2010 International Conference on Complex, Intelligent and Software Intensive Systems

Scalable Grid Resource Trading with Greedy Heuristics

Georg Buss, Kevin Lee, Daniel Veit
Dieter Schwarz Chair of Business Administration, E-Business and E-Government
University of Mannheim, Schloss
Mannheim, Germany
{*buss,lee,veit*}*@bwl.uni-mannheim.de*

***Abstract*—As Grid infrastructures become more widely used by the academic and commercial world, the problem of resource allocation increases in complexity. Resource trading markets are one mechanism that allows many resource owners and resource consumers to trade. To perform efficiently trading markets for grids require approaches to match consumers and producers. Solutions for optimal and non-optimal resource trading exist, but fail to scale effectively to meet the challenges of large numbers of traders. This paper first defines the problem of scalable resource trading in grids before describing and evaluating greedy approaches for scalability.**

***Keywords*-heuristics, grid, trading, markets, allocation**

I. Introduction

The vision of Grid computing emerged out of the need from science for vast on-demand processing resources not available at a single location [1]. As the underlying technology matured added value of remote computational power for industry was discovered [2]. Both, industry and science, benefit from leveraging the economics of scale provided by remotely accessible, aggregated computational resources. Besides access to resources the infrastructure allows for offering not utilized capacities to others. However these Grid infrastructures, often loosely coupled, regionally distributed and heterogeneous in nature require complex human and tool interactions to unlock their resource power.

Recent advances in fundamental technologies enable a seamless and more secure exchange of computational services in Grids [3], [4], [5]. This has improved the Grid platform as a whole, allowing system administrators to integrate Grids into their IT resources easier. Improvements to user tools, high level scientific support and application development has enabled a larger group of technical and non-technical users to use Grids.

Despite vast improvements in platform development, user support and infrastructure investment there are still many issues to be resolved. An open question is how to efficiently share computational resources. In situations in which resource demand exceeds resource supply, traditional management techniques based on the maximization of resource utilization fail to provide a fair utility maximizing allocation [6]. Because of this resources are often inefficiently utilized. Market based approaches can be used to allocate resources more efficiently based on the buyers valuation of the resources [6], [7]. An overview of approaches for market based resource allocation in Grids is given in [8].

The focus of this paper is on the multi attribute combinatorial exchange mechanism [9]. The mechanism allows for simultaneous trading of buyers and sellers, the submission of combinatorial, multi attributive bids and supports domain specific constraints. The main drawback of the mechanism is the $\mathcal{NP}$-completeness of the underlying multi-attribute winner determination problem (MWDP). The MWDP can become computationally intractable for small instances with less than hundred bidders [9]. However, the nature of Grids requires a scalable allocation mechanism to increase the size of the market for computational resources.

In the literature three options to tackle the winner determination problem (WDP) in combinatorial auctions have been identified [10]:

1) The restriction of the expressiveness of the bidding language
2) The design of specialized tree search algorithms that provably find the optimal solution
3) The design of approximation algorithms

The restriction of the bidding language would sacrifice the domain requirements. The application of tree search algorithm was found to become computational intractable for small scale scenarios. In this paper we investigate the application of approximation algorithms in the form of greedy algorithms to the MWDP. We present a formal definition of the problem of resource trading and the requirements of a market based solution. We introduce a representation of the problem and further present and evaluate three greedy approaches solving the problem in a scalable way.

The remainder of this paper is structured as follows. In Section II we discuss related work in the field of heuristic optimization approaches. Section III describes the MWDP problem formally. In Section IV we introduce three greedy algorithm-based solutions to the problem. Section V presents the results of an experimental evaluation of the different approaches. Section VI presents some conclusions.

II. Related Work

The MWDP is formulated as a generalization of the $\mathcal{NP}$-complete combinatorial allocation problem (CAP) [11]. Approximation algorithms that provide bounds on the quality

978-0-7695-3967-6/10 $26.00 © 2010 IEEE
DOI 10.1109/CISIS.2010.146

of the solution are not considered because even the CAP as well as the combinatorial exchange problem are found to be inapproximable [12], [13]. For inapproximable problems no reasonable bound can be provided for the quality of the solution. Thus a scalable solution for many deals requires the investigation of approaches that provide an approximately efficient solution to the MWDP.

A survey of optimization approaches reveals no heuristic approaches to the MWDP, although algorithms for the WDP in combinatorial auctions exist. In [14], [15] greedy algorithms are proposed to approximate the winner determination problem in combinatorial auctions based on bid sorting with the order determining the allocation order. The approach by Lehmann et al [14] serves as input for a hill climbing and simulated annealing [16] approach. Hoos and Boutilier [17] propose an algorithm based on stochastic local search which is based on scoring search states. [18] formulate WDP as a multi dimensional knapsack problem. This problem has been studied extensively in the domain of operations research [19] including using genetic algorithms [20]. However all of these approaches can not be applied directly to the MWDP as it is formulated as a generalization of the WDP.

III. Problem Description

To describe the problem of grid resource trading we take the multi-attribute combinatorial exchange mechanism described in [9]. This approach allows for trading multiple resources which can be described as attributes. In this scenario the unit of trade is computational resources including CPU cycles, data storage and RAM denoted by g. The set $G = \{g_1, \dots, g_{|G|}\}$ specifies the computational resources available in the exchange mechanisms where G denotes all the goods to be traded in a trading market and a g_k is a specific resource. A bundle S_i denotes a subset of all the resources in G. Therefore the set $S = \{S_1, \dots, S_{|S|}\}$ of bundles covers all the possible subsets of G. A computational resource g_k itself is defined by a set of cardinal quality attributes $A_k = \{a_1, \dots, a_{|A_k|}\}$.

The trading market used in this paper for this scenario is one in which sellers can sell resource bundles and buyers can bid on resource bundles. Both sellers and buyers of resources do this by placing blind orders in the marketplace; buyers specifying what resources are required, and sellers specifying what resources are available. Potential buyers n out of the set $N = \{n_1, \dots, n_{|N|}\}$ of buyers are allowed to submit an order of multiple bundle bids $B_n = \{B_{n,1}(S_1) \oplus \dots \oplus B_{n,u}(S_i)\}$. The respective bundle bids are XOR concatenated.

The submission of bundle bids allows for the expression of complementarities among bundles of computational resources. A buyer is allocated at maximum one complete bundle out of the order she placed. A single buyer bundle bid is of the form:

$$\begin{aligned} B_{n,f}(S_i) = \{ & \langle v_n(S_i), s_n(S_i), e_n(S_i), l_n(S_i), \\ & q_n(S_i, g_1, a_{g_1,1}), \dots, q_n(S_i, g_G, a_{g_G,A_j}), \\ & \gamma_n(S_i, g_1), \dots, \gamma_n(S_i, g_G), \\ & \varphi_n(S_i, g_1, g_2), \dots, \varphi_n(S_i, g_G, g_{G-1}) \rangle \} \end{aligned}$$

The valuation $v_n(S_i)$ is the amount the buyer is willing to pay for the bundle S_i per time slot. The number of slots the resources are required for is given by $s_n(S_i)$. A buyer bid defines a period of time slots within which the required slots have to be allocated. The period is given by $e_n(S_i)$ for the earliest possible time slot and $l_n(S_i)$ for the latest possible time slot. The minimum quality of the resources g_k contained in a bundle bid S_i is specified for each resource attribute a_{g_k,A_j} by $q_n(S_i, g_k, a_{g_k,A_j})$. In addition bundle bids may contain two types of fulfillment constraints. A coupling constraint $\gamma_n(S_i, g_1)$ specifies the maximum number of sellers allowed to allocate a required resource g_k. The co-allocation $\varphi_n(S_i, g_k, g_j)$ constraint requires a pair of resources g_k, g_j to be allocated from the same single seller.

Potential sellers m out of the set of $M = \{m_1, \dots, m_{|M|}\}$ may submit an order of multiple bundle bids $B_m = \{B_{m,1}(S_i) \vee \dots \vee B_{m,u}(S_i)\}$. The bundle bids are OR concatenated. Any number of seller orders may be part of the final allocation. A single seller bundle bid is of the form:

$$\begin{aligned} B_{m,f}(S_i) = \{ & \langle r_m(S_i), e_m(S_i), l_m(S_i), \\ & q_m(S_i, g_1, a_{g_1,1}), \dots, q_m(S_i, g_G, a_{g_G,A_j}), \rangle \} \end{aligned}$$

The reservation price $r_m(S_i)$ specifies the minimum price a seller is willing to sell the specified bundle of resources per time slot. It is assumed that a seller bid is valid for the range of time slots given by $e_m(S_i)$ and $l_m(S_i)$. The quality of the resource services g_k is given by $q_m(S_i, g_k, a_{g_k,A_j})$.

Given a collection of buyer and seller bundle orders the MWDP is to identify a set of winning bids out of the total set of bids. An optimal set of winning buyer and seller bids determines an allocation that maximizes the overall surplus while meeting time, capacity, coupling and co-allocation constraints. An allocation is described by the variables $x_{n,t}(S_i) \in \{0, 1\}$ and $y_{n,m,t}(S_i) \in [0, 1]$. The binary variable $x_{n,t} = 1$ if buyer n is allocated bundle S_i in time slot t. The real valued variable $y_{m,n,t}$ denotes the percentage of bundle S_i allocated from seller m to buyer n in time slot t. The surplus of an allocation is given by:

$$\begin{aligned} (x, y) \in arg\ max \Bigg(& \sum_{n \in N} \sum_{S_i \in S} \sum_{t \in T} v_n(S_i) x_{n,t} - \\ & \sum_{m \in M} \sum_{n \in N} \sum_{S_i \in S} \sum_{t \in T} r_m(S_i) y_{m,n,t} \\ & |(x, y)\ is\ a\ feasible\ allocation \Bigg) \end{aligned}$$

This description includes free disposal (buyers do not care about taking extra units, sellers do not care about keeping units of winning bids) except when resources are coupled.

IV. Greedy Optimization Approaches

Construction heuristics are the fastest kind of heuristic to identify a feasible solution to a given problem. The construction process can be divided into several phases. After an initialization phase the construction process is typically continued by a selection and a placement phase until a termination condition is met. In this paper we concentrate on greedy construction heuristics. Greedy heuristics construct a solution to a problem by making at each construction step a locally optimal decision. The locally optimal decision must not necessarily be optimal from a global point of view.

A problem instance is represented as depicted in Figure 1. The buyer orders are split up into the single bundle bids $B_{n,f}(S_i)$. The single buyer bundle bids are stored in a sequence. For each buyer bid $B_{n,f}(S_i)$ a list of possible time slots t is kept. For each of these time slots the available seller bundle bids $B_{m,f}(S_i)$ are listed. The overall idea is to reduce the $|n| : |m|$ allocation problem to be scheduled to into a given number of time slots to a $1 : |m|$ allocation problem to be solved for a single time slot t. The $1 : m$ allocation problem for a given buyer bundle bid $B_{n,f}(S_i)$ and a given time slot t can be formalized as follows:

$$y \in arg\ max \left(v_n(S_i) - \sum_{m \in M} \sum_{S_i \in S} r_m(S_i) y_{m,n} \right. \\ \left. |y\ is\ a\ feasible\ allocation \right)$$

In case of no coupling or collocation constraints the problem becomes a linear, continuous, optimization problem. This type of problem can be solved efficiently by a linear programming solver. If coupling or co-allocation constraints are present the problem is of combinatorial nature with complexity reduced significantly compared to the original $|n| : |m|$ allocation problem.

The problem representation is evaluated by passing through the sequence of buyer bundle bids starting with the first bundle bid. The time slots the buyer bid is valid for $(e_n(S_i), l_n(S_i))$ are checked in the given order. A check of a time slot requires solving the, possibly constrained, $1 : |m|$ allocation problem. As soon as the amount of computational resources requested is available for a sufficient number of time slots the buyer bid is included into the allocation and the construction process is continued. A check of a time slot is valid only if there is a valid solution to the $1 : |m|$ allocation problem. Therefore no infeasible solution can be encoded by the problem representation. In case the buyer is already part of the allocation with another bid (XOR constraint) the evaluation of the specific bid is skipped and the process is continued with the next bid in the sequence.

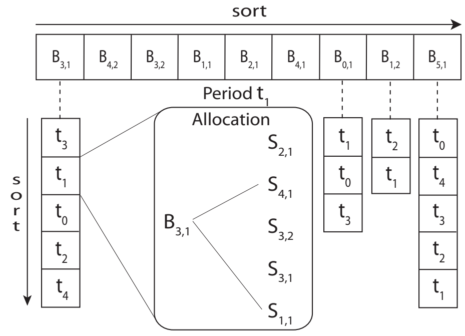

Figure 1. Problem representation

The greed algorithms proposed start with an empty allocation. In the initialization phase buyer bundle bids and the respective time slots are sorted according to a set of criteria. The sorting of the buyer bundle bids is inspired by the domain of knapsack problems [19, p. 257]. The basic procedure is to sort the buyer bundle bids in a way that the currently most promising buyer bundle bid is included in each construction step of the greedy heuristic. Prior to introducing the ordering procedures several concepts have to be introduced. The factor $flex = \frac{s_n(S_i)}{l_n(S_i) - e_n(S_i)}$ measures the flexibility of a buyer bundle bid in a time scheduling sense. A bundle bid is considered to be more flexible the more options there are for time scheduling. The higher the flexibility the closer $flex$ is to zero. The factor $c(S_i) = \left(\sum_{S_i \ni g_k} \max_{a_{g_k,j} \in A_j} q_n(S_i, g_k, a_{g_k,j}) \right)$ measures the average consumption of resources for a given bundle bid S_i by aggregating the maximum quality requirements for each resource. The set B_{sl} is defined as the set of all sellers m that offer the set or a subset of resources bundled in the bundle S_i in time slot t. The parameter $wac(S_i) = \frac{\sum_{S_i \ni g_k} \sum_{e_n(S_i)}^{l_n(S_i)} \max_{a_{g_k,j} \in A_j} \frac{q_n(S_i, g_k, a_{g_k,j})}{\sum_{B_{sl} \ni S_j} q_m(S_j, g_k, a_{g_k,j})}}{l_n(S_i) - e_n(S_i)}$ measures the weighted average consumption of resources of resources per time slot in dependency of the amount of resources offered.

The attractiveness of a buyer bundle bid can be assessed by the descending order according to the following criteria:

1) $v_n(S_i)$: The order of buyer bundle bids is determined by the respective bundle valuations.
2) $\frac{v_n(S_i)s_n(S_i)}{flex*c}$: The order of buyer bundle bids is based on the scaled valuations. Valuations are scaled by the flexibility of a bundle bid and by the average consumption of resources.
3) $\frac{v_n(S_i)s_n(S_i)}{flex*wac}$: The order of buyer bundle bids is based on adjusted valuations. In comparison to the previous ordering procedure the requested amount of resources is weighted by the available supplies. In consequence the demand for scarce resources is more significant in reducing the attractiveness of a buyer bundle bid.

The options for sorting the slots of a buyer bundle bid are:

1) $\frac{\sum_{S_j \ni B_{sl}} \frac{r(S_j)}{|g_k \in S_j|}}{|B_{sl}|}$: The time slots are sorted according to the ascending average seller reservation price (asr) for a single resource.
2) $asr \ * \ \sum_{S_i \ni g_k} \max_{a_{g_k,j} \in A_j} \frac{q_n(S_j, g_k, a_{g_k,j})}{\sum_{B_{sl} \ni S_j} q_m(S_i, g_k, a_{g_k,j})}$: The time slots are sorted according to the ascending average seller reservation price for a single resource weighted by the aggregated average resource demand versus supply ratios. The demand versus supply ratio for a single resource is defined as the sum of maximum ratios between demand quality and the aggregated supply qualities of all resources
3) optimal solution: Sorts the time slots according to the result of the optimal solution to the $1 : m$ allocation problem in a descending order.

Out of the nine possible combinations of bundle bid and slot sorting procedures the combinations 1/1, 2/2, 3/3 are chosen for evaluation. The combinations are designed according to the trade off of complexity of evaluation versus the power of additional information included into the sorting process. The goal is the analysis of the trade off described. As a lower bound benchmark a greedy algorithm based on random sorting procedures for buyer bids and slots is used. To solve the $1 : m$ allocation problem optimally a mixed integer linear programming solver is used. The random combination is chosen as a benchmark.

V. Experimental Evaluation

The three greedy heuristics presented in section III are evaluated by means of stochastic simulation. The approaches are compared according to the quality of the solution as well as CPU time required. To assess whether the greedy heuristics are superior to simply random they are benchmarked to a random greedy solution. In addition the results are compared to the results of a mixed integer linear programming solver applied to the analytical benchmark problem [9].

The experiments performed are based on six bidding scenarios each differing by the number of seller and buyer bundle bids. For each problem instance the number of seller bids matches the number of buyer bids. The scenarios comprise 20, 40, 60, 80, 100 and 120 seller and buyer bundle bids. The number of goods is set to five and are characterized by three attributes each. A bundle may contain any combination of goods. The earliest possible time slots are of the interval $[1, 5]$. The latest possible time slots are distributed between $[1, 12]$. The number of time slots required is distributed in the interval of $[1, 3]$. No coupling or co-allocation constraints are present. The time limit for the calculation of a result was set to three minutes.

For each of the scenarios described 50 instances are generated. To generate bids and valuations the combinatorial auction test suite (CATS) is used [21], [22]. CATS was designed to generate bids that are realistic in real world scenarios. Meaningful bundle bids require for example that certain goods are grouped more likely in a bundle than others. CATS offers five different options for the generation of bundle bids. For the paper at hand the arbitrary option was chosen. This option models the domain of trading computational resources best as arbitrary but constant complementarities between goods are assumed. If for example a service offering cpu-cyles is requested it is very likely that a storage service is requested too. The quality attributes are drawn from a normal distribution. The according mean and variance values depend on the valuation or the respective reservation price as well as the number of resources that are present in the bundle. The time attributes are drawn from a uniform distribution within the intervals introduced.

All experiments were performed on a Intel Core 2 CPU 1.86 Ghz with 1.49 GB RAM running Windows XP. The lp-solve [23] mixed integer linear programming solver version 5.5.0.14 with default parameters was used. The solver was configured to present intermediary results if the optimum was not found in time. The maximum time to compute a solution was set to 180s.

Two types of experiments were performed differing in the number of buyers submitting bundle bids. For type A of experiments the number of buyers equals the number of bundle bids submitted. Type B of experiments contains XOR concatenated bundle bids. The scenarios of type B have only half the number of buyers, submitting two bundle bids each. Therefore the number of bundle bids is equal for the respective scenarios of both experiments. For each of the experiments the solution to the MWDP and the time needed to compute the solution are recorded. Table I and table II show the results for the runtime of the experiments. The tables summarize the mean runtime (μ) and the standard deviation (σ) for each of the scenarios processed with the different algorithms. For the analytical benchmark solution an additional column shows the number of KO. A KO indicates that the analytical benchmark solution ran into the timeout of 180 seconds. There is no guarantee that the solution found is the optimal solution. A KO is differentiated into two subcategories. The first category indicates that a timeout occurred but a solution is returned. The second category indicates that the solver was not able to compute any solution in the given time frame. The Figures 2 and 3 summarize the mean solution quality (surplus) for each of the experiments. The solution quality is measured in comparison to analytical benchmark solution (100 percent).

A. Experiment A

The runtime results of experiment A show that show that within the time frame the optimal solution to the MWDP is computable for all of the 10/10 instances only. Starting from the number of 40 agents intermediary results are provided for some of the problem instances. The number of not optimal solutions increases from the 20/20 scenarios to the

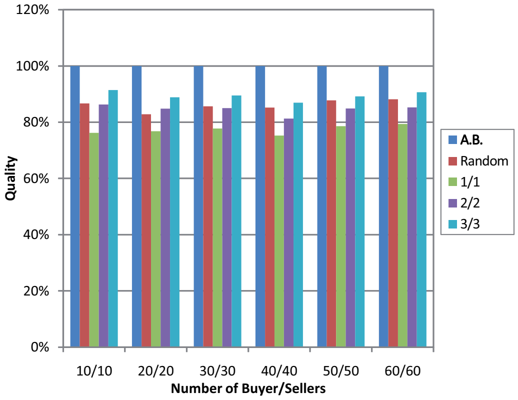

Figure 2. Mean Solution Quality Experiment A

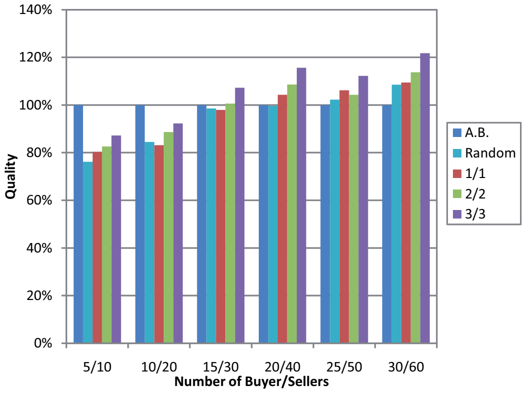

Figure 3. Mean Solution Quality Experiment B

60/60 scenarios to nearly 47 which is 94 percent. For all of these problem instances the execution of the solver has been cut off after 180 seconds. In comparison none of the greedy approaches required on average more than 4.6 seconds to compute a solution. The results for the solution quality show that none of the greedy approaches matched the solution of the analytical benchmark solution. In comparison of the greedy approaches the 3/3 algorithm performed best delivering on average 90 percent of the solution quality of the analytical benchmark solution. The 2/2 approach performs similar to the random strategy but requires more computational effort. The 1/1 approach requires a higher computational effort than the random algorithm delivering a results of inferior quality.

Table I
RUNTIME RESULTS EXPERIMENT A

	Analytical benchmark solution			Random		1/1		2/2		3/3	
Num. Bids	μ	σ	KO	μ	σ	μ	σ	μ	σ	μ	σ
10/10	2999	19856	0/0	40	17	38	14	44	15	232	47
20/20	49723	72929	11/0	103	27	109	29	129	27	664	120
30/30	133906	72121	33/0	197	42	203	42	239	35	1254	177
40/40	165324	45054	45/0	303	46	315	50	398	47	2065	223
50/50	172540	31068	47/0	460	78	488	103	627	88	3247	477
60/60	176740	23164	49/0	640	94	677	105	882	97	4601	621

B. Experiment B

Experiment B shows that the optimal solution to the MWDP can not be computed within the given time frame for 24 problem instances of 40 agents. The number of non optimal solutions for problem instances of a category increases to the 30/60 scenarios to 100 percent. Starting from the 15/30 scenario there are problem instances no solution is provided for. In comparison non of the greedy approaches required on average more than 4.2 seconds to compute a solution. The results for the solution quality show that the 3/3 and 2/2 algorithms outperform the random benchmark solution. The 1/1 algorithm performs equal to the random approach. All of the greedy approaches outperform the analytical benchmark solution from different degrees of scenario complexity on. In comparison of the greedy algorithms the 3/3 approach is the best in terms of solution quality. In terms of runtime the 2/2 approach is about the factor 6.5 faster than the 3/3 algorithm reaching 94 percent of its solution quality.

Table II
RUNTIME RESULTS EXPERIMENT B

	Analytical benchmark solution			Random		11		22		33	
Num. Bids	μ	σ	KO	μ	σ	μ	σ	μ	σ	μ	σ
5/10	5253	24212	0/0	27	16	30	16	29	15	197	66
10/20	121401	71390	24/0	67	26	74	29	76	26	593	110
15/30	177898	14975	47/2	120	38	140	36	145	42	1164	176
20/40	180000	0	47/3	193	51	212	57	245	49	1899	276
25/50	173429	30719	45/3	259	58	296	62	355	58	2938	488
30/60	180000	0	47/3	373	92	405	98	480	89	4272	538

C. Summary

The results of both experiments clearly indicate that all of the greedy solutions presented are less expensive in terms of computational effort than solving the analytical benchmark problem. An interesting observation comparing type A and type B experiments is that the quality of the greedy random solution highly depends on the type of experiment. A possible explanation is that in case of XOR bids the ordering of buyer bids (type B experiments), which decides about the inclusion into the allocation, becomes more important. Consequently the technique for ordering buyer bundle bids matches the problem structure. In turn the techniques used for ordering the slots may have to be improved to better fit the problem domain. Comparing the runtime of both experiments it is to be noted that type B experiments require more computational effort for the analytical benchmark solver. In contrast the greedy solutions provide the results faster than in type A experiment. This is because in the XOR case the check for inclusion into the allocation can be skipped for half of the bids.

In summary the greedy algorithm that performed in terms of solution quality best is the 3/3 algorithm. The algorithm outperformed the random greedy algorithm for each of the scenarios of type A and B experiments. The 2/2 greedy algorithms outperform the random benchmark for each of the type B scenarios while performing slightly worse for type A experiments. The 11 approach is found to perform worse than the random solution for type A experiments and slightly superior for type B experiments. The approach turned out to be not suitable for the approximation of the MWDP. In terms of computational complexity the random solution requires least effort followed by 1/1 and 2/2 which require about the same effort. The runtime of the 3/3 approach was in mean about the factor 6.5 higher than the one of 1/1 and 2/2.

VI. Conclusions

The core contribution of our paper is the problem specific adaption and evaluation of scalable greedy heuristics addressing the allocation problem in Grids formulated as a MWDP. Three greedy type algorithms were defined and tested on specific instances of the problem. In a quantitative comparison we have demonstrated the effectiveness of the 3/3 greedy algorithm outperforming the random benchmark solution and the results to the analytical benchmark solution for complex scenarios.

We plan to test the greedy type algorithms on additional problem instances to improve the sorting procedures further. In a next step we plan to use the results of greedy search as a starting point for a hill climbing approach or more sophisticated heuristics as simulated annealing or tabu search.

References

[1] I. Foster. The grid: A new infrastructure for 21st century science. *Physics Today*, 55(2):42–47, 2002.

[2] N. G. Carr. The end of corporate computing. *MIT Sloan Management Review*, 46(3):67–73, 2005.

[3] I. Foster. Globus toolkit version 4: Software for service-oriented systems. In *IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779*, pages 2–13, 2005.

[4] Open Grid Forum. http://www.ogf.org.

[5] V. Welch, F. Siebenlist, I. Foster, J. Bresnahan, K. Czajkowski, J. Gawor, C. Kesselman, S. Metier, L. Pearlman, and S. Tuecke. Security for grid services. In *Twelfth International Symposium on High Performance Distributed Computing (HPDC-12*, pages 48–57. IEEE Press, 2003.

[6] 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 *Proceedings of 10th on Hot Topics in Operating Systems*, 2005.

[7] K. Lai. Markets are dead, long live markets. *SIGecom Exch.*, 5(4):1–10, 2005. 1120719.

[8] J. Broberg, S. Venugopal, and R. Buyya. Market-oriented grids and utility computing: The state-of-the-art and future directions. *Journal of Grid Computing*, 6(3):255–276, 2008.

[9] B. Schnizler, D. Neumann, D. Veit, and C. Weinhardt. Trading grid services - a multi-attribute combinatorial approach. *European Journal of Operational Research*, 187(3):943–961, 2008.

[10] D. Lehmann, R. Müller, and T. Sandholm. The winner determination problem. In P. Cramton, Y. Shoham, and R. Steinberg, editors, *Combinatorial Auctions*, pages 297–317. The MIT Press, Cambridge, 2006.

[11] M. H. Rothkopf, A. Pekeč, and R. M. Harstad. Computationally manageable combinational auctions. *Management Science*, 44(8):1131–1147, 1998.

[12] T. Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135(1-2):1–54, 2002.

[13] M. Babaioff, P. Briest, and P. Krysta. On the approximability of combinatorial exchange problems. In *Algorithmic Game Theory*, Lecture Notes in Computer Science, pages 83–94. Springer, Berlin/Heidelberg, 2008.

[14] D. Lehmann, L. I. O'Callaghan, and Y. Shoham. Truth revelation in approximately efficient combinatorial auctions. *Journal of the ACM*, 49(5):577–602, 2002.

[15] E. Zurel and N. Nisan. An efficient approximate allocation algorithm for combinatorial auctions. In *Proceedings of the 3rd ACM conference on Electronic Commerce*, pages 125–136. ACM, 2001.

[16] N. Fukuta and T. Ito. Towards better approximation of winner determination for combinatorial auctions with large number of bids. In *International Conference on Intelligent Agent Technology*, pages 618–621, 2006.

[17] H. H. Hoos and C. Boutilier. Solving combinatorial auctions using stochastic local search. In *Proceedings of AAAI-00*, page 2229, 2000.

[18] R. Holte. Combinatorial auctions, knapsack problems, and hill-climbing search. In *14th Conference of the Canadian Society for Computational Studies of Intelligence*, Advances in Artificial Intelligence, Berlin, 2001.

[19] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer, Heidelberg, 2004.

[20] P. C. Chu and J. E. Beasley. A genetic algorithm for the multidimensional knapsack problem. *Journal of Heuristics*, 4(1):63–86, 1998. 10.1023/A:1009642405419.

[21] K. Leyton-Brown, M. Pearson, and Y. Shoham. Towards a universal test suite for combinatorial auction algorithms. In *In ACM Conference on Electronic Commerce*, pages 66–76, 2000.

[22] K. Leyton-Brown and S. Yoav. A test suite for combinatorial auctions. In P. Cramton, Y. Shoham, and R. Steinberg, editors, *Combinatorial Auctions*. The MIT Press, Cambridge, 2006.

[23] Lpsolve 5.5.0.14, a mixed integer linear programming (milp) solver, http://lpsolve.sourceforge.net/.