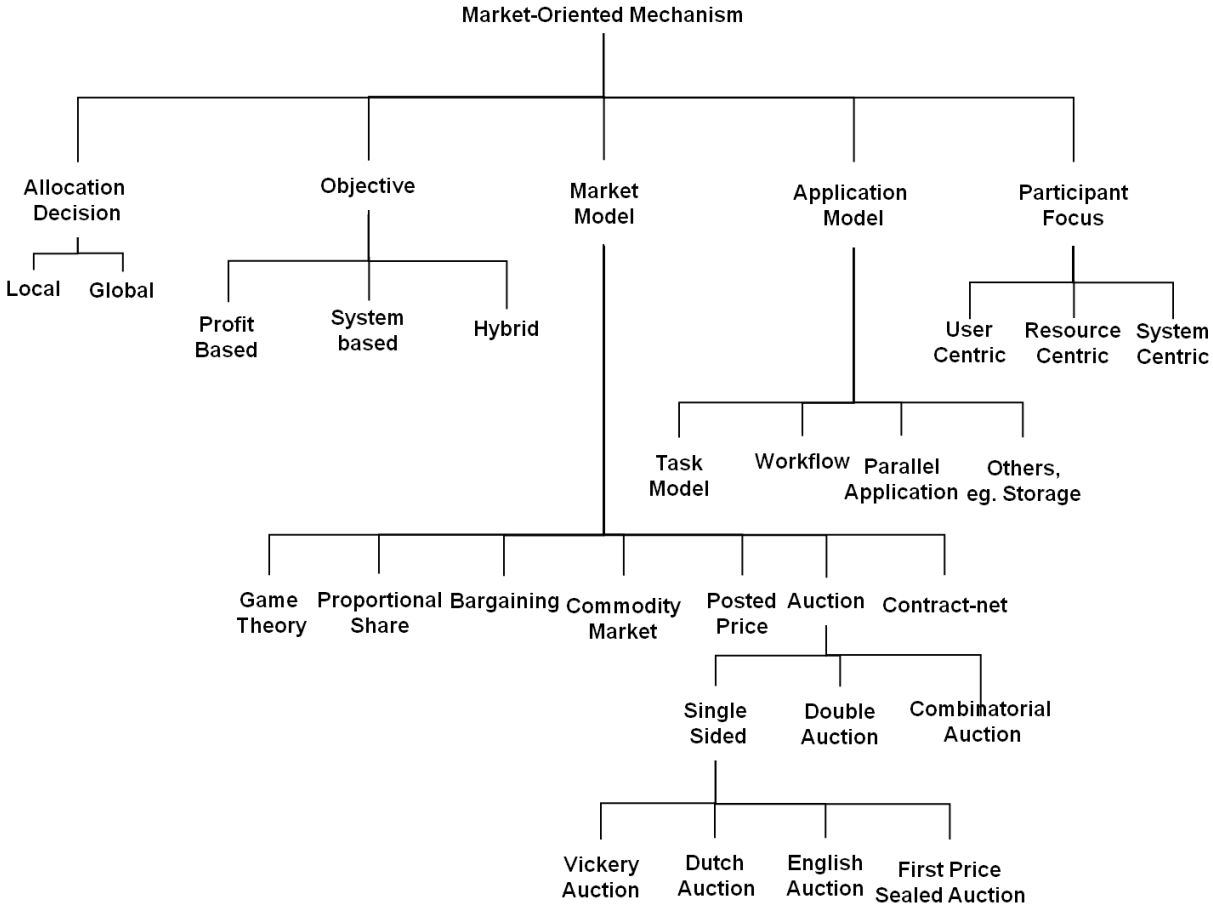


Figure 4. Taxonomy of Market Oriented Scheduling Mechanisms.

Proportional Share

The proportional share introduced and implemented in real cluster based systems such as Tycoon [38] to decrease the response time of jobs and to allocate them fairly. Neumann et al. [58] proposed a similar approach like proportional share where shares of resources is distributed using a discriminatory pay-as-bid mechanism to increase the efficiency of allocation and for the maximization of resource provider profit. This model makes an inherit assumption that resources are divisible which is generally not the case in when single CPU is needed to be allocated which is quite usual in cooperative problem-solving environments such as clusters (in single administrative domain).

In proportional share based market model, the percentage of resource share allocated to the user application is proportional to the bid value in comparison to other users' bids. The users are allocated credits or tokens, which they can use for having access to resources. The value of each credit depends on the resource demand and the value that other users place on the resource at the time of usage. One major drawback of proportional share is that users do not get any QoS guarantee.

Commodity Market Model

In this model, resource providers specify their resource price and charge users according to the amount of resource they consume. The resource allocation mechanisms consist of finding prices and allocations such that each economic participant maximizes their utility. One of the first evaluation works in grids on commodity market was presented by Wolski et al. [64] who analyzed and compared commodity market with other auction models. Many commercial providers such as Amazon [3] are using commodity market models with fixed and dynamic pricing.

The determination of equilibrium price is very crucial and a great tool in resource allocation decisions as participant want to maximize their utility. The prices depend on various factors such as investment and management cost of resource provider, current demand and supply and also future markets [53][2]. According to the prices, users can use various strategies to decrease their spending while getting satisfactory QoS level [49][56]. Various systems has been designed to automate this process; such as Dornemann et al. [17] designed and implemented of a workflow system based on business process execution language (BPEL) to support on-demand resource provisioning. Ernemann et al. [19] presented an infrastructure of the economic scheduling system for grid environments. HAJES (Kang et al. [33]) presented algorithm to increase revenue for providers who are under utilized by ensuring high availability to users. For the determination of the equilibrium pricing is done by many authors. Stuer et al. [59] presented pricing Commodity Resource Pricing in Dynamic Computational Grids. They proposed some refinements to the application of Smale's method for finding price equilibrium in such a grid market for price stability, allocative efficiency, and fairness.

Contract-net

In the contract net protocol, the user advertises its demand and invites resource owners to submit bids [55][67][32]. Resource owners check these advertisements with respect to their requirements. In case, the advertisement is favorable the resource owners respond with bids. The user consolidates all bids, compares them and selects the most favorable bids. The bidding process has only two outcomes: the bid is accepted or rejected in its entirety.

Bargaining

Bargaining models are employed in bi-lateral negotiations between providers and consumers and do not rely on 3rd parties to mediate the negotiation. During the negotiation, each player applies concessions until mutual agreement is reached by alternating offers [62]. Li and Yahyapour [39] proposed a concurrent bilateral negotiation model for grid resource management. The bargaining problem in Grid resource management is difficult because while attempting to optimize utility, negotiation agents need to: (i) negotiate for simultaneous access to multiple resources, (ii) consider the (market) dynamics of a computational grid, and (iii) be highly successful in acquiring resources to reduce delay overhead in waiting for resources.

Posted Price

It is similar to commodity market. In this model, providers may also make special offers such as discounts for new clients; differentiate prices across peak and off-peak hours. Prices do not vary relative to the current supply and demand but are fixed over a period of time.

Auction

An Auction is the process of trading resources by offering them up for bid and selling the items to the highest bidder. In economic terms it is also a method to determine the value of a resource whose price is unknown. A large amount of research studies bidding strategies and also mechanisms that are incentive compatible. An auction is a mechanism, organized by an auctioneer, to distribute grid resources from the providers to the users. The mechanism consists of determining the winner and setting the price. The auctions can be divided into three types based on participants and commodity exchanged: a) Single-sided auction, b) Double-sided auction, and c) Combinatorial auctions.

- **Single-sided Auction:** Single-sided auctions are mechanisms, where only buyers or sellers can submit bids or asks. Even though single-sided auction is the most widely applied market model, it often leads to inefficient allocation [51]. The most prominent single sided auctions are the Vickrey Auction, the Dutch Auction, First Price Sealed Bid (FPSB), and the English Auction.
    a. **English Auction:** In the English auction, the auctioneer begins the auction with a reserve price (lowest acceptable price) [16]. Auction continues in rounds with increasing bid prices, until there is no price increase. The item is then sold to the highest bidder.

b. **Dutch Auction:** In the Dutch auction the auctioneer begins with a high asking price which is lowered until some participant is willing to accept the auctioneer's price or a predetermined minimum price is reached. That participant pays the last announced price. This type of auction is convenient when it is important to auction resources quickly, since a sale never requires more than one bid.
c. **Vickrey Auction:** A Vickrey auction is a sealed-bid auction, where bidders submit sealed bids. The highest bidder wins, paying the price of the second highest bid. This gives bidders incentives to bid their true value. When multiple identical units are auctioned, one obvious generalization is to have all bidders pay the amount of the highest non-winning bid.
d. **First Price Sealed Bid (FPSB) Auction:** In this type of auction, all bidders simultaneously submit bids so that no bidder knows the bid of any other participant [16]. The highest bidder pays the price they submitted. In this case, the bid strategy is a function of one's private value and the prior belief of other bidders' valuations. The best strategy is bid less than its true valuation and it might still win the bid, but it all depends on what the others bid.

- **Double Sided Auction:** In Double auction, both providers and users submit bids which are then ranked highest to lowest to generate demand and supply profiles. From the profiles, the maximum quantity exchanged can be determined by matching selling offers (starting with the lowest price and moving up) with demand bids (starting with the highest price and moving down). This format allows users to make offers and providers to accept those offers at any particular moment. In double auction, the winner determination depends on different aspects such as aggregation, resource divisibility and if goods are homogeneous or are heterogeneous. Aggregation can come from the supplier side or from the buyer side. If no aggregation is allowed then each bid can be exactly matched to one ask. Divisible goods can be allocated partially. In the case that the bidder wants the entire good or nothing then its bid is considered indivisible. Kant et al. [34] proposed and compared various types of double auctions to investigate its efficiency for resource allocation in grid. Thus, Tan et al. [61] proposed stable continuous double auction to overcome high volatility.

- **Combinatorial Auctions:** The grid users may require a combination of multiple resources such as CPUs, memory, and bandwidth. Combinatorial auction allows users and providers to trade a bundle of multiple resources. It is advantageous to users as they do not need to participate in multiple negotiations with providers for each resource required. Moreover, in some cases it also leads to cost benefits. In combinatorial auction, users express their preferences as bundles of resources that need to be matched. The providers submit their asks and the auctioneer solves the optimization problem of allocation. Only drawback of combinatorial auction is the NP-hardness [42] of the matching problem which makes it inapplicable for large scale settings. Various variant of combinatorial auction are proposed in the literature to allocate computational resources among grid users [35][50].

1.4.2 Allocation Decision

In grids, the resource allocation to users can be done at two points. It can be done either by individual provider (local) or a middleman such as meta-scheduler or auctioneer (global). In local, the trading decisions are based on information of one resource provider. Generally in this case, users approach the resource provider directly to buy or bid for resource bundle advertised by resource provider. For instance, to buy compute resources of Amazon, users can directly negotiate with Amazon service. Most of the single sided auctions fall into this category.

In Global, the trading decisions are based on the global information of multiple providers. Users use the services of a meta-broker or auctioneer in the market exchange to get the required resources. Thus, the meta-broker or auctioneer makes the decision on behalf of the users to buy resources from providers. Double sided auction comes into this category. The local decision point is more scalable but can lead to contention. While the global decision point is more optimized and coordinate demand fairly.

1.4.3 Participant Focus

The two major parties in grid computing, namely, resource consumers who submit various applications, and resources providers who share their resources, usually have different motivations when they join the grid. The participant focus identifies the market side for whom market oriented systems or mechanisms explicitly designed to achieve benefit.

Application-Centric

In the application centric, mechanisms are designed such that application can be executed on the resources that meet user requirement within budget or minimum spending.

Resource-Centric

Similarly, a resource-centric mechanism focuses mainly on resource providers by fulfilling their desired utility goal in terms of resource utilization and profit.

System Centric

In the market scenario, there may be middlemen such as meta-brokers or meta-schedulers who act like an exchange, and coordinate and negotiate the resource allocations between consumers and producers. They try to maximize the utility for both users and provider.
Thus, the resource allocation decision involves multiple users and providers. In utility grid, mechanisms are required that can cater to the need of both the sides of market. For instance, they should be able to satisfy end-users' demand for resources while giving enough incentive to resource provider to join the market. Moreover, the specific requirements of participants should not be neglected. It is also possible for market-oriented resource management systems (RMS) to have multiple participants focus such as in Double auctions.

1.4.4 Application Type

Market-oriented resource allocation mechanisms need to take into account job attributes to ensure that different job types with distinct requirements can be fulfilled successfully. The application model affects not only the scheduling mechanism but also other aspect of utility grid such as resource offerings by providers, negotiation with providers, and formation of SLAs and their monitoring. For the applications consisting of independent task, application can be distributed across multiple providers and thus optimization of the user's utility is easier. For parallel application model, all task may be needed to be mapped on single resource site. In workflow type of application, there is a precedence orders existing in tasks, that is, a task cannot start until all its parent are done. This may require coordination between multiple providers and the problem is difficult since single failure may result in large utility loss.

1.4.5 Allocation Objective

The market-oriented mechanisms can be used to achieve different objectives both in the utility grids and the traditional grids. The allocation objective of mechanism can be profit oriented, system oriented or hybrid of both of them. The objectives of various participants decide the trading relationship between them. The profit based objective in terms of monitory gains, in general, encourages the competition between participants. Thus, each participant tries to maximize their own utility without considering other consumers. The objective of market-oriented scheduling mechanism could be to achieve optimization of system metrics such as utilization, fairness, load balancing and response time. This application of market-oriented mechanism, categorized in taxonomy as "system based objective", is quite common. For example, OurGrid [6] uses a resource exchange mechanism termed network of favours which is used to share resources among distributed users. Bellagio [8] is another system deployed on PlanetLab for increasing system utilization on non peak time. The objective of market-oriented scheduling mechanism can be of hybrid type. For example, user may simultaneously want to minimize the response time and the cost of application execution. A provider may accept less profitable application to increase its utilization rather than waiting for more profitable jobs.

1.5 Survey of Market Based System and Meta-schedulers

Table 1 shows a summary listing of existing market-oriented brokers, exchanges, and RMSs that have been proposed by researchers for various computing platforms. RMSs chosen for the survey can be classified into two broad categories: Market-oriented or system-oriented. Since this survey focuses on market-based Grid computing, market-oriented grid RMSs are surveyed to understand current technological advances and identify outstanding issues that are yet to be explored so that more practical market-oriented RMSs can be implemented in future. On the other hand, surveying system-oriented RMSs allow analyzing and examining the applicability and suitability of these systems for supporting market-based grid computing in practice. This in turn helps us to identify possible strengths of these systems that may be leveraged for market-based Grid computing environments. In traditional Grids, user accesses the grid services either through RMS or User Brokers (UB) or Local Resource Managers (LRM). These systems schedule the jobs using system-centric approaches optimizing the metrics such as response time, utilization etc. Thus we use "System-oriented Schedulers" to differentiate from schedulers in

market-oriented grids. In the market-oriented grid also there are three systems which participate in scheduling. One is user broker that provides access to users on multiple resources. On other side, the resource providers also have resource brokers which do admission control and pricing and also negotiate with user brokers. To facilitate the interaction, there can meta-brokers which match multiple users with multiple resources. The meta-brokers are generally part of market exchanges that provide other services such as resource discovery, banking, buying and selling compute resources.

1.5.1 Tycoon

Tycoon [38] is a market-based distributed resource allocation system based on Proportional Share scheduling algorithm. The user request with the highest bid is allocated the processor time slice. The bid is computed as the pricing rate that the user pays for the required processor time. Tycoon allocates the resources to the self-interested users in environments where service hosts are unreliable with changing availability. Tycoon distinguishes itself from other systems in that it separates the allocation mechanism (which provides incentives) from the agent strategy (which interprets preferences). This simplifies the system and allows specialization of agent strategies for different applications while providing incentives for applications to use resources efficiently and resource providers to provide valuable resources. A host self-manages its local selection of applications, thus maintaining decentralized resource management. Hosts are heterogeneous since they are installed in various administrative domains and owned by different owners. Tycoon's distributed markets allow the system to be fault tolerant and to allocate resources with low latency.

1.5.2 Spawn

Spawn [36] uses sealed-bid second-price auctions for market-oriented resource allocation in a network of heterogeneous computer nodes. Users place bids to purchase CPU resources for executing hierarchy-based concurrent programs in auctions held privately by each computer node and are not aware of other users' bids. The concurrent applications are then represented using a tree structure where a hierarchy of tasks expand or shrink in size depending on the resource cost. This mechanism limits the ability of customers to express fine-grained preferences for services.

1.5.3 Bellagio

The Bellagio [8] is a resource management system that allocates resources using Combinatorial auction in order maximise aggregate end-user utility. Users identify their resources of interest via a SWORD [45] based resource discovery mechanism and register their preference to centralized auctioneer for said resources over time and space as a combinatorial auction bids using a bidding language, which support XOR bids [44]. The bids are formulated in virtual currency. The auction employed in Bellagio is periodic. Unlike other work that focuses on the contention for a single resource (CPU cycles), they are motivated by scenarios where users express interest in 'slices' of heterogeneous goods (e.g. disk space, memory, bandwidth). Bellagio employs Share [13] for resource allocation in order to support a combinatorial auction for heterogeneous resources.

1.5.4 SHARP

SHARP [26] is not exactly a complete resource management system but it is an architecture to enable secure distributed resource management, resource control and sharing across sites and trust domains. The real management and enforcement of allocations are created by resource provider middleware which process the tickets and leases issued by SHARP. SHARP stands for Secure Highly Available Resource Peering and is based around timed claims that expire after a specified period, following a classical lease model. The resource claims are split into two phases. In the first phase, a user agent obtains a 'ticket', representing a soft claim that represents a probabilistic claim on a specific resource for a period of time. In the second phase, the ticket must be converted into a concrete reservation by contracting the resources site authority and requesting a 'lease'. These two phases allows SHARP system to oversubscribe by issuing more tickets than it can support. SHARP also presents a very strong security model to exchange claims between agents, either site agents, user agents or 3rd party brokers, that achieves identification, non-repudiation, encryption, and prevents man-in-the-middle and replay attack.

1.5.5 Shirako

Shirako [31] is a generic and extensible system that is motivated by SHARP for on-demand leasing of shared networked resources across clusters. Shirako framework consists of distributed brokers which provision the resources advertised by provider sites to the guest applications. Thus, it enables users to lease groups of resources from multiple providers over multiple physical sites through broker service. Site authorities compute and advertise the amount of free resource by issuing resource tickets to the selected brokers. When a broker approves a request, it issues a ticket that is redeemable for a lease at a site authority. The ticket specifies the type of resource, the number of resource units granted and the interval for which the ticket is valid. SHIRAKO allows 'flexible' resource allocation through leases which can be re-negotiated and extended via mutual agreement. A request can be defined as 'elastic' to specify a user will accept fewer resources if its full allocation is not available. Requests can be 'deferrable' if a user will accept a later start time than what is specified in the lease if that time is unavailable. The function of broker is to prioritize the request and match to appropriate resource type and quantity. Provider side is represented by site authorities that use Cluster on Demand [12] to configure the resources allocated at the remote sites.

1.5.6 OCEAN

OCEAN (Open Computation Exchange and Arbitration Network) [46] is a market based system for matching user applications with resources in the high performance computing environments, such as Cluster and Grid computing. It consists of all major components required to build utility grid, such as user node which submit trade proposals, computational resource and underlying market mechanism. Ocean first discovers potential sellers by announcing a buyer's trade proposal using optimized P2P search protocol. Then, the user node can negotiate with sellers based on the rules dynamically defined in a XML format. The ability to define negotiation rules is a remarkable characteristic of OCEAN that allows the adaptation of the economic model to diverse applications. The two possible negotiation allowed by OCEAN are "yes/no" and automated bargain.

1.5.7 CATNET

CATNET Project [21] proposed a Catallaxy based market place where trading is divided into two layers, the application and service layer. The notion of Catallaxy based market for grids was proposed by Austrian economist F.A. von Hayek. In this market, prices evolve from the actions of economically self-interested participants which try to maximise their own gain whilst having limited information available to them. In the application layer, complex services are mapped to basic services. The service layer maps service requests to actual resources provided by local resource managers. There are two market operate simultaneously- one for buying resources by service providers from resource providers and one for buying services by clients from service providers. Thus, the client is not aware of the details of the resource provider, and vice versa. The prices are fixed in two markets by bilateral bargaining. CATNETS offers very interesting features but lacks comprehensive support (e.g., monitoring, multi platform deployment).
In both layers, the participants have varying objectives which change dynamically and unpredictably over time. In the application/service layer, a complex service is a proxy who negotiates the access to bundles of basic service capabilities for execution on behalf of the application. Basic services provide an interface to access computational resources Agents representing the complex services, basic services and resources participate in a peer-to-peer trading network, on which requests are disseminated and when an appropriated provider is found, agents engage in a bilateral bargaining [22].

1.5.8 Nimord/G

Nimrod/G is a automated and specialized resource management system which allow execution of parameter sweep applications on Grid to scientists and other type of users. Nimrod/G follows mainly the commodity market model and provides four budget and deadline based algorithms [10] for computationally-intensive applications. Each resource provider is compensated for sharing their resources by the users. The users can vary their QoS requirement based on expense of execution and urgency. Nimrod/G consists of a Task Farming Engine (TFE) for managing an execution, a Scheduler that talks to various information services and decides on resource allocations, and a Dispatcher that creates Agents and sends them to remote nodes for execution. It is widely used by scientific community for their computation-intensive simulations in the areas of bioinformatics, operations research, network simulation, CAD, ecological modeling and Business Process Simulation.

Table 1. Market-oriented Scheduling Systems

NAME	Architecture	Economic Model	Mechanism	Traded Commodity	Pricing	Target Platform	Application Model	User Role
Tycoon (RMS**)**	Distributed, centralized	Proportional Share	Proportional Share	CPU Cycles	Pricing based on bids	Clusters, Grids	Task allocation	Discrete bid
Spawn (RMS**)**	Decentralized	Auction	Vickery Auction	CPU Time	Second price	Cluster	Task allocation	Discrete bid
Bellagio (RMS)	Centralized	Combinatorial Auction	Vickery Auction	CPUs and Storage	Second price	P2P	Not Specified	Bidding
Sharp (RMS)	Centralized	Commodity Market	Leases	CPU Time	Fixed	Grid	Lease allocation	Lease request to Sharp
Shirako (I)	Centralized	Not specified seems to be commodity	Negotiation, leasing generic	Virtual Machine and Storage	NA	Virtual Machines	Not specified	Lease request to broker
OCEAN (I)	Distributed	Bargaining, Tendering, Contract-Net, Continuous Double Auction	Discovering potential seller and Negotiation	CPU Cycles	Fixed	Any distributed resource	Not Specified	Discover and Negotiation .

CatNets (I)	Decentralized	Bargaining	Negotiation	Complex Services and Basic Services	Through negotiation . Dynamic pricing depend on available servers	Grid and Service-Oriented Computing	Not Specified	Bidding type
Nimrod-G (UB)	Centralized	Commodity Market, Spot Market, and Contract-Net for price establishment	Deadline and Budget Constrained Algorithms	CPU Cycles and Storage	Fixed pricing	World Wide Grid (resources Grid enabled using Globus middleware)	Independent multiple tasks & data parallel applications	Time and Budget
SORMA (I)	Centralized	Combinatorial Auction	Greedy Mechanism	NS	K-Pricing, based on auction	Commercial Providers	Not Specified (Simulation are based on independent tasks)	bidding
GridEcon (I)	Centralized	Commodity Market	NS	Resources managed by commercial service providers	Fixed	Commercial Resource Providers	Not Specified	Price specified by resource
Gridbus (UB)	NA	Commodity Market	Time and Budget based Algorithm	Compute and Storagee	NA	Commercial Providers	Bag of Task and workflow	Budget and time
Java Market (I)	Centralized	Commodity Market	Cost-based Greedy	Compute	Fixed	WWW	Java program	Bidding done by resources

Mariposa (RMS)	Centralized	Commodity Market	Cost minimization	Storage	Pricing based on load and historical info	Distributed database	Data	Budget and queries
GRIA (RMS)	P2P	Contract-based	NS	Compute	Through negotiation between providers.	Grid	NS	NA
PeerMart (RMS)	P2P	Auction	Double Auction	NS	Mean Price based on matched ask and bid	P2P	NS	Bids
G-Commerce (I)	Centralized	Commodity Market, Auction	NA	NA	Dynamic Pricing	Simulates hypothetical consumers and produces	NA	Bids

1.5.9 SORMA

Based on market engineering principles, the SORMA project [43] proposed Open Grid Market which is build above the existing resource management systems. It consists of self-interested resource brokers and user-agents. The users submit their bids for resources to Open Grid Market using an autonomous bidding agent. On the other side of market, the resource side bidding agents publish automatically available resources based on their predefined policies. The Open Grid Market matches requesting and offering bids and executes them against each other using Combinatorial Auction. The matches (i.e. allocations) are formulated in SLAs (i.e. contracts). The Grid middleware is responsible for the resource provisioning and the payment system (such as PayPal) for the monetary transfer of funds. The open infrastructure of Open Grid Market allows various resource providers with different virtualisation platforms or with different resource managers to easily plug in the market.

1.5.10 GridEcon

GridEcon Project [3] proposed a market exchange technology that allows many (small and medium) providers to offer their resources for sale. To support buying and selling of resources, GridEcon market offers various services that makes exchange of commodity convenient, secure, and safe. The GridEcon market also proposed to design a series of value-added services on top of the market exchange (e.g. insurance against resource failures, capacity planning, resource quality assurance, stable price offering), ensuring quality of the traded goods for Grid users. Currently, GridEcon support only commodity market model where commercial resource providers can advertised their spare resources. The fixed pricing is used to allow users to sell and buy resources. The GridEcon market acts as middleman between consumers and providers, while the real resource management is done by commercial service providers.

1.5.11 GridBus Broker

Gridbus Broker [63] is a single user resource broker that supports access to both computational and data grids. Gridbus can transparently interact with multiple type of computational resource which are exposed by various local-grid middleware's such as Globus, Alchemi, Unicore, Amazon EC2 and scheduling systems such as PBS and Condor. For scheduling, two general strategies are available which take into account budget and deadline of applications. Additionally, the design of the broker allows for writing a custom scheduler that implements a custom scheduling algorithm. Job-monitoring and status-reporting features are provided. Gridbus-Broker supports two types of application model i.e., parametric-sweep and workflow.

1.5.12 Java Market

Java Market [5] is one of the oldest market-oriented systems developed by John Hopkins Univ. It is an Internet-wide meta-computing system that brings together people who have worked to execute and people who have spare computing resources. One can sell CPU cycles by pointing Java-enabled browser to portal and allow execution of applets in a QoS-based computational market. The goal of Java Market is to make it possible to transfer jobs to any participating

machine. In addition, in Java Market, resource provider receives payments or awards which are function of execution time of job and amount of work done.

1.5.13 Mariposa

Mariposa [57] is a distributed database system developed at the University of California. It supports query processing and storage management based on budget. Users submit queries with time-dependent budget to brokers who then select servers for executing the queries based on two protocols. One protocol is expensive as it solicits bids from all servers, requiring many communication messages. The expensive protocol adopts a greedy algorithm that aims to minimize cost to schedule sub-queries so as to select the cheapest server for the user. The other protocol is cheap since it selects specific server based on historical information. In Mariposa, bids on queries are based on local and selfish optimization of each user.

1.5.14 GRIA

GRIA (Grid Resources for Industrial Applications) [60] is a web-services based grid middleware for business-to-business (B2B) service procurement and operation. It aims at the development of business models and processes that make it feasible and cost effective to offer and use computational services securely in an open grid market exchange. It also helps the Industries to achieve better utilization and manage demand peaks on resources. GRIA software is based on and uses web services standard specifications and tools such as Apache AXIS. GRIA aiming to make Grid Middleware reliable for industrial application, thus, provides various software packages for performance estimation and quality of service, workflow enforcement, cluster management, security and interoperability semantics. Thus, each service provider using GRIA middleware has an account service and a resource allocation service, as well as services to store and transfer data files and execute jobs to process these data files. Service provider's interaction is based on B2B model for accounting and QoS agreement.

1.5.15 PeerMart

PeerMart [28] is a Peer-to-Peer market based framework which allows completely decentralized trading of services between peers. It includes with capability of dynamic pricing and efficient price dissemination and services discovery over a P2P network. Using PeerMart, peers can bid prices for services, which enable them to govern the desired service performance. PeerMart implements an economically efficient distributed double auction mechanism where each peer being responsible for matching several services. PeerMart uses the overlay network infrastructure to map the services onto particular sets of peers following a fully distributed and redundant approach for high reliability and scalability to the number of participating peers. Its main limitation is the tightly integration of auctions model in the framework, making it inflexible with respect of the market model.

1.5.16 G-Commerce

G-Commerce [65] provides a framework for trading computer resources (CPU and hard disk) in commodity markets and Vickrey auctions. It is designed to compare resource allocation using

either commodity market or auction strategy based on four criteria: price stability, market equilibrium, consumer efficiency, and producer efficiency. While the Vickrey auction has the aforementioned shortcomings in grid, the commodity market typically works with standardized products. Additionally, the commodity market cannot account for the complementarities among the resources, as only one leg of the bundle is auctioned off, exposing the bidder to the threshold risk. G-commerce is a grid resource allocation system based on the commodity market model where providers decide the selling price after considering long-term profit and past performance. It is argued and shown in simulations that this model achieves better price predictability than auctions. However, the auctions used in the simulations are quite different from the ones we use in our work. The simulated auctions are winner-takes-it-all auctions and not proportional share, leading to reduced fairness. Furthermore, the auctions are only performed locally and separately on all hosts leading to poor efficiency across a set of host. In our work the best response algorithm ensures fair and efficient allocations across resources. An interesting concept in G-commerce is that users are allocated budgets that may expire, which could be useful for controlling periodic resource allocations and to avoid price inflation. The price-setting and allocation model differs from our work in that resources are divided into static slots that are sold with a price based on expected revenue. However, the preemption and agile reallocation properties inherit in the bid-based proportional share allocation mechanism employed in our system to ensure work conservation and prevent starvation are missing in the G-commerce model.

1.6 Other Grid Resource Management Systems

For over a decade various technologies have enabled applications to be deployed on the Grids, including Grid middleware such as Globus [24], Legion [11], and gLite[7]; schedulers such as Application Level Schedulers (AppLeS) [9]; and resource brokers including Gridbus Resource Broker[63], Nimrod/G[1], Condor-G [25], and GridWay[30]. These meta-schedulers or resource management systems interact with Local Schedulers or Grid Middlewares of various resource sites. The Local Scheduler supported such as Load Sharing Facility (LSF)[14], Open Portable Batch System (Open PBS [29] and Grid Engine (SGE / N1GE)[27]. In following section, we will discuss some of the Scheduling systems in detail and compare them using a Table 2.

1.6.1 Community Scheduler Framework (CSF)

The Community Scheduler Framework (CSF) [54] is an open source tool set for implementing a grid meta-scheduler, with the use of the Globus Toolkit Services, which provides an environment that can dispatch jobs to various resource managers. CSF was developed by Platform Computing in cooperation with the Jilin University, China. The CSF provide plug-in for various heterogeneous schedulers such as Platform Load Sharing Facility (LSF), Open Portable Batch System (Open PBS) and Grid Engine (SGE / N1GE), however CSF is designed for the best compliance with Platform LSF. CSF provide by default two basic scheduling algorithms i.e. Round-robin and reservation based algorithm. For later algorithm, CSF provides users advance reservation capabilities to user. Thus users can make reservation of resources using Resource Manager tool of CSF, in order to guarantee the resource availability at specified time on specified resource. It also provides the submission and monitoring tools for dispatching the job and queries its status information.

1.6.2 Computing Centre Software (CCS)

CCS [47] is vendor-independent resource management software that manages geographically distributed High Performance Computers. It is analogous to the well-known GLOBUS and consist of three main components the CCS, which is a vendor-independent LRMSs for local HPC systems; the Resource and Service Description (RSD), used by the CCS to specify and map hardware and software components of computing environments; and the Service Coordination Layer (SCL), which co-ordinates the use of resources across computing sites. CCS schedules and maps interactive and parallel jobs using an enhanced first-come-first-served (FCFS) scheduler with backfilling [9]. Deadline scheduling is another feature of CCS which gives the flexibility to improve the system utilization by scheduling batch jobs at the earliest convenient and at the latest possible time. It also supports jobs with reservation requirements. At the meta-computing level, the Centre Resource Manager (CRM) is a layer above the CCS islands that exposes CSS scheduling features. When a user submits an application, the CRM maps the user request to the static and dynamic information regarding available resources through Centre Information Server (CIS). Centre Information Server (CIS) is a passive component that contains information about resources and their statuses. Once the CRM finds resources, it interacts with selected CCS islands for resource allocations. If not all resources are available, the CRM either re-schedules the request or rejects it.

1.6.3 GridWay

GridWay [30] is a meta-scheduler framework developed by a team working for Distributed Architecture Group from Universidad Complutense in Madrid, Spain. GridWay provides a transparent job execution management and resource brokering to the end user in a 'submit and forget' fashion. GridWay uses the Globus GRAM to interface with remote resource and, thus it can support all remote platforms and resource managers (for example fork, PBS and LSF) compatible with Globus. GridWay offers only simple scheduling capabilities even though custom scheduling algorithms are also supported. By default, GridWay follows the "greedy approach", implemented by the round-robin algorithm. The collective scheduling of many jobs is not supported by meta scheduler. GridWay also provides sophisticated resource discovery, scheduling, constant monitoring and self-adaptive job migration to increase performance. Thus, an application is able to decide about resource selection as it operates, i.e. it can modify its requirements and request a migration. GridWay also enables the deployment of virtual machines in a Globus Grid.

1.6.4 Moab (Silver) Grid Scheduler

Moab Grid Schedule is a grid meta-scheduler developed by Cluster Resources Inc. Maob allows combining the resources from multiple high performance computing systems while providing a common user interface to all resources. It supports intelligent load balancing and advanced allocation allowing a job to be run over multiple machines in a homogeneous way or in a heterogeneous way resulting in better overall utilization and better time. Maob supports all major scheduling systems and even optionally rely on Globus Toolkit grid middleware for security and

user account management purposes. It manages the resources on any system where Moab Workload Manager (a part of Moab Cluster Suite) is installed. Moab Workload Manager is a policy engine that allows sites to control the allocation of available resources to jobs. The meta-scheduler supports fine-grained grid level fairness policies. Using these policies, the system manager may configure complex throttling rules, fairshare, a hierarchical prioritization and cooperation with allocation managers. Maob also has support for advanced reservations. This feature enables the use of scheduling techniques such as backfilling, deadline based scheduling, QoS support, and grid scheduling. One of the most interesting features going to be added in Maob is support for resource selection based on utility function where job completion time, resource cost and other parameters are taken into account. This feature allows easy transition of Maob meta-scheduler to market-based Grid.

1.6.5 Condor–G

Condor-G [25] is a fault tolerant job submission system that can access various computing resources which employs software from Globus and Condor [40] to allocate resources to users in multiple domains. Condor-G is not a real broker but a job manager, thus it does not support scheduling policies but it provides framework to implement scheduling architecture about it. Condor-G can cooperate with the following middleware: Globus Toolkit (2.4.x - 4.0.x), Unicore and NorduGrid, and it can submit jobs to Condor, PBS and Grid Engine (SGE / N1GE) scheduling systems. Condor's Classified Advertisement language (ClassAd) MatchMaking tool allows users to specify which resource to allocate. The mechanism allows both jobs and machines to describe attributes about themselves, their requirements and preferences, and matches result in a logical-to physical binding. The GlideIn mechanism is also provided in Condor-G that starts a daemon processes which can advertise resource availability which is used by Condor-G to match locally queued jobs to resources advertised. The command-line interface is provided to perform basic job management such as submitting a job, indicating executable input and output files and arguments, querying a job status or cancelling a job. Most striking capability of Condor-G is its failure management which can deal with crashes at various levels.

1.6.6 GRUBER/DI-GRUBER

To avoid bottleneck of a central broker, DI-Gruber [19] is implemented as a completely distributed resource broker. It has been developed as an extension of the SLA based GRUBER broker deployed on the Open Science Grid. The GRUBER system [18] consists of four main components. The engine implements several algorithms necessary for determining optimized resource assignments. The site monitor acts as a data provider that publishes the status of Grid resources. The site selector provides the information about sites which is used for selecting a resource provider for execution of new tasks. It communicates with the engine to select the resource provider. The queue manager resides on submitting hosts, deciding which jobs can be

executed at what time. The GRUBER can be utilized as the queue manager that controls the start time of jobs and enforces Virtual Organization (VO) policies, or as a site recommender when the queue manager is not available.

1.6.7 eNanos Resource Broker

eNANOS [48] is a general purpose OGSI-compliant resource broker developed by the Barcelona Supercomputing Center. It abstracts Grid resource use and provides an API-based model of Grid access. The eNanos Grid resource broker is implemented on top of Globus Toolkit (GT) and supports both GT2 and GT3. It focuses on resource discovery and management, failure handling, and dynamic policy management for job scheduling and resource selection. The eNanos Grid resource broker provides dynamic policy management and multi-criteria user requirements which are described in an XML document. These multi-criteria descriptions are used for resource filtering and ranking. The job scheduling in eNanos broker is divided into three phases, first is to select job to be schedule, second to select resource for selected job, and finally using meta-policy which consists of selection of the best job and the best resource. For job scheduling, several polices are implemented such as FIFOjobPolicy (First In First Out), REALTIMEjobPolicy (minimizes REALTIME=deadline time-estimated time of job finalization), EDFjobPolicy (Earlest Deadline First). Similarly for resource selection RANKresPolicy (resource selection based in the greatest rank obtained from the resource filtering process), ESTresPolicy (Earliest Starting Time, based in the estimated waiting time for a job in a local queue). Jobs are queued up in local system, and periodically scheduled by the resource broker.

1.6.8 AppLeS Parameter Sweep Template (APST)

APST [9] is an application level scheduler that provides an environment for scheduling and deploying large-scale parameter sweep applications (PSAs) on Grid platforms. APST supports scheduling and job submission on different Grid middleware and schedulers that take into account PSAs with data requirements. The APST scheduler allocates resources based on several parameters including predictions of resource performance, expected network bandwidths and historical data. The scheduler takes help of tools such as DataManger and ComputeManager to deploy and monitor data transfers and computation respectively which in turn get information from sources such as Network Weather Service (NWS) [66] and the Globus Monitoring and Discovery Service (MDS) [36]. AppLeS interacts directly with resource managers, perform all application management tasks, including, e.g., file staging, and can enact collations of applications. APST is compatible with different low-level Grid middleware through the use of Actuators and also allows for different scheduling algorithms to be implemented.

Table 2. System-oriented Scheduling Systems

Name	Allocation Mechanism	Scheduling Type	Scheduling Objective	Architecture	Application Type	Advance Reservation	Target Platform
CSF	Round Robin & reservation Based	Online	NA	Centralized	Task Model	Yes	LSF, Open PBS, SGE, Globus
CCS	FCFS	Online/Interactive	Minimizing Response Time	Decentralized	Parallel Application	Yes	NA
GridWay	Greedy/ Adaptive scheduling	Online	Minimize Response Time	Centralized	Parallel application, Parametric Sweep	No	Globus
Maob (Silver)	Fairshare, job prioritization	Online	Load balancing and response time minimization	Centralized	Task Model	Yes	Globus, Maui Scheduler
Condor/G	Matchmaking	Online	NA	Centralized	NS	No	Globus, Unicore, NorduGrid
Grubber/Di-Grubber	FCFS	Online	NS	Decentralized	Task Model	Yes	Globus
eNanos	Job Selection policies based on arrival time and deadline; Resource Selction based on EST	Batch/periodic	Minizing response time,	Centralized	Task Model	No	globus
APST	Divisible Load Scheduling-based algorithms	Single application	Optimize response time	Centralized	Parametric-Sweep	Yes/No	Globus

1.7 Discussion and Gap Analysis

After understanding the basic features of various market based systems and grid resource management systems, based on presented taxonomy and requirements of utility grids, we can identify several outstanding issues that are yet to be explored to adopt grid for creating a utility computing environment.

1.7.1 Scheduling Mechanisms

The market based scheduling mechanisms varies based on market model used for trading resources. For example, if auction is the main market model than strategies for bidding, and auction selection is required to maximize the chances of winning the auction. While in commodity model, aggregation of resources from different provider is required to maximize users utility. The challenges which are needed to be tackled more deeply can be categorized as following:

Support for Multiple QoS Parameters

In utility grid, other than traditional QoS requirements of users such as response time, additional Quality of Service (QoS) issues need to be addressed. For example, for HPC application, one to minimize the execution time, thus, resource capability and availability becomes essential which may be contradictory to the budget constraint. Many factors, such as deadline, resource reliability and security, need to be considered with monetary cost while making a scheduling decision on utility Grids. Similarly, resource management system should support QoS based scheduling and monitoring to deliver good quality service.

Support for Different Application Type

The market-based scheduling mechanisms proposed mainly support simpler applications/job models such as parametric sweep. But, in reality, more advanced job models that comprise parallel job processing type and multiple-task data intensive applications, such as message-passing and workflow applications are also required by users. Thus, advanced algorithms which require concurrent negotiation with multiple resource providers are needed to be designed.

Support for Market-oriented Meta-scheduling Mechanisms

Currently most economics based approaches to grid scheduling are studied using an auctions perspective [51]. However, auctions based scheduling may not always be suitable when users want immediate access to resources or they are part of same community. As an example of a user community, we consider the financial institution Morgan Stanley that has various branches

across the world. Each branch has computational needs and QoS (Quality of Service) constraints that can be satisfied by grid resources. In this scenario, it is more appealing for the company to schedule various applications in a coordinated manner. Furthermore, another goal is to minimize the cost of using resources to all users across the community. Thus, mechanisms are required for user selection and then resource allocation for utility maximization across all users.

1.7.2 Market Based Systems

In the previous sections, we discussed major systems which support market based mechanisms to buy and sell resources, and execute applications. Some of the most important outstanding issues from user, provider and market exchange perspective are presented as follows:

User Level Middleware

User level Middleware such as Gridbus broker [63] and GridWay [30] are designed only to participate in commodity market model. Moreover, they are not designed to trade in market exchange for leasing resources. Thus, these infrastructure supports is needed to provide flexibility to user to trade resources in any market. Moreover, automatic bidding support is required to participate in auctions used by systems such as Tycoon [38].

Market Exchange

As discussed previously, users and providers can also start negotiation using market exchange's services. The market exchange needed to match multiple users to multiple providers. The market exchange systems such as Catnet [21], Bellagio [8], GridEcon [3] and SORMA [43] have restrictive price setting and negotiation policies. In a real market exchange, the choice of negotiation and pricing protocols are decided by participants in the system. This flexibility is critical because the choice of negotiation protocol (auction, commodity market, and one-to-one) and pricing (fixed, variable) can affect the participants utility enormously depending on the current demand and supply. As number of consumers and providers grows scalability of the market exchange will be become an issue. Thus, some of the components such as job submission and monitoring which are already well supported by user brokers and meta-schedulers can be delegated to each user. It makes the system more decentralized in the sense that, market exchange mainly acts as the middleman for matching users demand to providers supply, and other responsibilities during job submission and execution will be delegated to user and provider's brokers (or resource management systems). A reputation system would also complement the market exchanges by removing the unreliable and malicious users from the market. In addition to that, market exchanges are needed to be flexible enough to provide the participants to use market protocol of their choice. It will require co-existence of multiple negotiations between consumers and providers.

Core Middleware (i.e., Resource Level Middleware)

Similar to user level middleware, the existing resource middleware needed to be extended to participate in market exchange. In addition to that, these systems support simple job models, and thus more advanced job models such as parallel applications and workflows needed to be considered. In addition to that, SLA monitoring is required to ensure that user's QoS satisfaction.

1.8 Summary

In this chapter, a taxonomy and survey of market based resource allocation approaches and systems are presented. This chapter also provides an overview of the key requirements and components that are needed to be added in the grid middleware to support utility grids. The taxonomy categorized the market based resource allocation approaches from five different angles: (i) allocation decisions (ii) mechanism's objective (iii) market model (iv) application model (v) participant focus. Then, the survey of various market based grid middleware is presented to examine current state-of-the-art in the utility grids, identifying gaps which are needed to be filled in. To mature utility grids, several projects are working to solve issues from both user and provider perspectives. We identified various areas of further work which can further enhance the capabilities of these systems. The rapid emergence of utility computing infrastructures such as Amazon's Elastic Cloud [69], combined with industrial and academic HPC demand has increased the development of open marketplaces. However, still significant work is required to get full benefit of utility grid, and make it main-stream paradigm that can serve the growing demand of computing infrastructures.

REFERENCES

[1] D. Abramson, R. Buyya, and J. Giddy. A computational economy for grid computing and its Implementation in the Nimrod-G Resource Broker. Future Generation Computing System, 18(8):1061–1074, 2002.

[2] D. Allenotor and R. Thulasiram. Grid resources pricing: A novel financial option based quality of service-profit quasi-static equilibrium model. In Proceedings of the 2008 9th IEEE/ACM International Conference on Grid Computing, 2008.

[3] J. Altmann, C. Courcoubetis, G. D. Stamoulis, M. Dramitinos, T. Rayna, M. Risch, and C. Bannick. GridEcon: A market place for computing resources. In Proceedings of 5th International Workshop on Grid Economics and Business Models, Spain, 2008.

[4] Amazon. Amazon. elastic compute cloud (ec2). http://www.amazon.com/ec2/2009.

[5] Y. Amir, B. Awerbuch, and R. Borgstrom. The Java market: Transforming the Internet into a metacomputer. Department of Computer Science, Johns Hopkins University, 1998.

[6] N. Andrade, W. Cirne, F. Brasileiro, and P. Roisenberg. OurGrid: An approach to easily assemble grids with equitable resource sharing. Lecture Notes in Computer Science, 2862:61–86, 2003.

[7] P. Andreetto, S. Andreozzi, G. Avellino, S. Beco, A. Cavallini, M. Cecchi, V. Ciaschini, A. Dorise, F. Giacomini, A. Gianelle, et al. The gLite workload management system. In Journal of Physics: Conference Series, volume 119, page 062007. Institute of Physics Publishing, 2008.

[8] A. AuYoung, B. Chun, A. Snoeren, and A. Vahdat. Resource allocation in federated distributed computing infrastructures. In Proceedings of the 1st Workshop on Operating System and Architectural Support for the On-demand IT Infrastructure, 2004.

[9] F. Berman and R. Wolski. The AppLeS Project: A Status Report. In Proceedings of the 8th NEC Research Symposium, Berlin, Germany, May 1997.

[10] R. Buyya, M. Murshed, D. Abramson, and S. Venugopal. Scheduling Parameter Sweep Applications on Global Grids: A Deadline and Budget Constrained Cost-Time Optimisation Algorithm. International Journal of Software: Practice and Experience (SPE), 35(5):491–512, 2005.

[11] S. Chapin, J. Karpovich, and A. Grimshaw. The Legion resource management system. In Proceedings of the 5th Workshop on Job Scheduling Strategies for Parallel Processing, San Juan, Puerto Rico, Apr. 1999. IEEE CS Press, Los Alamitos, CA, USA.

[12] J. Chase, D. Irwin, L. Grit, J. Moore, and S. Sprenkle. Dynamic virtual clusters in a grid site manager. In Proceedings of the Twelfth International Symposium on High Performance Distributed Computing (HPDC-12). Citeseer, 2003.

[13] B. Chun, C. Ng, J. Albrecht, D. Parkes, and A. Vahdat. Computational Resource Exchanges for Distributed Resource Allocation. 2004.

[14] P. Computing. LSF-Load Sharing Facility, 2003.

[15] K. Czajkowski, I. Foster, C. Kesselman, V. Sander, and S. Tuecke. SNAP: A protocol for negotiating service level agreements and coordinating resource management in distributed systems. Lecture Notes in Computer Science, 2537:153–183, 2002.

[16] E. David, R. Azoulay-Schwartz, and S. Kraus. Protocols and strategies for automated multi-attribute auctions. In Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1, pages 77–85. ACM New York, NY, USA, 2002.

[17] T. Dornemann, E. Juhnke, and B. Freisleben. On-Demand Resource Provisioning for BPEL Workflows Using Amazon's Elastic Compute Cloud. In Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid-Volume 00, pages 140–147. IEEE Computer Society, 2009.

[18] C. Dumitrescu and I. Foster. Gruber: a grid resource usage sla broker. Lecture Notes in Computer Science, 3648:465, 2005.

[19] C. Dumitrescu, I. Raicu, and I. Foster. Di-gruber: A distributed approach to grid resource brokering. In Proceedings of the 2005 ACM/IEEE conference on Supercomputing (SC'05), Seattle, WA, USA, November 2005. IEEE CS Press,Los Alamitos, CA, USA.

[20] C. Ernemann, V. Hamscher, and R. Yahyapour. Economic Scheduling in Grid Computing. In Proceeding of the 7th international workshop on Job scheduling strategies for parallel processing, Cambridge, MA, USA, June 16, 2001.

[21] T. Eymann, M. Reinicke, O. Ardaiz, P. Artigas, F. Freitag, and L. Navarro. Decentralized resource allocation in application layer networks. In Proceedings of the 3rd International Symposium on Cluster Computing and the Grid, pages 645–650, 2003.

[22] T. Eymann, M. Reinicke, F. Freitag, L. Navarro, O´ . Arda´iz, and P. Artigas. A Hayekian self-organization approach to service allocation in computing systems. Advanced Engineering Informatics, 19(3):223–233, 2005.

[23] M. Feldman, K. Lai, and L. Zhang. A price-anticipating resource allocation mechanism for distributed shared clusters. In Proceedings of the 6th ACM conference on Electronic commerce, page 136. ACM, 2005.

[24] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. International Journal of High Performance Computing Applications, 11(2):115, 1997.

[25] J. Frey, T. Tannenbaum, M. Livny, I. Foster, and S. Tuecke. Condor-G: A Computation Management Agent for Multi-Institutional Grids. Cluster Computing, 5(3):237–246, July 2002.

[26] Y. Fu, J. Chase, B. Chun, S. Schwab, and A. Vahdat. SHARP: An architecture for secure resource peering. In Proceedings of the nineteenth ACM symposium on Operating systems principles, pages 133–148. ACM New York, NY, USA, 2003.

[27] W. Gentzsch et al. Sun grid engine: Towards creating a compute power grid. In Proceedings of the 1st International Symposium on Cluster Computing and the Grid, page 35, 2001.

[28] D. Hausheer and B. Stiller. Peermart: the technology for a distributed auctionbased market for peer-to-peer services. In Proceedings of the IEEE International Conference on Communications, volume 3, pages 1583–1587. Citeseer, 2005.

[29] R. Henderson. Job scheduling under the portable batch system. page 279, 1995.

[30] E. Huedo, R. Montero, I. Llorente, D. Thain, M. Livny, R. van Nieuwpoort, J. Maassen, T. Kielmann, H. Bal, G. Kola, et al. The GridWay framework for adaptive scheduling and execution on grids. Software-Practice and Experience, 6(8), 2005.

[31] D. Irwin, J. Chase, L. Grit, A. Yumerefendi, D. Becker, and K. Yocum. Sharing networked resources with brokered leases. In Proceedings of the USENIX Technical Conference, pages 199–212, 2006.

[32] L. Kale, S. Kumar, M. Potnuru, J. DeSouza, and S. Bandhakavi. Faucets: efficient resource allocation on the computational grid. In Parallel Processing, ICPP 2004.

[33] W. Kang, H. Huang, and A. Grimshaw. A highly available job execution service in computational service market. In Proceedings of the 8th IEEE/ACM International Conference on Grid Computing, pages 275–282. IEEE Computer Society, 2007.

[34] U. Kant and D. Grosu. Double auction protocols for resource allocation in grids. In Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC05), pages 366–371, 2005.

[35] T. Kelly. Generalized knapsack solvers for multi-unit combinatorial auctions: Analysis and application to computational resource allocation. Technical Report HPL- 2004-21, HP Labs, Palo Alto, CA, USA, 2004.

[36] H. Keung, J. Dyson, S. Jarvis, and G. Nudd. Performance evaluation of a grid resource monitoring and discovery service. IEE Proceedings - Software, 150(4):243–251, 2003.

[37] Y. Kwok, S. Song, and K. Hwang. Selfish grid computing: Game-Theoretic modeling and NAS performance results. In Proceedings of the Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid'05), 2005.

[38] K. Lai, L. Rasmusson, E. Adar, L. Zhang, and B. Huberman. Tycoon: An implementation of a distributed, market-based resource allocation system. Multiagent and Grid Systems, 1(3):169–182, 2005.

[39] J. Li and R. Yahyapour. Learning-based negotiation strategies for grid scheduling. In Proceedings of the International Symposium on Cluster Computing and the Grid (CCGRID2006), pages 576–583. Citeseer, 2006.

[40] M. Litzkow, M. Livny, and M. W. Mutka. Condor - a hunter of idle workstations. In Proceedings of the 8th Int'l Conference of Distributed Computing Systems, San Jose, CA, June 1988.

[41] R. Montero, E. Huedo, and I. M. Llorente. Grid Scheduling Infrastructures with the GridWay Metascheduler, 2006.

[42] M. Narumanchi and J. Vidal. Algorithms for distributed winner determination in combinatorial auctions. Lecture Notes in Computer Science, 3937:43, 2006.

[43] D. Neumann, J. Stoesser, A. Anandasivam, and N. Borissov. Sorma-building an open grid market for grid resource allocation. Lecture Notes in Computer Science, 4685:194, 2007.

[44] N. Nisan. Bidding and allocation in combinatorial auctions. In EC '00: Proceedings of the 2nd ACM conference on Electronic commerce, pages 1–12, New York, NY, USA, 2000. ACM.

[45] D. Oppenheimer, J. Albrecht, D. Patterson, and A. Vahdat. Scalable wide-area resource discovery. In USENIX WORLDS, volume 4. Citeseer, 2005.

[46] P. Padala, C. Harrison, N. Pelfort, E. Jansen, M. Frank, and C. Chokkareddy. OCEAN: The open computation exchange and arbitration network, a market approach to meta computing. In Proceedings of the 2nd International Symposium on Parallel and Distributed Computing, pages 185–192, 2003.

[47] F. Ramme, T. Romke, and K. Kremer. A distributed computing center software for the efficient use of parallel computer systems. Lecture Notes in Computer Science, 797:129–136, 1994.

[48] I. Rodero, F. Guim, J. Corbalan, and J. Labarta. eNANOS: Coordinated Scheduling in Grid Environments. Parallel Computing (ParCo), pages 13–16, 2005.

[49] T. Sandholm, K. Lai, J. Ortiz, and J. Odeberg. Market-Based Resource Allocation using Price Prediction in a High Performance Computing Grid for Scientific Applications. In High Performance Distributed Computing, 2006 15th IEEE International Symposium on, pages 132–143, 2006.

[50] B. Schnizler. MACE: a multi-attribute combinatorial exchange. In Dagstuhl Seminar Proceedings, volume 6461. Springer, 2006.

[51] B. Schnizler. Resource allocation in the Grid. A market engineering approach, Ph.D.thesis. Studies on eOrganisation and Market Engineering, 2007.

[52] Shneidman, C. Ng, D. Parkes, A. AuYoung, A. Snoeren, A. Vahdat, and B. Chun. Why markets could (but dont currently) solve resource allocation problems in systems. In 10th USENIX Workshop on Hot Topics in Operating System, 2005.

[53] G. Singh, C. Kesselman, and E. Deelman. Adaptive pricing for resource reservations in shared environments. In Proceedings of the 8th IEEE/ACM International Conference on Grid Computing, pages 74–80. IEEE Computer Society, 2007.

[54] C. Smith. Open source metascheduling for virtual organizations with the community scheduler framework (CSF). Technical report, 2003.

[55] R. Smith. The Contract Net Protocol High-Level Communication and Control in a Distributed Porblem Solver, IEEE Transaction on Computer, 4:1104–1113, 1980.

[56] O. Sonmez and A. Gursoy. A novel economic-based scheduling heuristic for computational grids. International Journal of High Performance Computing Applications, 21(1):21, 2007.

[57] M. Stonebraker, R. Devine, M. Kornacker, W. Litwin, A. Pfeffer, A. Sah, and C. Staelin. An economic paradigm for query processing and data migration in Mariposa. In 3rd International Conference on Parallel and Distributed Information Systems, 1994.

[58] J. Stosser, P. Bodenbenner, S. See, and D. Neumann. A Discriminatory Pay-as- Bid Mechanism for Efficient Scheduling in the Sun N1 Grid Engine. In Hawaii International Conference on System Sciences, Proceedings of the 41st Annual, pages 382–382, 2008.

[59] G. Stuer, K. Vanmechelen, and J. Broeckhove. A commodity market algorithm for pricing substitutable Grid resources. Future Generation Computer Systems, 23(5):688–701, 2007.

[60] M. Surridge, S. Taylor, D. De Roure, and E. Zaluska. Experiences with GRIA: industrial applications on a Web Services Grid. In Proceedings of the First International Conference on e-Science and Grid Computing, pages 98–105. Citeseer, 2005.

[61] Z. Tan and J. Gurd. Market-based grid resource allocation using a stable continuous double auction. In Proceedings of the 8th IEEE/ACM International Conference on Grid Computing, pages 283–290. IEEE Computer Society, 2007.

[62] S. Venugopal, X. Chu, and R. Buyya. A negotiation mechanism for advance resource reservation using the alternate offers protocol. In Proceedings of the 16th International Workshop on Quality of Service (IWQoS 2008), pages 40–49. Citeseer, 2008.

[63] S. Venugopal, K. Nadiminti, H. Gibbins, and R. Buyya. Designing a resource broker for heterogeneous grids. Software-Practice and Experience, 38(8):793–826, 2008.

[64] R. Wolski, J. Plank, J. Brevik, and T. Bryan. Analyzing market-based resource allocation strategies for the computational grid. International Journal of High Performance Computing Applications, 15(3):258, 2001.

[65] R. Wolski, J. Plank, J. Brevik, and T. Bryan. G-commerce: Market formulations controlling resource allocation on the computational grid. In Proceeding of 15th International Parallel and Distributed Processing Symposium (IPDPS), pages 46–66. Citeseer, 2001.

[66] R. Wolski, N. Spring, and J. Hayes. The Network Weather Service: A Distributed Resource Performance Forecasting Service for Metacomputing. Journal of Future Generation Computing Systems, 15:757–768, 1999.

[67] L. Xiao, Y. Zhu, L. Ni, and Z. Xu. GridIS: An Incentive-Based Grid Scheduling. In Proceedings of 19th IEEE International Parallel and Distributed Processing Symposium, 2005, pages 65b–65b, 2005.

[68] J. Yu, S. Venugopal, and R. Buyya. A market-oriented grid directory service for publication and discovery of grid service providers and their services. The Journal of Supercomputing, 36(1):17–31, 2006.

[69] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic. Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility, Future Generation Computer System, 25(6): 599-616, 2009.