

Energetic Resource examine setup for Dissolution Cloud Applications

Lokesh Pasupulati¹, G. Lakshmikanth²

P.G. Student, Department of Computer Engineering, Sree Rama Engineering College, Tirupati, India¹

Associate Professor, Department of Computer Engineering, Sree Rama Engineering College, Tirupati, India²

ABSTRACT: An Intelligent Economic approach for Dynamic resource Allocation (IEDA) is proposed with the improved combinatorial double auction protocol devised to enable various kinds of resources traded among multiple consumers and multiple providers at the same time enable task partitioning among multiple providers. To make bidding and asking reasonable in each round of the auction and determine eligible transaction relationship among providers and consumers, a price formation mechanism is proposed, which is consisted of a Back Propagation Neural Network (BPNN) based price prediction algorithm and a price matching algorithm. A reputation system is proposed and integrated to exclude dishonest participants from the cloud market. The Winner Determination Problem (WDP) is solved by the improved Paddy Field Algorithm (PFA). Simulation results have shown that IEDA can not only help maximize market surplus and surplus strength but also encourage participants to be honest.

In this paper we present a mechanism for service provisioning in distributed clouds where applications compete for resources. The mechanism operates by enabling execution zones to assign resources based on Vickrey auctions, and provides high-quality probabilistic models that applications can use to predict the outcomes of such auctions. This allows applications to use knowledge of the locality distribution of their clients to accurately select the number of bids to be sent to each execution zone and their value. The proposed mechanism is highly scalable, efficient and validated by extensive simulations.

KEYWORDS: Cloud Resource Management, Decentralized Cloud Applications, Vickrey Auctions, Quality of Experience.

I. INTRODUCTION

INTRODUCTION:

We present a resource allocation mechanism for service provisioning in distributed clouds where computation resources are made available much closer to users, possibly within routers themselves. This will save bandwidth, provide better user QoE and potentially comply with regulatory frameworks that demand services to run in specific regions/countries/states. Applications can then use these resources to deploy demanding services. Of course, when securing resources, applications will need to balance the benefit that these resources bring to the QoE of their users with the costs that they need to pay to the computation resource providers. To this end, we propose an auction-based resource allocation approach that allows cloud providers to provide computation capacity in a decentralized manner, while at the same time allowing applications to implement effective cost/benefit tradeoffs. We take as given the well-known properties of the Vickrey auction (eg. truthful revelation and ex-post efficiency [2]) and focus on their use as a vehicle to implement a self-adaptive mechanism for the dynamic allocation of service slots, obviating the need of a logically centralized clearinghouse.

CLOUD computing provides virtually unlimited computing power as utility service to consumers. It enables different provisioning models for on-demand access to applications (Software as a Service, or SaaS), platforms (Platform as a Service, or PaaS), and computing infra-structures (IaaS). It has created a competitive market where consumers pay providers for using resources and are usually billed using a pay-as-you-go model. To facilitate trading, a market mechanism should be explored to allocate and utilize resources within their capacities without over-provisioning or

under-provisioning[1]. In our proposed system we assume the existence of a large number of execution zones (EZ), where computational capabilities are offered to applications. Resources in each EZ are managed by a zone manager (ZM), which is operated by the infrastructure provider. Applications can deploy services (i.e. self-contained application logic components) on these EZs under the direction of an application manager (AM), which is operated by the application provider and implements the appropriate cost-benefit tradeoffs for that application. The AMs belonging to various applications then compete for resources on the EZs. Rather than relying on a centralized resource allocator to grant AMs access to EZ resources, we allow each EZ to individually perform resource allocation decisions. There is a rich literature on how to solve this problem [3]–[5]; we propose the use of a market system to allow EZs to allocate resources to those applications that value them the most (see [6]–[9]). We take advantage of the well known properties of Vickrey auctions [2] to allocate resources to applications (see [10]–[15] for other examples of auctions used for resource allocation in grid, peer-to-peer and cloud computing). Our mechanism allows each AM to bid for resources on the desired EZs according to its individual policies, and to implement its own tradeoff between user satisfaction and infrastructure cost. This involves considering the monetary cost of the resources together with the cost of users being blocked due to resource unavailability.

We consider latency- and bandwidth-sensitive services which require accurate placement of service instances so that the network performance metrics between users and cloud-based services are within acceptable bounds to ensure QoE. There are two dimensions to QoE in our model. Firstly there is the quality experienced by the users of an application in terms of the latency, bandwidth and response time of a session. This is determined by an AM selecting EZs that are topologically close to the users in terms of network metrics, such as latency, and by the EZ offering sufficient computational capabilities (e.g. CPU, memory) for the application.

II. AUCTION-BASED RESOURCE ALLOCATION FOR THE DECENTRALIZED CLOUD

We assume that the AM can determine, by external means, the total user demand associated with a given EZ. Then, for each AM the resource allocation problem becomes: given the bidding profile at every EZ, for how many service resource units should the AM bid, and at what price? Formally, each AM b_j must determine the number of service slots m_{ij} it will bid for at each EZ j , as well as the bid price v_{ij} it must offer, according to the estimated user demand and expected service quality. We now present our analytic model of the EZ, and how it implements resource allocation using Vickrey auctions. In a standard Vickrey second price auction the bidders bid for a single item. The highest bid wins and pays the price of the second highest bid. This can be shown (under certain assumptions) to mean that rational bidders will bid according to their true valuation for the item. In a multi-unit Vickrey auction with identical items, the bidders bid for K items and may each bid multiple times. The highest K bids win and they pay the combined value of the next K bids (the first K losing bids). This preserves the property that rational bidders bid according to how much they value the item.

A. RESOURCE ALLOCATION PROPERTIES

Even though the Vickrey auction has interesting advantages (e.g. incentive compatibility and ex-post efficiency), it is not revenue-maximizing [2] [17]. In fact, for uncontested Vickrey auctions, the bidders can obtain items at zero cost. For the present paper we assume that AMs pay a monthly fixed fee to gain access to the EZ infrastructure; this would allow them to use uncontested resources. If AMs wish to gain access to EZ resources for which there is contention, they will need to submit nonzero bids and pay for these bids in addition to the fixed flat fee. In contrast to other contributions in the literature, we assume that bids are made in terms of actual currency and the EZs receive income from both the monthly rental fees and non-zero winning bids from AMs.

B. ESTIMATING THE EXPECTED NUMBER OF SUCCESSFUL BIDS

To allow an AM to estimate the expected number of bids it will win and the total price it will pay when placing a given number of bids at a given price, each with a given ZM, it needs to estimate the level of competition it will face in the auction from other AMs. Formally, this is achieved by determining the number of competing bids it will encounter. Thus, rather than the total number of bids N_i received by n_i , an AM b_j is interested in the number of bids with which its bids must compete. This involves subtracting from N_i those bids sent to n_i by b_j itself. We shall denote this number of competing bids as N_{ji} , the total competing load of manager b_j at EZ n_i . By definition, $N_{ji} = N_i - m_{ji}$.

C. THE EXPECTED RESOURCE COST

We now calculate the expected cost of an AM when bidding at a given EZ.

$$\begin{aligned} E[C_{ij}] &= v_{ij}\mu_{ij} - \int_0^{v_{ij}} \sum_{k=1}^{X_{\max}^{ij}} F_{(\phi_{ij}+k)}^{V^i}(\omega) d\omega \\ &= v_{ij}\mu_{ij} - \int_0^{v_{ij}} \mu_{ij}(\omega) d\omega. \end{aligned}$$

D. THE EXPECTED QOS COST

In this section we consider the QoS cost that an AM expects. The QoS cost can be considered as a penalty to QoS that the application has, given the number of service slots secured.

$$\mu_{ij}^Q = Q_{ij}(0) + \sum_{k=1}^{X_{\max}^{ij}} \Delta Q_{ij}(k) \mathbb{P}[V_{(\phi_{ij}+k)}^i < v_{ij}].$$

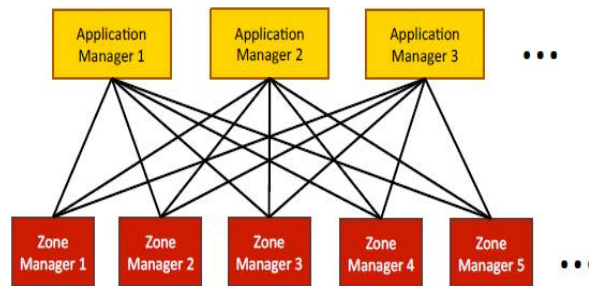
E. BID DETERMINATION: THE EXPECTED TOTAL COST

Suppose that an AM b_j wishes to determine the number and value of bids that it must submit to a given execution manager n_i , and it wishes to do so taking into account both the resource cost and client QoS. This can be achieved by selecting the combination of m_{ij} and v_{ij} that minimize a total cost measure that includes both QoS and monetary bidding effects.

$$Q_{ij}(k) = \frac{\frac{\rho_{ij}^k}{k!}}{\sum_{n=0}^k \frac{\rho_{ij}^n}{n!}} B_{ij}.$$

F. STABILITY

A common pitfall in decentralized optimization mechanisms, such as the one presented in this paper, is the possibility for oscillation (see eg. [23], [24]). We address this concern by using the optimal answers provided by (11) to guide an incremental bidding strategy, thus dampening oscillations and ensuring convergence. During each epoch, each AM calculates the optimal m_{ij} and v_{ij} based on the level of competition and resource availability in each EZ. Instead of using these directly, AMs use approximations $\hat{m}_{ij}$ and $\hat{v}_{ij}$ that are designed to vary without risking large oscillations, meaning, free from large step changes. We do this by limiting the change in $\hat{m}_{ij}$ and $\hat{v}_{ij}$ from one epoch to the next. Furthermore, we also ensure that both $\hat{m}_{ij}$ and $\hat{v}_{ij}$ remain bounded by specifying respective maximal values $\hat{m}_{\max}$ and $\hat{v}_{\max}$.



Each application and ZM in our simulator is implemented as an independent thread and triggered by means of a thread pool [25]. As already stated, simulation progresses by discrete time steps denoted as epochs. Each epoch consists of two steps, and each one of these steps is implemented as a map/reduce operation. Simulation state is synchronized after each reduce step. We now broadly explain the two main steps of each simulation epoch by focusing in turn on the EZs and the AMs (see Fig. 3). The first step, denoted as the execution step, implements functionality to allow EZs to receive bids from all AMs, choose the winning set of bids, and calculate the QoS experienced by application users. Each EZ can then report to each AM the total number of bids N_i it received, the total number of service slots M_i it offers, and the distribution FV I (v_{ij}) of its received bid values. The second step, denoted as the bid planning step, implements functionality to allow each AM to evaluate (1), (6) and (7) as a function of m_{ij} and v_{ij} . Then, each AM can determine the optimal m_{ij} , v_{ij} pair that minimizes its expected total cost as expressed by (8), and use this information to drive its bidding behaviour.

- 1) Initialize simulation - generate EZs and AMs according to QoE tolerance and population density distributions
- 2) Execution step
 - a) AMs make bids and send to EZs
 - b) Winning bids determined and results disseminated
- 3) Bid planning step
 - a) AMs calculate the number and value of bids they would optimally make next round (if no other bids changed)
 - b) AMs adjust their bidding towards this optimal number/value using Approach 1, 2 or 3 from previous section
- 4) Return to step 2)

III. COMPETITION, QOS AND PRICING

The first variable that we examine is the average cost incurred by each AM. Each subgraph in Fig. 4 corresponds to a combination of values as defined in Table II (see the top- and right-most edges of each panel). Because resource allocation is performed by means of a Vickrey auction, the total cost is defined by the losing bids in each EZ (the reader is reminded that in our multi-item Vickrey auction model a bidder winning n bids will pay the sum of the first n losing bids). Hence, in situations where there is a significant overprovisioning of resources, cost will remain close to zero. Such is the case for $N_{App_QoS} = 10$, $N_{App_Price} = 10$ and $N_{App_BG} = 10$; in this case, the cost paid is very low for all AM classes. As competition increases, however, we see that Quality-sensitive managers are much more aggressive in trying to secure EZ resources, and consequently end up paying higher costs.

IV. SCALABILITY CONSIDERATIONS

One of the challenges faced by many auction based systems is that they can exhibit high communications and/or computational complexity [17]. This means that, depending on the way in which the auction is resolved, a very large

number of either messages or computations are required to achieve an optimal result. Hence, many of our engineering choices will be aimed to reducing the communications and computational overhead of the system. First, as detailed above, we eschew global optimality in which a central optimizer performs the optimal allocation in favour of a system based on local optima for each EZ. This breaking of the entire market into a number of local markets allows the computational problem to be massively parallelized, and has been shown to impose a tolerable impact on the quality of the solution [26].

V. DISCUSSION

The convergence approaches used (see Section III-F) were all tested in simulation with a variety of parameters. For reasons of space, only one of these can be presented in this paper. The cases studied were Approach 2 with values of w between 0.05 and 0.5 and Approach 3 with similar values for w but with d between 0.9 and 0.99. In these cases the final results (after all iterations) were nearly identical. Two exceptions were noted. Using Approach 3 (where the rate of change of bids shrank with time) the algorithm could simply stop moving bids too soon and end on a very different result if w shrank quickly (low values of d). This is as would be expected since that algorithm is designed to stop all changes in bidding as time progresses. With Approach 2 then higher values of w (for example 0.5) give end results which oscillate slightly more than the results shown but still giving a similar pattern overall.

VI. CONCLUSIONS

Cloud applications have different requirements in terms of cost and quality, and different applications will coexist in the same cloud infrastructure that have different tradeoffs between these. The mechanism presented in this article allows different applications to implement their own tradeoffs transparently by abstracting away the task of allocating resources between applications with differing requirements to an auction mechanism. This allows the system to be responsive to the needs of each service, without the need for specialized treatment. In addition, because applications themselves use the prices provided by the system as signals to drive their own cost/quality tradeoffs, they experience better outcomes than those that would be possible in a static system. Many properties of the system are predictable, because they are rooted on closed-form solutions of the Vickrey auction, a well-researched mechanism. In addition, we carry out extensive simulations to verify that the system as a whole (that is, the global market in which applications bid for resources according to their needs) is both stable and efficient. We find that it is stable in the sense that, if the input parameters of the system do not change, it converges to an equilibrium state. It is efficient in the sense that resources are allocated to those applications that need them the most, and in those EZs where they provide the best improvement to the quality experienced by its users. In the process of reaching an efficient equilibrium, the system will track any changes in its inputs, and hence, it will adapt to the changing needs of those applications using it. This provides self-tuning, that is, the ability of the system to naturally configure its resources to best fit the demand profiles imposed by heterogeneous populations of cloud users.

REFERENCES

- [1] L. M. Vaquero and L. Roderio-Merino, "Finding your way in the fog: Towards a comprehensive definition of fog computing," SIGCOMM Comput. Commun. Rev., vol. 44, no. 5, pp. 27–32, Oct. 2014. [Online]. Available: <http://doi.acm.org/10.1145/2677046.2677052>
- [2] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, Algorithmic Game Theory. New York, NY, USA: Cambridge University Press, 2007.
- [3] K. Krauter, R. Buyya, and M. Maheswaran, "A taxonomy and survey of grid resource management systems for distributed computing," Softw. Pract. Exper., vol. 32, no. 2, pp. 135–164, Feb. 2002.
- [4] W. Wang and B. Li, "Market-driven bandwidth allocation in selfish overlay networks," in Proc. of IEEE INFOCOM, 2005, pp. 2578–2589.
- [5] R. Landa, R. Clegg, E. Mykoniati, D. Griffin, and M. Rio, "A Sybilproof Indirect Reciprocity Mechanism for Peer-to-Peer Networks," in Proc. Of INFOCOM, 2009.
- [6] R. Gupta, V. Sekhri, and A. K. Somani, "CompuP2P: An Architecture for Internet Computing Using Peer-to-Peer Networks," IEEE Trans. Parallel and Distrib. Syst. (TPDS), vol. 17, no. 11, 2006.
- [7] C. A. Waldspurger, T. Hogg, B. A. Huberman, J. O. Kephart, and W. S. Stornetta, "Spawn: A Distributed Computational Economy," IEEE Trans. Software Eng., vol. 18, no. 2, pp. 103–117, 1992.
- [8] J. Shneidman, C. Ng, D. C. Parkes, A. AuYoung, A. C. Snoeren, A. Vahdat, and B. Chun, "Why Markets Could (But Don't Currently) Solve Resource Allocation Problems in Systems," in Proc. of ACM HotOS. USA: USENIX - (TCOS), June 2005.

- [9] Y. Fu, J. Chase, B. Chun, S. Schwab, and A. Vahdat, "SHARP: an architecture for secure resource peering," in Proc. of ACM Symp. On Op. Sys. Princ. (SOSP), 2003.
- [10] M. Feldman, K. Lai, and L. Zhang, "A Price-Anticipating Resource Allocation Mechanism for Distributed Shared Clusters," in Proc. ACM EC, Jun. 2005.
- [11] D. Levin, K. LaCurts, N. Spring, and B. Bhattacharjee, "BitTorrent is an auction: analyzing and improving BitTorrent's incentives," in Proc. of SIGCOMM, 2008, pp. 243–254.
- [12] B. N. Chun, P. Buonadonna, A. AuYoung, C. Ng, D. C. Parkes, J. Shneidman, A. C. Snoeren, and A. Vahdat, "Mirage: A Microeconomic Resource Allocation System for SensorNet Testbeds," in Proc. Of EmNetsII, 2005.
- [13] M. Stillwell, D. Schanzenbach, F. Vivien, and H. Casanova, "Resource allocation algorithms for virtualized service hosting platforms," J. Parallel Distrib. Comput., vol. 70, no. 9, pp. 962–974, Sep. 2010.
- [14] A. Karve, T. Kimbrel, G. Pacifici, M. Spreitzer, M. Steinder, M. Sviridenko, and A. Tantawi, "Dynamic placement for clustered web applications," in Proc. of ACM WWW, USA, 2006, pp. 595–604.
- [15] S. Zaman and D. Grosu, "A Distributed Algorithm for the Replica Placement Problem," IEEE Trans. Parallel and Distrib. Syst. (TPDS), vol. 22, no. 9, pp. 1455–1468, 2011.
- [16] W. Vickrey, "Counterspeculation, Auctions, and Competitive Sealed Tenders," The Journal of Finance, vol. 16, no. 1, pp. 8–37, 1961.
- [17] P. Cramton, Y. Shoham, and R. Steinberg, Combinatorial Auctions. MIT Press, 2006.
- [18] M. Yokoo, "False-name bids in combinatorial auctions," ACM SIGecom Exchanges, vol. 7, no. 1, pp. 48–51, 2007.
- [19] Q. Wang, B. Ye, B. Tang, S. Guo, and S. Lu, "ebay in the clouds: False-name-proof auctions for cloud resource allocation," in Distributed Computing Systems (ICDCS), 2015 IEEE 35th International Conference on. IEEE, 2015, pp. 153–162.
- [20] H. A. David and H. N. Nagaraja, Order Statistics, ser. Wiley Series in Probability and Statistics. Wiley-Interscience, 2003.
- [21] N. Balakrishnan and C. R. Rao, Order Statistics: Theory and Methods, ser. Handbook of Statistics. Elsevier, 1998, vol. 16.
- [22] D. Bertsekas and R. Gallager, Data Networks, 2nd ed. P. Hall, 1992.
- [23] D. Bertsekas, "Dynamic behavior of shortest path routing algorithms for communication networks," IEEE Trans. on Auto. Control, vol. 27, no. 1, pp. 60–74, 1982.
- [24] Z. Wang and J. Crowcroft, "Analysis of Shortest-path Routing Algorithms in a Dynamic Network Environment," SIGCOMM Comput. Commun. Rev., vol. 22, no. 2, pp. 63–71, Apr. 1992.
- [25] R. P. Garg and I. Sharapov, Techniques for Optimizing Applications - High Performance Computing. Prentice-Hall, 2002.
- [26] S. M. Kakade, M. Kearns, and L. E. Ortiz, "Graphical Economics," Learning Theory (Springer LNCS), vol. 3120, pp. 17–32, 2004.
- [27] P. Wendell, J. W. Jiang, M. J. Freedman, and J. Rexford, "DONAR: Decentralized Server Selection for Cloud Services," in SIGCOMM, 2010.
- [28] Y. Shoham and K. Leyton-Brown, Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations. Cambridge University Press, 2009.
- [29] A. Jin, W. Song, P. Wang, D. Niyato, and P. Ju, "Auction Mechanisms Toward Efficient Resource Sharing for Cloudlets in Mobile Cloud Computing," Transactions on Services Computing, 2015.
- [30] F.-E. Zaheer, J. Xiao, and R. Boutaba, "Multi-provider service negotiation and contracting in network virtualization," in Network Operations and Management Symposium (NOMS), 2010 IEEE, 2010, pp. 471–478.
- [31] J.-W. Jang, M. Jeon, H.-S. Kim, H. Jo, J.-S. Kim, and S. Maeng, "Energy Reduction in Consolidated Servers through Memory-Aware Virtual Machine Scheduling," IEEE Trans. on Computers, vol. 60, no. 4, pp. 552–564, 2011.
- [32] M. B. Reynolds, D. R. Hulce, K. M. Hopkinson, M. E. Oxley, and B. E. Mullins, "A Bin Packing Heuristic for On-Line Service Placement and Performance Control," IEEE Transactions on Network and Service Management, vol. 10, no. 3, pp. 326–339, 2013.
- [33] C. Adam, R. Stadler, C. Tang, M. Steinder, and M. Spreitzer, "A Service Middleware that Scales in System Size and Applications," in Proc. Of IFIP/IEEE IM (Intl. Symp. on Int. Net. Man.), 2007, pp. 70–79.
- [34] C. Adam and R. Stadler, "Service middleware for self-managing largescale systems," IEEE Transactions on Network and Service Management, vol. 4, no. 3, pp. 50–64, 2007.
- [35] F. Wuhib, R. Stadler, and M. Spreitzer, "A gossip protocol for dynamic resource management in large cloud environments," IEEE Transactions on Network and Service Management, vol. 9, no. 2, pp. 213–225, 2012.
- [36] H. Moens, J. Famaey, S. Latré, B. Dhoedt, and F. De Turck, "Design and evaluation of a hierarchical application placement algorithm in large scale clouds," in Proc. of IFIP/IEEE IM (Intl. Symp. on Int. Net. Man.), 2011, pp. 137–144.
- [37] H. Moens and F. De Turck, "A scalable approach for structuring largescale hierarchical cloud management systems," in Proc. of IEEE/IFIP CNSM (Int. Conf. on Net. and Serv. Man.), 2013, pp. 1–8.
- [38] J. Famaey, W. De Cock, T. Wauters, F. De Turck, B. Dhoedt, and P. Demeester, "A latency-aware algorithm for dynamic service placement in large-scale overlays," in Proc. of IFIP/IEEE IM (Intl. Symp. on Int. Net. Man.), 2009, pp. 414–421.
- [39] B. Jennings and R. Stadler, "Resource management in clouds: Survey and research challenges," Journal of Network and Systems Management, pp. 1–53, 2014.
- [40] C.-W. Cheng, J.-J. Wu, and P. Liu, "QoS-aware, access-efficient, and storage-efficient replica placement in grid environments," J. Supercomput., vol. 49, no. 1, pp. 42–63, Jul. 2009.
- [41] A. Benoit, V. Rehn-Sonigo, and Y. Robert, "Replica Placement and Access Policies in Tree Networks," IEEE Trans. Parallel and Distrib. Syst. (TPDS), vol. 19, no. 12, pp. 1614–1627, 2008.