RJ10526 (ALM1501-003) January 15, 2015
Computer Science

IBM Research Report

A Multidimensional Marketplace for Cloud Services

Ray Strong, Susanne Glissmann-Hochstein
IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099
USA

A multidimensional marketplace for cloud services

Ray Strong
IBM Almaden Research Center
650 Harry Rd
San Jose, CA 95120, USA
hrstrong@us.ibm.com

Susanne Glissmann-Hochstein
IBM Almaden Research Center
650 Harry Rd
San Jose, CA 95120, USA
smglissm@us.ibm.com

***Abstract*— This paper proposes a marketplace (platform) supporting a multidimensional auction for cloud services. In this auction price varies with multiple dimensions associated with the service provided, some dimensions continuous and some discrete. Users of the marketplace are either requesters or providers of cloud services. In the course of their usage of the marketplace, users construct and observe a dynamic market for cloud services. To users the marketplace provides an easy to use, intuitive interface that includes the automatic construction of contracts and payment mechanisms. Behind the scenes the marketplace is driven by complex algorithms manipulating a price hypersurface for each offer or bid from a user. We offer a way to make these hidden computations feasible and also an open problem for computational geometry, where we seek an efficient method for managing such hypersurfaces for tens of thousands of users.**

Keywords: cloud services; platform-as-a-service, marketplace

I. INTRODUCTION

Each provider of cloud services has, at any one time, limited capacities of multiple types that must be managed in order to meet service level agreements. For simplicity consider only three of these multiple types: compute power, compute memory, and persistent storage. Moreover, the hardware providing these capacities may start out homogeneous, but if the service business is growing, new hardware, with different performance characteristics, will be ordered and installed from time to time as part of capacity management. Since demand is not very predictable, there will be times when capacities are so stretched that a provider must pay penalties for not meeting agreements, and there will be times when capacities are so underutilized that a profit margin drops below acceptable levels. Thus there will be times when a provider would reasonably charge higher rates for specific types of service and other times when the provider would reasonably charge lower rates for the same types of service.

The situation of requesters of cloud services is roughly symmetric. There are times when a requester would reasonably offer to pay a very high rate for a specific type of service and other times when a requester would reasonably offer to pay only a much lower rate. The timeliness of delivery of a new cloud service or upgraded (e.g. by higher contractual penalty for failure to meet a specific agreement) cloud service also offers a range of possibilities in terms of the market for the specific service in question.

In this paper we describe a marketplace that provides for dynamic and variable pricing of services, price varying continuously over multiple continuous dimensions. For example, rather than setting exactly one price for one guarantee of timeliness, one could offer a price that varied linearly with the deadline. One of the features of our proposed auction is that such continuously varying prices can be set dynamically by the user by choosing a few exemplary price points. Moreover, these price points could be chosen in response to the existing market rather than set in a manner independent of current demand.

A. Features of the proposed auction and marketplace

The primary goal of this research is to propose a multi-dimensional auction for cloud services with each bid and ask position corresponding to a hypersurface of prices over the dimensions. It is very important that users not have to think in terms of hypersurfaces, but rather in terms of desirable and undesirable points within the multidimensional space. The choices each user makes determine the hypersurface without the user having to provide a function from the multidimensional space to price. The choices multiple users make determine a market that is visualized in two dimensions for each user. The auction includes two phases: in the first phase each user displays and determines points of interest where potential matches exist (a match is always between a requester and a provider of cloud service); in the second phase either role may initiate a tender (a point binding offer), which if accepted is converted immediately into a contract binding on both parties, the specific point within the multidimensional space being converted into the details of the contract.

1) Symmetric views for cloud service provider and cloud service requester. The asymmetry in possible prices requiring prices to be positive notwithstanding, the view of matches presented to a service requester is a symmetric view to that presented to a service provider. Normally, the prices are rates per multi-dimensional unit (dimensions being temporal or relating to other forms of consumption of cloud service such as peak storage capacity, and work measured as a quantity of operations).

Phase 1 of the auction is dynamic in the sense that matches appear and disappear from a user view as other users demonstrate preferences that change their positions. A user can ask to see more or fewer matches, which will result in an altered view. Phase 1 allows users to construct a dynamic market for cloud services that is responsive to changes in both capacity and demand for service. This responsiveness, combined with the two phase nature of the auction, could allow an opportunity for a user to adopt a pretend role in order to give a false picture of competition, fraudulently moving the market in a preferred direction. In a later section we will discuss measures to prevent or ameliorate the effects of fraud on the market.

2) Continual. The pace of this two phase auction of cloud services will naturally be much slower than that of typical commodity auctions. The market will operate continually with no need for closings.

3) Transparent. In phase 1, all relevant positions are visible to any user (including competitive positions). The user controls how many matches are displayed, and only sees competition with respect to one match at a time; but in principle any user can view the closest point match corresponding to the position of any other user. The user sees only the closest match to the user's position. No user sees more than this indication of any other user's position.

4) Intuitive. Having set up bounds on the range of interest for each dimension, the user then establishes an initial flat price. User actions include moving the entire position up or down by a constant amount, requesting more or fewer matches, and indicating rejection of a match displayed.

5) Develops relevant view of current market for each user. The two phase auction allows users to explore and learn the current market, developing their own positions as they proceed. This means that the user has time to develop a position that fits both the dynamic market and the specific requirements or capacities associated with the user's role. Moreover, the position of a user represents a continuum of prices, corresponding to these specific requirements or capacities, removing the necessity of taking multiple point positions concurrently.

B. Related Work

There is a wide literature on multidimensional auctions ([Branco 1997], [Parsons 2011]) much of which is focused on fairness and prevention of unfair bidding tactics that "game the system" ([Bichler 2009], [Hudson 2001]). Our work focuses on computational feasibility and ease of use, with an estimate of the dimension set required for a cloud service market. Work on comparing cloud service providers in ([SPEC et al. 2012], [Liu et al. 2011]) provides clues to the kinds of dimensions required. The multidimensional auction we propose is based on that presented in [Cefkin 2013]. Here we emphasize dimensions relevant to a market of cloud services, and provide detail on both the computational feasibility and the computational geometry involved (See the work of David Mount, in particular [Arya 2011]).

C. Outline

Section 2 discusses the dimensions relevant to a market for cloud services and the user interactions with these dimensions before entering the multidimensional auction. Section 3 covers the behind the scenes operational support for our proposed multidimensional auction. Section 4 is a brief discussion of methods for gaming the system and how their impact can be minimized. Section 5 covers relevant computational geometry. Section 6 describes a grand challenge for computational geometers: how to make the support algorithms scalable to tens of thousands of users.

II. DIMENSIONS

The proposed marketplace offers a set of core cloud services dimensions of continuous and discrete nature to be set by the customer. In the case of a discrete dimension, such as the location dimension the customer may select one to many desired locations, or will leave it empty if the location is irrelevant for him. In the case of a continuous dimension the customer determines an acceptable range with a minimum and a maximum point.

Building on customer experience and needs, the dimensions defined in this paper can be further extended in future versions of the marketplace. For instance, dimensions may be added to the marketplace to provide more detailed requirements settings for the different cloud layers (see II.B). In a future PaaS-specific dimension, for instance, the customer may select the cloud programming language.

As the various cloud providers currently use different definitions of the dimensions differently, in the marketplace it is required that the providers agree on one standard to measure each dimension.

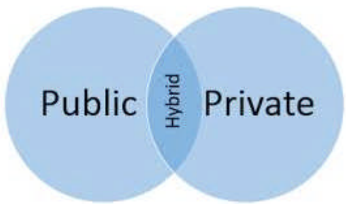
A. Service Layer

Cloud computing can be decomposed into the three layers Software-as-a-Service (SaaS), Infrastructure-as-a-Service (IaaS), and Platform-as-a-Service (PaaS) [Vaquero et al. 2009]. In the SaaS model the applications are hosted on the web. Thereby, the provider has the control over the infrastructure, including individual applications, network, servers, operating systems, and physical storage applications [SPEC et al. 2012]. The customer has only limited access to configure the application. The PaaS model represents the middle layer between SaaS and IaaS. The customer can deploy acquired software or create his own applications using the programming languages and tools made available on PaaS. Other services, such as authentication and data storage, may be offered to the developers [Voorsluys et al. 2011]. Within the platform, developers can write code, launch the application, test the applications for bugs, and edit the programs.

B. Deployment Model

The customer can choose between different types of cloud computing infrastructures ([SPEC et al 2012], [Liu et al. 2011]). In the private cloud the cloud resources are exclusively used by and configured for one organization or a small group of organizations. It can also be compared to the traditional outsourced data center. The public cloud is used by a large group of organizations. As the organization share the cloud resources the configuration settings are the same for every organization. The hybrid cloud combines private and public cloud aspects. It is a composition of distinct infrastructures in order to retain the proprietary technology by the consumer. The organization may use from the same provider some private cloud offerings as well as public offerings for different application areas. The community cloud is offered to an exclusive use for a community of organizations. The selection of the deployment model has a large impact on the costs and therefore the price for the consumer. Sharing resources among a large pool of consumers in general will reduce the provider's costs and therefore increase the price for the organization.

FIGURE I. DEPLOYMENT MODEL



C. Location

It is most cost-efficient for cloud providers to operate their cloud services at one location [Glissmann 2010]. However, due to customer requirements and data protective legislations companies may not be allowed to outsource their data to certain countries. For this reason, some providers operate worldwide data centers. Providers also operate more than one data center in order to decrease the transfer rate and to have a backup in case of outages [Chou 2004, pp.29-40]. In the marketplace a customer may select between a specific location (e.g., Belgium), an entire area (e.g., Europe) or not have any restrictions regarding the location.

D. Price Model

The price model differs largely among the Infrastructure-as-a-Service providers with an instances range from $0.02/hour to $6.82/hour (see Table II). In addition to the prices for instances, the providers also define a storage price. In some cases providers offer more detailed prices for the storage, differentiating between provisioned storage, snapshot storage, file storage or block storage pricing. Moreover additional charges may apply for application services and data egress. Also volume discounts may be given by the provider based on volume, commitment and prepayment. As a bonus the provider might offer an incentive for new users or a free trial period. The customer may be subjected to a penalty when changing contract conditions or canceling the contract.

In the case of Software-as-a-Service, the providers may charge a user per month subscription fee, or the number of transactions [Carraro/Chong 2006]. Additionally, providers may bill further costs, including support services [Pring et al. 2007]. For large numbers of users, a fix price might be offered for a user range (e.g., 500 – 1,000 users).

TABLE I. IAAS PRICE COMPARISON [SULLIVAN 2014]

Provider	Pricing	Bonus
Amazon Web Services	Instances range from $0.113/hour to $6.82/hour, with volume discounts available for reserved instances. Storage prices range from $0.095/GB/month to $0.125/GB/month. Additional charges for application services and data egress may apply.	New users can get 750 hours, 30GB storage and 15GB bandwidth for free with AWS's Free Usage Tier.
Windows Azure	Instances range from $0.02 to $1.60 per hour. Storage prices range from $0.07/GB/month to $0.12/GB/month, depending on level of redundancy.	Free 30-day trial with a limit of up to $200 is available for new users.
Google Compute Engine	Instances range from $0.019/hour to $1.659/hour. Provisioned storage is $0.04/GB/month; snapshot storage is $0.125/GB/month.	Google charges by the minute after a minimum of 10 minutes in an hour.
Rackspace Open Cloud	Instances start at $0.04/hour and go up to $5.44/hour. File storage starts at $0.10/GB/month and block storage is $0.12/GB/month.	Rackspace is currently offering a $100 credit on your first month bill.
IBM SmartCloud Enterprise	Hourly and monthly pricing available, however numbers are not disclosed. Contact IBM for details.	Free cloud server for one month from IBM's SoftLayer.
HP Enterprise Converged Infra-structure	Instances range from $0.03/hour to $3.40/hour. Block storage is $0.10/GB/month while object storage costs $0.09/GB/month.	Free 90-day trial with a $100 credit for each of your first three HP Public Cloud monthly invoices.

E. Penalties for the Provider

The Service Level Agreement may contain information about the penalties that are imposed if the service provider does not meet his service guarantees [Myerson 2013]. Similar to the previous dimensions, significant differences may exist among the providers. One partner may pay a penalty of 10 percent rebate of service fees for cloud service downtime exceeding one hour, while another partner may pay a penalty of 12 percent for downtime exceeding half an hour.

TABLE II. IAAS PROVIDER PENALTY COMPARISON [DIMENSIONDATA 2014]

Provider	Pricing
Terremark	pays USD 96 per 24-hour period of downtime, up to 50% of monthly usage fees.
Amazon Web Services	Amazon's SLA caps credits at 10% of the entire monthly bill.
HP Enterprise Converged Infrastructure	HP caps credits at 30% of the costs specifically associated with the resources that failed.
Rackspace Open Cloud	Rackspace Cloud pays up to 100% of the invoiced amount, but provides credits only against the costs of individual servers that failed

F. Response time

The response time is a critical measurement for cloud services [SPEC et al 2012]. It is the interval between when a client makes a request and when he receives the response. In the context of web workloads, for instance, the response time measures the time to return an entire page. Furthermore the response time can be defined for escalation or core support services. Rackspace guarantees a 2-hour live response time to any emergency ticket for escalation support and a 15-minute live response time to any emergency ticket for Core Support [Rackspace 2014]. Amazon Web Services provides different response times for severity level and client type [see Table III].

TABLE III. AMAZON WEB SERVICES' RESPONSE TIME FOR SUPPORT SERVICES [AWS 2014].

Severity Level	***Response Time***	***Available for***
Critical	15 minutes	Enterprise
Urgent	1 hour	Business Enterprise
High	4 hours	Business, Enterprise
Normal	12 hours	Developer, Business, Enterprise
Low	1 day	Developer, Business, Enterprise

G. Availability

Availability is defined as the percentage of time a system or a component is accessible [SPEC et al 2012]. A telephone system might, for example, have an availability of 99.9999%. Internet Service Providers often guarantee a higher availability for their paying email customers (99.9%) than for their non-paying customers (95 to 98%).

H. Elasticity (cl-slastandards-pdf)

Elasticity refers to internal provisioning, agility, scaling up and down [SPEC et al 2012]. Internal provisioning measures the time that is needed to increase or decrease rapidly and in some cases the resource demand. This can be a new instance (resource) in the IaaS model, a new instance of an application server in the PaaS model. Agility measures thereby how well the workload can be scaled and how well the system is provisioned close to the needed workload.

I. Throughput

Throughput in cloud services is measured in the same way throughput is measured in traditional systems [SPEC et al 2012]. This dimension represents the time it takes to receive an entire file, i.e., the units of work by the cloud per unit time. Further network throughput examples include the total number packets received per second, packets transmitted per second, bytes received per second, and bytes transmitted per second.

Standardization

Our marketplace requires standardization of dimensions in order to hold competitive auctions for cloud services. We first distinguish between discrete dimensions and continuous dimensions (See Table IV). The service layer, deployment model, and location are represented as discrete dimensions. Standardization for discrete dimensions is a matter of precise definition suitable for inclusion in a contract. Each point in the discrete dimension space has its own auction in the marketplace independent of the other points. Standardized continuous dimensions representing penalty, response time, availability, elasticity, throughput, and duration of contract, form the base dimensions of each auction.

TABLE IV. CORE DIMENSIONS.

Dimension	***Continuous / Discrete***	***Sample***
Cloud Characteristics		
- Service Layer	discrete	SaaS, PaaS, IaaS
- Deployment Model	discrete	private, public, hybrid, community
- Location	discrete	e.g., Belgium, Europe, not selected
- Price Model	continuous	instances range from $0.02/hour to $6.82/hour, additional pricing for storage
- Penalties	continuous	e.g., credits on 10-100% monthly price for 1 – 24 hour downtime
SLAs		
- Response Time	continuous	e.g., 15-minute live response time to any emergency ticket
- Availability	continuous	e.g., 99.999%
- Elasticity	continuous	e.g., several minutes for the acquired Virtual machine to be ready to use
- Throughput	continuous	e.g., 70 Mbit/s in a 100 Mbit/s Ethernet connection (70% efficiency)

A position in an auction is a partial function from such a space to price (which is regarded as a rate per standard unit of time). We propose to standardize these continuous dimensions by developing efficiently measurable benchmarks for each, with each dimension representing a composite score based on standard benchmark measurements. One of the functions of our marketplace will be to enforce and measure compliance with contractual provisions while acting as a conduit for payment. The particular set of measurements, the frequency, and the randomness will all depend on the specific point in the discrete dimension space. We expect our marketplace will select a small number of initial market points in the discrete space and add new market points when demand justifies their addition.

Our intention is to combine highly correlated continuous dimensions into single dimensions by means of simple combining functions like weighted averages. In most market points, one or two dimensions should suffice for each of availability, response time, and throughput (combining storage benchmarks with computational benchmarks when experience warrants). However, we may need several dimensions to describe the discounts and penalties relating to these dimensions and to the more complex dimensions we have described as elasticity. It appears that ten continuous dimensions may be sufficient but it would be extremely desirable to reduce this number to a much smaller number of dimensions that are both intuitively understandable to users and a sufficient basis for the construction of contracts. Thus we anticipate a uniform way in which penalties are assessed including penalties for terminating a contract. For example, we could have a single parameter k% that described the penalty for each failure to meet a service level agreement and also described the penalty for unilateral contract termination. For each duration when a condition of sla failure obtained, the penalty could be k% of the payment amortized over the duration. When a requester of service terminated a contract unilaterally, the penalty would be k% of the payment amortized over the remaining time on the contract. When a service provider terminated a contract unilaterally, the penalty would also be k% of the payment amortized over the remaining time on the contract.

If this kind of uniformity were too constraining, we could provide any number of extra dimensions for flexibility, but at the cost of added complexity for all users. Based on the business impact of the penalty agreements experienced, we could perform a non-linear adjustment to the scaling of a dimension in order to provide a more reasonable auction space. We anticipate using market experience to tune the scaling of various dimensions.

III. REPRESENTATION AND DISPLAY OF USER POSITIONS AND MATCHES

In its simplest form, the hypersurface representing a user position is the convex hull of a polytope possibly excluding a base face, the base being a surface at constant price (greater than or equal to other prices for a cloud service provider, less than or equal to other prices for a cloud service requester). The initial user position is simply the base. When the user requests at least one new match, the nearest matches are added to the user visualization at the same time the user's position is expanded to the convex hull of the polytope determined by the base and the nearest matches, the base being adjusted to the maximum price for a cloud service provider and the minimum price for a cloud service requester.

A. Commissions

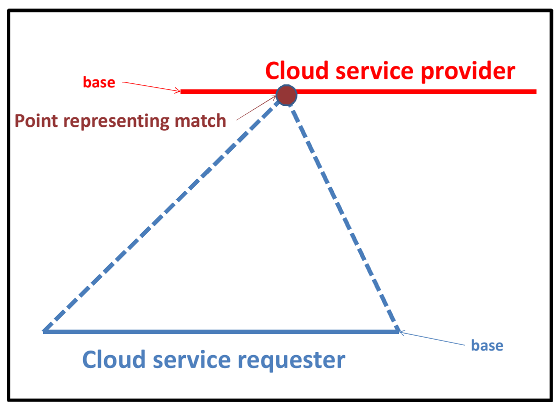
In order to allow the marketplace to charge a commission on the contracts generated, the price displayed for a requester may include a built-in commission. The commission would be included in the contract as a fee (rate) charged to the requester but paid to the marketplace rather than to the provider. In the rest of this paper, we will ignore commissions.

B. Maintain nearest (in price) matches

With many thousands of users arriving and departing from the marketplace, the most complex task of the marketplace is to maintain, for each user, the user's position and an ordered list of nearest matches. It is a computationally hard problem to determine whether two arbitrary medium (>3) dimension polytopes have non-empty intersection. However, our polytopes are far from arbitrary. Each of our polytopes has a flat base (flat, in the sense that the price is constant on the base). Moreover, every point in the polytope has a projection on the base, along the price dimension. To determine the minimum price separating two such polytopes, one with its base on top, the other with its base on the bottom, it is sufficient to determine the same quantity for the two polytopes resulting from restricting the original polytopes to have projection along the price dimension on the intersection of the bases.

In Figure II, we illustrate the management of two polytopes when one is trivial and the other requests an additional match. The dashed lines are not seen by the users. User cursor-over results in a display of the details of the match. Each user sees its own base price and the match points. A user does not see the base price for the other user involved in a match. Neither user sees the dashed line representation of hypersurface. In practice the base will have on the order of ten dimensions. The hypersurfaces we show in this paper represent projections onto one base dimension and the one price dimension. In these projections, the hypersurface of prices appears as a convex polygon with one missing side (the base). In the trivial case of one fixed price the hypersurface is identical to the base.

FIGURE II. REQUESTER HYPERSURFACE IS EXTENDED TO INCLUDE A MATCH WITH A TRIVIAL PROVIDER POLYTOPE CONSISTING OF ONE PRICE FOR THE BASE.



C. Maintain age of matches

Our suggestion for reducing the effect of fraud on the dynamic market of positions involves keeping track of how long each position has remained without either issuing or accepting a tender. Each user could independently decide when to ignore an old match, with a default age limit on the market visualization. Note that a match is irrelevant unless it is a vertex of the user's polytope.

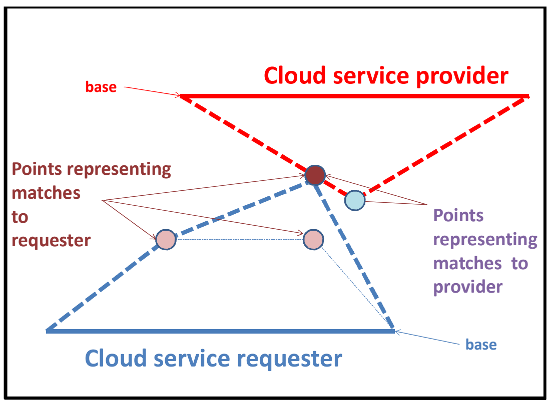
D. Raise (or lower) price hypersurface

When the user asks to raise or lower the user's position, the entire polytope is moved in the price dimension by the amount the user specifies, except that no part of the polytope may have a negative price.

E. Add new match

When the user asks to add a new match, all matches with minimum price difference are added to the visualization as new match points and possible new vertices of the polytope. If there are multiple minimum price difference points for two users, the center of gravity of the set of such points is used as the single point for the visualized match. The age of the position being matched contributes to the position of the display so that price tied matches are separated if there is an age difference.

FIGURE III. REQUESTER HYPERSURFACE EXPANDS FROM THIN DASHED LINES TO THICK DASHED LINES TO ACCOMMODATE A NEW MATCH WITH A PROVIDER.



If there is an exact tie between two matches so that the matching points are identical, then the points are perturbed randomly for display only. The visualization is maintained as a two dimensional projection of the space relevant to the user plus one extra dimension for the age of a match. The projection is perturbed until it separates all points that it is possible to separate. Then a further perturbation separates exact ties randomly. In Figure III, a new match is added and becomes visible to both requester and provider. The additional match is requested by the requester, so the hypersurface of the provider does not change. In this illustration there is a unique point (projection) with minimum price difference.

F. Remove a match

When the user asks to reduce the number of matches, all displayed matches with maximum price difference are removed. A user may also specify or change an age threshold for displayed matches. When a position has existed for a time over the threshold without contract related activity, the match is no longer displayed for the user, but no change is made to the polytope.

G. Delete a specific match

When the user asks to remove a specific match, the user's polytope is modified to remove the specified match and as few others as are required to maintain convexity. Ties are resolved by the user or randomly.

H. Display competition at a match

A user may request a display of competition at a specific match. For example, if the user is a requester, the user may request a display of the matches between other requesters and a specific provider. In this mode the user may increase or decrease the number of matches displayed as if the user were the provider. Such changes do not affect the display of the provider.

I. Change dimension scale for one user

It is sometimes possible to change the polytope by changing the scale of one or more dimensions until the specified match is a vertex and then to remove the specified match only. Polynomial scale changes may allow this type of removal. It is an open problem to determine by how much allowing unilateral polynomial scale change must increase the computational work required to maintain all the polytopes and all the matches.

J. Submit or accept a tender at a match.

Either requester or provider may submit a tender. A tender is a binding contract proposal. When a tender is submitted at a match, the contract is generated by the marketplace based on the details of the base point of the match. When a tender is accepted, the contract becomes binding on both parties.

A user may submit a special tender that specifies a requirement for acceptances by more than one user before the contract becomes binding.

TABLE V. SUMMARY OF THE USER INTERFACE.

Action	*Description*
change or set a dimension boundary	moves vertices of polytope to conform to the new base.
set an initial price	creates trivial polytope = base with fixed price
raise or lower the position by a constant amount	increase or decrease the price of every point in the polytope by a constant amount
increase or decrease the number of matches displayed	add nearest matches not displayed or delete farthest (from base) matches displayed
remove a specific match	remove a minimal number of matches required to remove the specific match and preserve convexity; remove oldest match of any set of alternatives
submit a tender at a match	offer a binding contract specified by the match
accept a tender at a match	accept a binding contract specified by the match
display competition at a match or resume displaying matches	change into or out of competition display mode where more or fewer matches to the position of a match are

	displayed
set age disqualification threshold	change the default age at which matches are not displayed

IV. GAMING THE MARKET

The market supported by the marketplace consists of the set of all pairs of polytopes that can potentially be displayed as matches. There are many ways that users could attempt to fraudulently distort this market. The only mechanisms we suggest as a defense against this distortion are

1) Identification of users to prevent one user from misrepresenting itself as multiple bidders; and
2) Removal of inactive matches from display after a user set age threshold.

The age threshold is applied to a position that has had no contract related activity for the threshold duration, contract related activity being submission or acceptance of tenders at any minimum price difference point of position corresponding to the specific match.

V. COMPUTATIONAL GEOMETRY

Any finite set of points with a projection via the price dimension to a maximum or minimum base defines a bounded convex polytope consisting of the convex closure of the set of points, including the base.

The set of vertices of a bounded convex polytope provides a convex basis for all points in the polytope, in the sense that any point in the polytope has a unique vector representation as a non-negatively weighted sum of the vector representations of the vertices in which the sum of the weights is 1. The vertices of our special polytopes consist of a small finite set of off base points to one side of the base in the price dimension together with the base vertices (represented by any vector in which the quantity for each dimension is either the maximum or the minimum for that dimension. All displayed matches are maintained as points of interest in both involved polytopes. Both convex sum representations are maintained for every such point of interest. Since every vertex has a price, and every point in a special polytope has a projection onto its base via the price dimension, each special polytope determines a function from its base to the price a maximum distance away from the base price. A match is represented in each involved polytope as a price and a point in the intersection of the bases that is the average of the two maximum distance points from the base prices.

When the bounds on a base dimension change, the convex sum representation of points remaining in the polytope must change because some of the base vertices change and some of the off base vertices may be removed. Our convention is that reduction in the size of the base does not affect the price ranges associated with what remains of the old base. A reduction in size of base by a change in one bound of one dimension creates a new bounding half space for the polytope. For each edge from an off base vertex in the removed half space to a vertex in the remaining half, a new off base vertex is created at the intersection between the edge and the new boundary of the polytope.

When a base is expanded to include new points, the off base vertices remain the same but some of the base vertices move, and the function from base point to price range changes via the new convex sum representations of points.

A change to the number of defining points is necessarily a change to the number of off base vertices. When an off base vertex is removed or added, a new convex sum representation is developed for the remaining polytope.

There is always a region (perhaps a single point) of minimum price distance between two of our polytopes that share a base because price distance is a continuous function from a closed convex set. The point in the base used to represent the match is the center of gravity of this region. When one of the bases has a change in bounds for some dimension, a new convex sum representation is developed for at least one of the polytopes. The resulting price distance function is then used to compute the new center of gravity for the new region of minimum price distance.

A. *Summary of required geometric operations*

In light of the above discussion, there are four required operations:

1) Given a set of base vertices and a set of off base points, develop a convex sum representation for all points of the polytope. This representation should be a (continuous) function from Euclidean vector representation to a vector of weights for the corresponding convex sum representation.
2) Given such a convex sum representation, develop a price function from the base to a price.
3) Given convex sum representations for two polytopes that share a base, develop a price difference function from a point in the base to a difference in the price dimension.
4) Given such a price difference function, compute the center of gravity of the region of minimum price difference in the base.

VI. OPEN PROBLEMS

These functions can all be performed by means of linear programming. Other methods have been explored for general convex polytopes (see [Arya 2011]). The basic open question is whether there are more efficient ways to perform these functions in our context.

A. *Computational efficiency at large scale*

The first open problem is to determine whether there is a computationally efficient way to perform these operations concurrently on large numbers (tens of thousands) of pairs of our special convex polytopes. Note that we must perform these operations on every pair of polytopes (one requester and one provider) with intersecting bases.

B. *Computational efficiency with unilateral rescaling of dimensions*

The second open problem is to determine whether there is a computationally efficient way to extend these operations to the case in which the objects involved are the images of our special convex polytopes under polynomial rescaling of dimensions. Note that such objects are not necessarily convex.

VII. CONCLUSION

This paper combines reasoning about how to represent the procurement of cloud services as a multidimensional auction with how to provide such an auction with a simple and intuitive interface in a way that allows a user to explore the market before making any binding offer. We present a framework and examples for standardization of the cloud service marketplace.

Answers to our open questions in computational geometry could determine how well the computational support for such auctions scales to large numbers of concurrent users. We could base computational support on linear programming. However, we offer the conjecture that supporting our auction is easier than arbitrary linear programming.

I. REFERENCES

[Arya 2011] Arya, S. da Fonseca, G.D., and Mount, D.M., Approximate polytope membership queries, STOC2011: 579-586, 2011.

[ASW 2014] AWS Support Services – Response Times, https://aws.amazon.com/premiumsupport/features/#Response_Times, Nov. 14, 2014.

[Bichler 2009] bichler, M., Pikovsky, A., Setzer, T., An analysis of design problems in combinatorial procurement auctions, http://dss.in.tum.de/files/bichler-research/2009_bichler_procurement_auctions.pdf , last accessed 10 Nov.,2014, 2009.

[Branco 1997] Branco, F. The design of multi-dimensional auctions, RAND J. of Economics 28:1, 63-81,1997.

[Carraro/Chong 2006] Carraro, G., Chong, F., Software as a Service (SaaS): An Enterprise Perspective,

http://msdn.microsoft.com/en-us/library/aa905332.aspx, last accessed on 2008/07/14.

[Cefkin 2013] Cefkin, M., Lehman, T., Rankin, Y., and Strong, H.R., INFORMS, October, 2013.

[Chou 2004] Chou, T., The End of Software - Transforming Your Business for the On Demand Future, Sams, 2004.

[DimensionData 2014] Comparing Public Cloud Service Level Agreements, Dimension Data, http://cloud.dimensiondata.com/am/en/about/resources/white-papers/comparing-public-cloud-service-level-agreements, last accessed on 2014/11/14.

[Glissmann 2010] IT Service-Oriented Investment Analysis - A Comparison of In-House versus Software-as-a-Service Deployment Solutions, Glissmann, S., University of St. Gallen, 2010.

[Hudson 2001] An analysis of uniform and discriminatory price auctions in restructured electricity markets, http://certs.lbl.gov/pdf/ornl-pricing.pdf , last accessed 10 Nov., 2014, 2001.

[Parsons 2011] Parsons, S., Rodriguez-Aguilar, J. A., Klein, M., Auctions and bidding: A guide for computer scientists, ACM Computing Surveys 43:2, January, 2011.

[Myerson 2013] Best practices to develop SLAs for cloud computing - Develop a standard way to create service level agreements that multiple partners can use, Myerson, J., IBM developer works, 2013.

[Pring et al. 2007] The Costs and Benefits of SaaS vs. On-Premise, Pring, B., Desisito, R. P., Bona, A., Report, Gartner Group, 2007

[Liu et al. 2011] NIST Cloud Computing Reference Architecture - Recommendations of the National Institute of Standards and Technology, Liu, F., Tong, J., Mao, J., Bohn, R., Messina, J., Badger, L., Leaf, D., US Department of Commerce, 2011.

[Rackspace 2014] Two levels of support: Escalation Support & Core Support, Rackspace, http://www.rackspace.com/cloud/private/support/, Nov. 14, 2014.

[SPEC et al 2012] SPEC Open Systems Group, Cloud Computing Working Group, "Report on Cloud Computing to the OSG Steering Committee", 2012.

[Vaquero et al. 2009] Vaquero, L.M., Rodero-Merino, L., Caceres, J., Lindner, M., A Break in the Clouds: Towards a Cloud Definition, Computer Communication Review, January 2009.

[Voorsluys et al. 2011] Voorsluys, W., Broberg, J. Buyya, R., Introduction to Cloud Computing, In: Cloud Computing: Principles and Paradigms, Editors: Buyya, R., Broberg, J., Goscinski, A., Wiley, March 2011.