Matchmaking in Multi-attribute Auctions using a Genetic Algorithm and a Particle Swarm Approach

Simone A. Ludwig[1] and Thomas Schoene[2]

[1]*Department of Computer Science, North Dakota State University, Fargo, North Dakota, U.S.A.*
simone.ludwig@ndsu.edu
[2]*Department of Computer Science, University of Saskatchewan, Saskatoon, Saskatchewan, Canada*
ths299@mail.usask.ca

An electronic market platform usually requires buyers and sellers to exchange offers-to-buy and offers-to-sell. The goal of this exchange is to reach an agreement on the suitability of closing transactions between buyers and sellers. This paper investigates multi-attribute auctions, and in particular the matchmaking of multiple buyers and sellers based on five attributes. The proposed approaches are based on a Genetic Algorithm (GA) and a Particle Swarm Optimization (PSO) approach to match buyers with sellers based on five attributes as closely as possible. Our approaches are compared with an optimal assignment algorithm called the Munkres algorithm, as well as with a simple random approach. Measurements are performed to quantify the overall match score and the execution time. Both, the GA as well as the PSO approach show good performance, as even though not being optimal algorithms, they yield a high match score when matching the buyers with the sellers. Furthermore, both algorithms take less time to execute than the Munkres algorithm, and therefore, are very attractive for matchmaking in the electronic market place, especially in cases where large numbers of buyers and sellers need to be matched efficiently.

1 Introduction

The impact of E-commerce trading is rising rapidly due to the enhancement of the internet and the customer's need to comfortably search and buy products online. E-commerce trading is more efficient than alternative methods because of active pricing mechanisms, up-to-date databases, and streamlined procurement processes.

An electronic market platform usually requires buyers and sellers to exchange offers-to-buy and offers-to-sell. The goal of this exchange is to reach an agreement on the suitability of a transaction between buyers and sellers. A transaction transfers one or more objects (e.g. a product, money, etc.) from one agent to another and vice versa. The transaction can be described by sets of properties such as the delivery date for the transaction, the color of the object, or the location of the agent. A property has a value domain with one or more values [1].

Two kinds of the negotiating strategies commonly used are distinguished by the relationship with markets [2]:

- Cooperative Negotiation: For cooperative negotiations, multiple items of trading attributes are negotiable (quality, quantity, etc.). Because the participants have their own preferences on different trading attributes, it is possible for the two parties to obtain satisfactory results out of the bargain.
- Competitive Negotiation: For this kind of negotiation, the objectives of both sides are conflicting. When one side gets more benefit out of a certain bargain, the other side will face some loss. This is a zero-sum game from the point of view of the game theory. Auction is an example of this kind of negotiation.

Hence, cooperative negotiation is a better bargaining method for two parties, as each can obtain a satisfactory result. Many trading attributes can be coordinated in such a bargain (quality, quantity, payment, etc.), and the participants can negotiate based on their preferences.

Automatic negotiation plays the most important role among processes in an E-marketplace as it seeks to maximize benefits for both sides. In advanced multi-agent systems, when a buyer and a seller are interested in trading with each other, both will be represented by agents who may hold opposite grounds initially, and then will start to negotiate based on available information in order to reach common ground. Two critical challenges are faced here. The first one is to provide a global platform in which efficient searching, publishing, and matching mechanisms can be enforced in order to minimize the load and make processes more efficient. The second challenge is to come up with autonomous processes that can capture essential human negotiation skills such as domain expertise, learning and inference. During the matching process, parties advertise offers-to-buy or offers-to-sell. These offers include consumer/provider properties and constraints [3]. Constraints expressed by one party represent the reservation value set on some aspect of a given transaction. The reservation value is the minimum value the party wants to achieve, and thus is similar to the reservation price in an auction [4].

Electronic negotiations are executed in the intention and agreement phase of an electronic market. A definition of matchmaking in electronic negotiations can be found in [5]. The steps in the intention phase are as follows:

- Offer and request specification: The agents have to specify offers and requests indicating their constraints and preferences towards the transaction object. This specification may also include the provision of signatures or the definition of timestamps. This specification can be executed instancing a candidate information object that has been designed for this specific marker.
- Offer and request submission: To submit an offer or a request, the agent can actively send the offer or request to a specific agent (middle agent) or notify the middle agent of the completion of the specification.
- Offer validation: When the middle agent receives the offer or request, the information object is checked for completeness and the compliance with certain rules.

The steps in the agreement phase are defined as follows:

- Offer and request matching: The aim of this phase is to find pairs of offers and requests that stratify potential counterparts for a transaction execution. This includes the identification of all offers that match a given request. In this phase, within the matchmaking framework, a ranking of all offers with respect to the current request is computed and returned as a ranked list to the requesting agent.
- Offer and request allocation: In this task the counterparts for a possible transaction is determined using the information from the matching and scoring phase. The duties of the single agents are determined and assigned. The final configuration has to be determined in this phase if the selected offers and requests still feature certain value ranges or options.
- Offer and request acceptance: This final stage confirms the acceptance of the terms and conditions, which have been determined. The agents have to accept the conditions in order to execute the transactions and complete the deal.

Electronic auctions, also referred to as internet auctions, are also widely studied [6,7,8]. Internet auctions are seen as an electronic mechanism, which provides a digital version of allocation mechanisms. Electronic auctions define an economic mechanism that is designed to solve competitive resource allocation problems. Generally speaking, an electronic auction is the implementation of auction protocols using electronic media. In electronic markets, many dimensions can be considered that are too complex to express in non-electronic negotiations. Bichler et al. [9] state that the item characteristics are an important factor for determining the type of the appropriate negotiation and matchmaking mechanism to be applied. There are several terms that provide a framework for the design of negotiation characteristic referring to concrete item characteristics:

- Multi-phased auctions: Several phases are carried out determining the auctions outcome.
- Multi-stage auctions: Similar to multi-phased auctions, several stages have to be passed before the auction terminates. In this case the order of the stages is relevant.
- Multi-unit auctions: Multi-unit auctions describe auctions in which several units of the same object are auctioned.
- Multi-item auctions: Multi-item auctions describe auctions in which several, possibly heterogeneous, items are auctioned.
- Multi-attribute auctions: As Bichler [10] defines multi-attribute auctions as auctions by which the overall score computation is not limited to bids on the mere price, but several aspects of a good can be bid on. Usually, a virtual currency is introduced to provide the overall score, which in turn is mapped to a price.
- Multi-dimensional auctions: Bichler and Werthner [11] see multi-dimensional auctions as an umbrella term for multi-unit, multi-item and multi-attribute auctions.

Matchmaking plays a crucial role within electronic auctions. Within each bidding procedure a winner has to be determined. In single-attribute auctions, where only the price can be bid on, the highest price wins. In this case, no sophisticated matchmaking mechanism has to be introduced. In multi-attribute actions however, where several attributes are bid on, mechanisms are needed to compute an overall score. In general, the more attributes and sub-attributes are provided, i.e. the more complex the bid structure is defined, the more complex matchmaking procedures have to be introduced.

This paper is organized as follows. The following section introduces related work that has been done in the past. Section 3 describes the approaches implemented. In Section 4, the experiment setup and the results are given. Section 5 concludes this paper with the findings and gives an account to future research.

2 Related Work

Many matchmaking models and frameworks have been introduced in the past and are introduced below.

SILKROAD [12] presents a matchmaking framework that is based on constraint satisfaction. The offers and requests can be supported in a subject-based structure. This enables a wide variety of application domains. However, the matchmaking mechanism is limited.

The INSPIRE system [13] provides communication support among offering parties and requesting parties to submit individual preferences. The matchmaking is performed by the parties that accept or reject an offer or request. The advantage of this system is the openness in negotiating the position. However, as system-based matchmaking is missing, the aim of the system is not to provide a complete matchmaking procedure, but to provide matchmaking support to the participating parties.

INSULA [14] provides a rule-base matchmaking unit. Several attributes can be supported in a domain specific way. These attributes are then matched using the constraints of the attributes of the counterpart applying the matching rules. This design limits the matchmaking complexity, but enables domain specific attributes.

The EMS framework by Stroebel and Stolze [1] contains a matchmaking unit that allows free definable offer and request structures. These structures are application dependent and adaptable. A disadvantage that limits the application domains of this framework is that the offer and request attributes are matched only based on the constraints and no discrete values are allowed.

The SHADE approach by Kuokka and Harada [15] defines one of the first generic free text matchmakers. This system has the advantage to provide distance functions from information retrieval that make it fairly flexible and domain independent. The main disadvantage of this system is that it does not provide mechanisms to match structure offer and requests.

The IMPACT matchmaker by Subrahmanian et al. [16] is based on a simple offer and request structure. It allows only verb-noun terms, consisting of two verbs and a noun as offer and request structures. On the other hand, hierarchies enable a powerful matchmaking that can also be applied in specific domains. However, it is limited due to its fixed offer and request structures.

LARKS [17] is one of the most powerful matchmaking approaches known so far. It provides several matchmaking stages responsible for different processes, which enables high matchmaking quality. However, the application domains are limited as the offer and request structure is static.

The GRAPPA framework [18] combines the benefits of a generic approach with the key advantages of domain specific solutions. The GRAPPA framework is explicitly defined to enable flexibility of generated offer and request structures as well as to be completely domain independent by supporting distance function and metric interfaces that allow easy integration of domain dependent and generic functions.

The aim of the research provided in this paper has a slightly different focus. First of all, it is envisioned that matchmaking support will increase in future, not only because electronic market places are seen to becoming more and more utilized, but also the number of participants in the marketplace (buyers and sellers) will rise gradually. Therefore, a robust, time-efficient and scalable assignment algorithm is needed to perform the task of matchmaking. One optimal algorithm, known as the Munkres algorithm, has a cubic time complexity and therefore, does not scale well with increasing numbers of buyers and sellers. Thus, approximate algorithms are necessary, which on one hand provide an optimized assignment, and on the other hand scale linearly with increasing numbers of buyers and sellers.

3 Matchmaking Approaches

Multi-attribute auctions not only use the price of the item, but a combination of different attributes of the deal, such as delivery time, terms of payment, product quality, etc. However, this requires a mechanism that takes multiple attributes of a deal into account when allocating it to a participant. The mechanism should automate multi-lateral auctions/negotiations on multiple attributes of a deal. Three matchmaking approaches are presented in the following subsections. The matchmaking function is introduced first. Then, the GA algorithm, PSO algorithm, Munkres algorithm and the Random approach are explained in detail.

3.1 Matchmaking Function

The problem of matchmaking in an electronic marketplace is having an effective algorithm that can match multiple buyers and sellers efficiently, while optimizing multiple attributes. The problem is twofold: firstly, multiple buyers requesting the same product should be satisfied, and secondly, the assignment process of the buyers and the sellers should be optimized. Please note that one buyer can only be matched with one seller.

Matchmaking is concerned with matching buyers with sellers based on a range of negotiation attributes. The five objective measures or negotiation attributes are: quality, quantity, price, delivery and payment. We assume that sellers have a fixed value s_i for each attribute i and the buyers have a fixed value b_i for each attribute i as well. The match value for each attribute i is calculated as follows based on the difference between b_i and s_i:

$$v_i = 1 - |b_i - s_i| \tag{1}$$

The match score m_j of one buyer-seller pair is the sum of all five match values multiplied by the weight value w_i for attribute i, divided by the number of negotiation attributes. The weight value w_i allows specifying preferences on the different attributes:

$$m_j = \frac{1}{n}\sum_{i=1}^{n}\omega_i v_i \tag{2}$$

The matchmaking algorithms implemented use Equations (1) and (2) to calculate the match score for each buyer-seller pair. However, the aim of this research is to match several buyers with several sellers as closely as possible. Therefore, the overall match score of p buyers and p sellers is defined as:

$$o = \frac{1}{p}\sum_{j=1}^{p} m_j \tag{3}$$

3.2 Genetic Algorithm

GA is a global optimization algorithm that models natural evolution [19]. In GA, individuals form a generation. An individual (similar to a particle in the PSO) corresponds to one match. The match is implemented as a vector, which is also referred to as a chromosome. Dimensions in the vector correspond to sellers, and values correspond to buyers. Thus, if the vector has value 3 at its 5th position (dimension), buyer 3 is matched with seller 5. Every number representing a buyer can only be present at one position in the vector, otherwise, the vector is a non-valid match.

At the beginning, the first population is randomly initialized. After that, the fitness of the individuals is evaluated using the fitness function (Equations (1)–(3)). After the fitness is evaluated, individuals have to be selected for paring. The selection method used is tournament selection. Always two individuals are paired, resulting in an offspring of two new individuals. In the pairing phase, a random crossover mask is used, i.e. the positions (dimensions) for which crossover occurs are selected randomly. If crossover occurs at certain positions (dimensions), individuals that are mated exchange their values at that position and the resulting individuals are used as offspring. The crossover has to make sure that the offspring present a valid match. Therefore, if two values are exchanged, other positions in the two match vectors are usually effected as well. The offspring faces mutation with a certain low probability. After mutation, the fitness of the offspring is calculated. Then, either all individuals from the last generation compete against the whole offspring, or the offspring only compete with its corresponding parents. In this implementation, all individuals from the old generation compete with all individuals in the new generation. To achieve this all individuals are ordered by their fitness score, using a non-recursive advanced quicksort algorithm. After sorting, the lower half is truncated. After the new generation is selected, the GA will start over, and continue with parent selection and crossover again.

Configurable parameters in the implementation include number of iterations, tournament size (the size of the tournament used to select parents), crossover probability, effected positions (how many positions are set to crossover in the crossover mask), and mutation probability.

3.3 Particle Swarm Optimization Approach

PSO, as introduced in [20], is a swarm based global optimization algorithm. It models the behavior of bird swarms searching for an optimal food source. The movement of a single particle is influenced by its last movement, its knowledge, and the swarm's knowledge. In terms of a bird swarm this means, a bird's next movement is influenced by its current movement, the best food source it ever visited, and the best food source any bird in the swarm has ever visited.

PSO's basic equations are:

$$x_i(t+1) = x_i(t) + v_{ij}(t+1) \tag{4}$$

$$\begin{aligned} v_{ij}(t+1) = {} & w(t)v_{ij}(t) + c_1 r_{1j}(t)\left(xBest_{ij}(t) - x_{ij}(t)\right) \\ & + c_2 r_{2j}(t)\left(xGBest_j(t) - x_{ij}(t)\right) \end{aligned} \tag{5}$$

where x represents a particle, i denotes the particle's number, j the dimension, t a point in time, and v is the particle's velocity. $xBest$ is the best location the particle ever visited (the particle's knowledge), and $xGBest$ is the best location any particle in the swarm ever visited (the swarm's knowledge). w is the inertia weight and used to weigh the last velocity, c_1 is a variable to weigh the particle's knowledge, and c_2 is a variable to weigh the swarm's knowledge. r_1 and r_2 are uniformly distributed random numbers between zero and one. PSO is usually used on real and not discrete problems. In order to solve the discrete assignment problem using the PSO approach, several operations and entities have to be defined. This implementation follows and adapts the implementation for solving the traveling salesman problem as described in [21]. First, a swarm of particles is required. A single particle represents a match, i.e., every particle's position in the search space must correspond to a possible match. The match, that is the position, is implemented as a vector. Dimensions in the vector correspond to sellers, and values correspond to buyers. Therefore, if the vector has value 3 at its 5th position (dimension), buyer 3 is matched with seller 5. Every number representing a buyer has to be unique, otherwise, the vector represents a non-valid match.

Velocities are implemented as lists of changes that can be applied to a particle (its vector) and will move the particle to a new position (a new match). Changes are exchanges of values, i.e., an entity containing two values that have to be exchanged within a vector. This means that any occurrence of the first value is replaced by the second value, and any occurrence of the second is exchanged by the first value. Further, minus between two matches (particles), multiplication of a velocity with a real number, and the addition of velocities have to be defined. Minus is implemented as a function of particles. This function returns the velocity containing all changes that have to be applied to move from one particle to another in the search space. Multiplication is implemented as a function of velocities. Multiplication randomly deletes single changes from the velocity vector, if the multiplied real number is smaller than one. If the real number is one, no changes are applied. For a real number larger than one, random changes are added to the velocity vector. Addition is also implemented as a function of velocities. When a velocity is added to another velocity, the two lists containing the changes will be concatenated.

The implemented PSO uses guaranteed convergence, which means that the best particle is guaranteed to search within a certain radius, implying that the global best particle will not get trapped in local optima.

Configurable parameters in the implementation include numbers of particles (size of the swarm), number of iterations, c_1 (the weighting of the local knowledge), c_2 (the weighting of the global knowledge), w (the weighting of the last velocity), radius (defines the radius in which the global best particles searches randomly), global best particle swarm optimization (determines whether global best particle swarm or local best particle swarm optimization is used), and neighborhood size (defines the neighborhood size for local best particle swarm optimization).

3.4 Munkres Algorithm

The Hungarian algorithm is a combinatorial optimization algorithm that solves the assignment problem in polynomial time. It was developed by Harold Kuhn in 1955 [22,23], who gave the name "Hungarian method" because the algorithm was largely based on the earlier works of two Hungarian mathematicians: Denes Koenig and Jeno Egervary.

In 1957 James Munkres reviewed the algorithm and observed that it is (strongly) polynomial. Since then, the algorithm has been known also as Kuhn-Munkres algorithm or Munkres assignment algorithm [24,25]. The time complexity of the original algorithm was $O(n^4)$, however, Edmonds and Karp, and independently Tomizawa noticed that it can be modified to achieve an $O(n^3)$ running time.

The Munkres algorithm is used to serve as a benchmark for the negotiation matchmaking as it is an optimal algorithm. However, one drawback that the Munkres algorithm has is the time complexity of $O(n^3)$ as previously mentioned, and hence is not very time efficient. The assignment problem as formally defined by Munkres [24]:

"Let r_{ij} be a performance ratings for a man M_i for job J_i. A set of elements of a matrix are said to be independent if no two of them lie in the same line ("line" applies both to a row and a column of a matrix). One wishes to choose a set of n independent elements of the matrix (r_{ij}) so that the sum of the element is minimum."

Similarly, the problem of matchmaking can be defined as an $n \times m$ buyer-seller matrix, representing the match scores of each buyer with every other seller. The match score matrix is the matrix where each element of the matrix represents the match score for an individual buyer-seller pair. The Munkres algorithm works on this matrix, to assign the buyer requests to the sellers, as to achieve an overall maximum total match score. Please note that one buyer can only be matched with one seller.

An implementation developed by Nedas in Java (freely available at [26]) was used and slightly adopted to serve as a benchmark for this matchmaking investigation, as it provides the base match score of the optimal assignment of buyer and seller pairs.

3.5 Random Approach

The random approach, as the name indicates, randomly selects and assigns buyer-seller pairs. Sellers and buyers are held in separate vectors and one buyer and one seller is randomly selected and matched up until both vectors are empty and all sellers and buyers are uniquely assigned.

4 Experiments and Results

All four algorithms as introduced in the previous section were implemented using Java. Experiments were designed to measure the overall match score and the execution time of all approaches. The GA and PSO algorithms were furthermore analyzed with regards to the population size (GA algorithm), as well as with regards to the particle sizes used (PSO algorithm). All measurement points shown are average results taken from 30 runs, in order to guarantee statistical equal distribution. The data sets for the buyers and sellers were randomly generated.

The following parameters are used throughout the experiments if not stated otherwise: The number of buyer-seller pairs is set to 500; for the GA algorithm the population size is set to 1000, the crossover and mutation probabilities are set to 0.6 and 0.05 respectively, the tournament size is 10%, and elitism is set; for the PSO algorithm the iteration is set to 100, the number of particles is set to 10, the weight is set to 0.001, the local and global constants are both set to 0.5, and guaranteed convergence is enabled.

Figure 1 shows the overall match score of all approaches. The optimal matchmaking of 100% is achieved by the Munkres algorithm, followed by the GA algorithm with an average match score of 91.34±4.37, followed by the PSO algorithm with an average match score of 87.03±4.62%, and 79.82±0.63% achieved by the random approach.

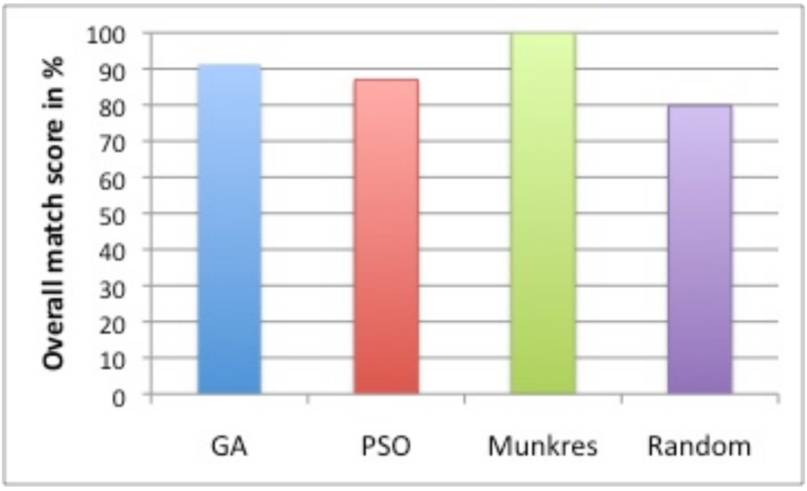

Figure 1. Overall match score of all approaches.

Figure 2 shows the execution time in seconds of all approaches. The Munkres algorithm has the longest running time with 212.97 seconds, followed by the PSO algorithm with 54.29 seconds. The GA algorithm is fairly fast with a run time of 35.59 seconds, whereas the fastest algorithm is, as expected, the random approach with 5.93 milliseconds (not visible on Figure 2 because its execution time is too small).

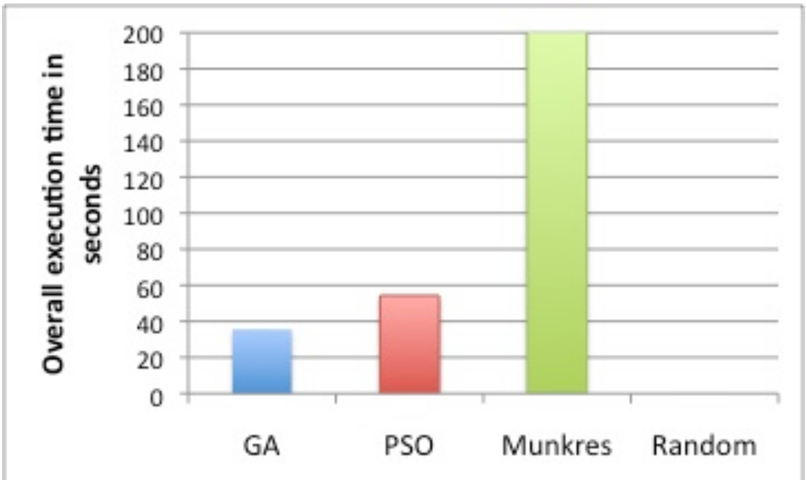

Figure 2. Overall execution time of all approaches.

Figure 3 shows the overall match score for the increasing numbers of buyer-seller pairs. As expected, it can be seen that the GA as well as the PSO algorithm have higher match score values (GA is outperforming the PSO algorithm) than the random approach, and it can be stated that the match scores are slightly higher for smaller numbers of buyer-seller pairs.

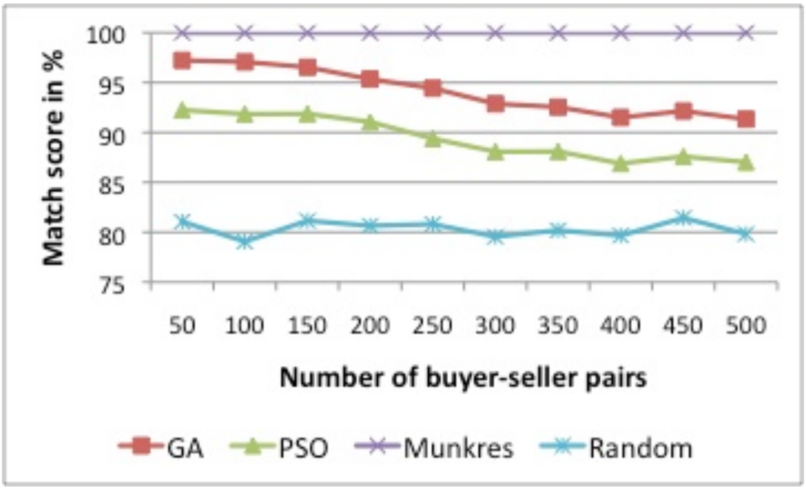

Figure 3. Match score of all algorithms for increasing numbers of buyer-seller pairs.

Figure 4 shows the execution time of the algorithms for the increasing numbers of buyer-seller pairs. As mentioned before, the execution time of the Munkres algorithm is cubic as can clearly be observed. The GA and PSO algorithm as well as the Random approach are less time-consuming; whereby the Random approach outperforms all algorithms showing the smallest execution time.

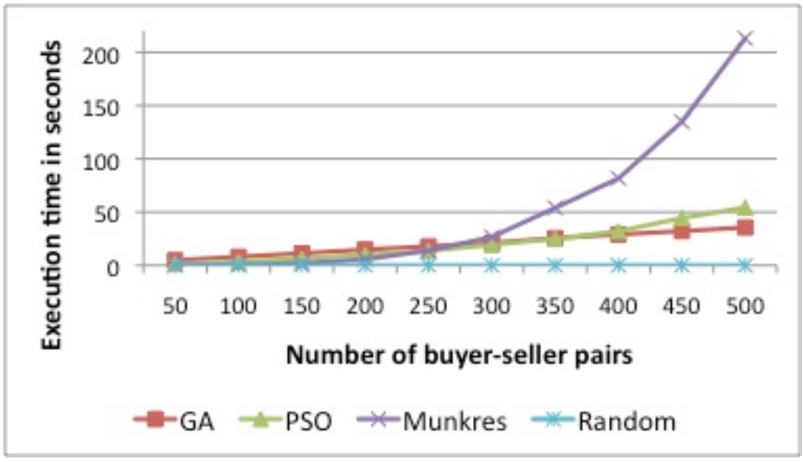

Figure 4. Execution time of all algorithms for increasing numbers of buyer-seller pairs.

Figure 5 shows the overall match score for an increase in the number of iterations. As the increase in the number of iterations has only an effect on the GA and PSO algorithm, the Munkres and Random approaches are only plotted for comparison reasons. It can be seen that the overall match score of the GA algorithm as well as for the PSO algorithm is increasing with the increase in the number of iterations.

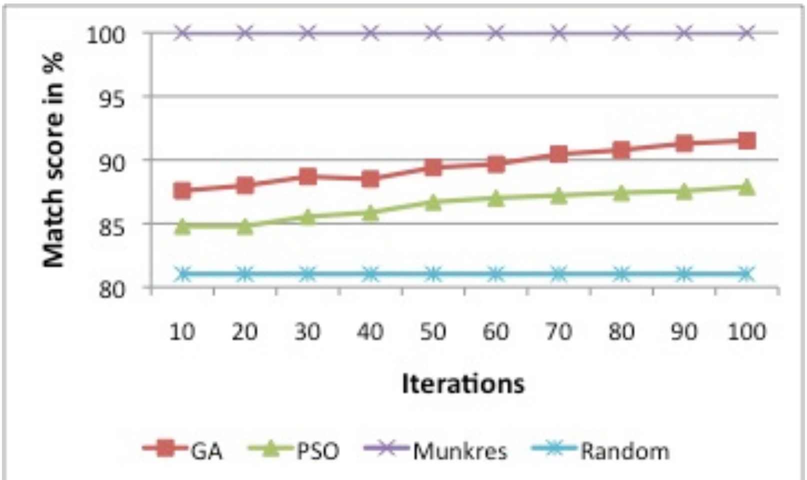

Figure 5. Match score of all algorithms for increasing numbers of iterations.

Figure 6 shows the execution time for an increase in the number of iterations. It can be seen that the execution time of the GA, PSO and Random approach are much smaller than the Munkres approach. Both GA and PSO algorithm scale linearly with increasing numbers of iterations.

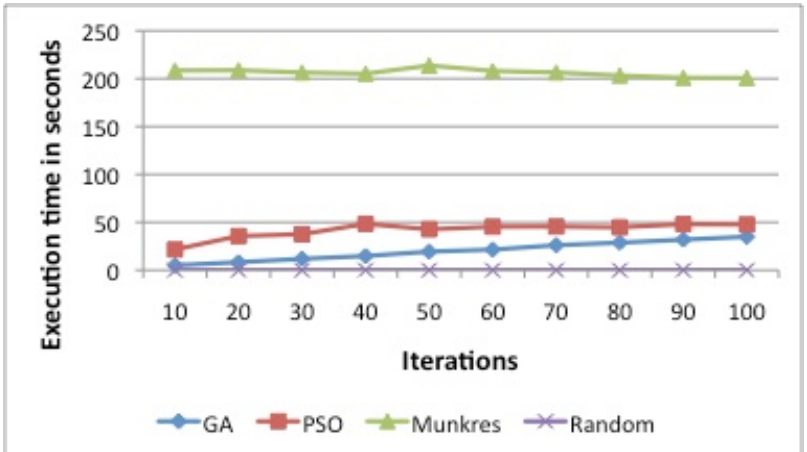

Figure 6. Execution time of all algorithms for increasing numbers of iterations.

Figure 7 displays the match score for increasing population size of the GA algorithm. Population sizes of 250 up to 2500 were investigated and the match score varies between 83.71% and 85.77%, as observed with the previous measurements.

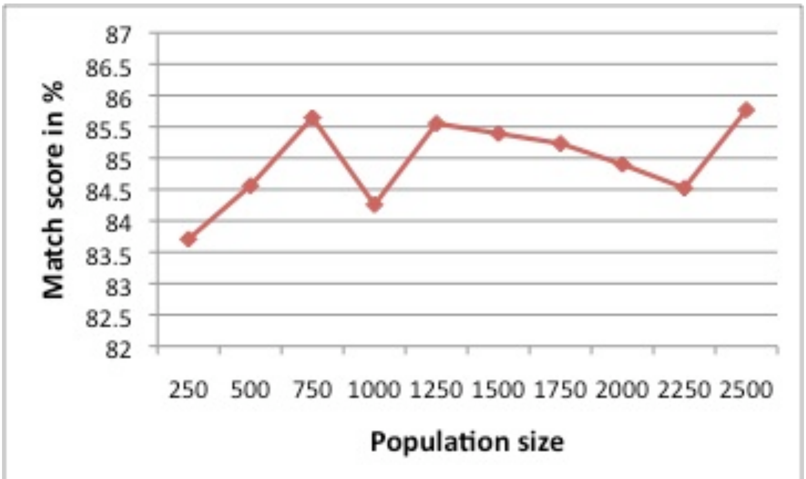

Figure 7. Match score of GA algorithm for increasing numbers of population size.

Figure 8 shows the execution time for increasing population size of the GA algorithm. Population sizes of 250 up to 2500 were investigated and the execution time shows a linear increase with larger population sizes.

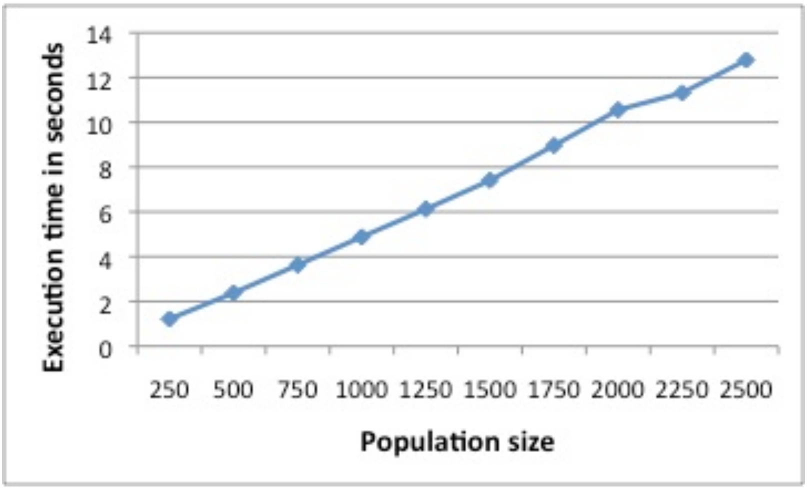

Figure 8. Execution time of GA algorithm for increasing numbers of population size.

Figure 9 shows the match score of the PSO algorithm for increasing numbers of particles. The match score varies between 82.62% and 84.33% as previously observed.

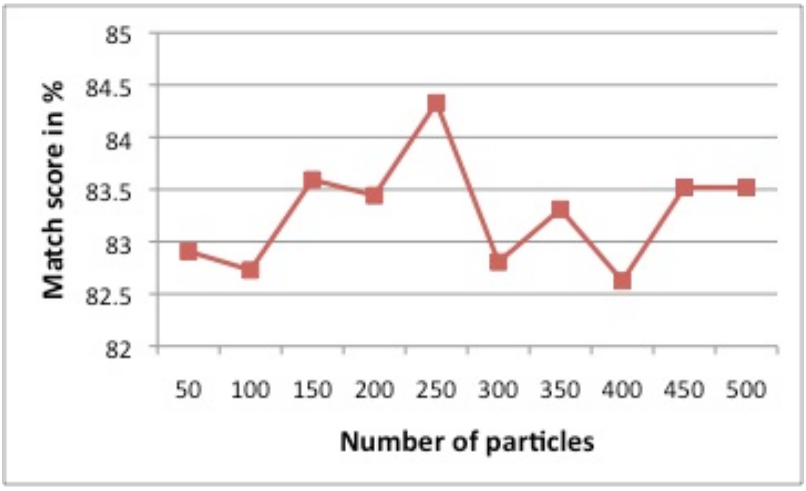

Figure 9. Match score of PSO algorithm for increasing numbers of particles.

Figure 10 shows the execution time of the PSO algorithm for increasing numbers of particles. It can be seen that the execution time increases linearly with increasing numbers of particles. An execution time of 72.69 seconds is measured for the particle size of 500.

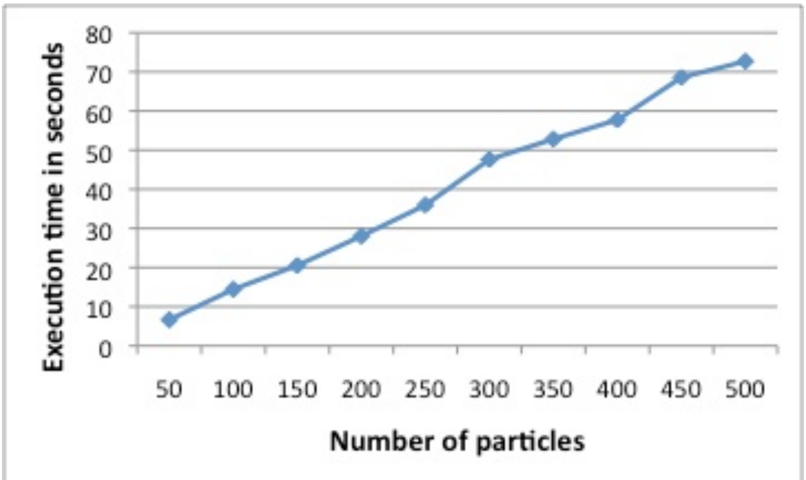

Figure 10. Execution time of PSO algorithm for increasing numbers of particles.

5 Conclusion

This paper investigated four approaches for the matchmaking of multi-attribute auctions, and in particular the matchmaking of multiple buyers and sellers based on five attributes. The scenario was envisioned, that in future the number of multi-attribute auctions will rise, and therefore, an efficient algorithm is necessary to provide this matchmaking functionality. The Munkres algorithm provides an optimal assignment, however, it has a cubic computational time complexity, and thus, does not scale very well. Therefore, other approximate approaches were investigated such as a GA-based approach as well as a PSO approach was chosen. Overall, the two approximate algorithms performed fairly well, reducing the execution time by 17% with the GA algorithm and 25% with the PSO algorithm for 500 buyer-seller pairs, thereby achieving match scores of 91.1% and 87.0% respectively. The GA algorithm in particular achieved higher match scores as well as having shorter execution times. It seems that the theory following the evolutionary principles works very good for the scenario of matchmaking. Even though the PSO approach achieves higher match scores than the Random approach, the application of swarm intelligence is outscored by the evolutionary approach (GA).

As a recommendation, given that the demand for matchmaking services is rising, approximate algorithms are necessary as in most auction scenarios time is of essence. Therefore, the GA and PSO algorithm should be chosen, in particular the GA algorithm. However, if the time is not critical and if the match quality of buyers and sellers is paramount, then the Munkres algorithm should be chosen.

Future work will follow three directions. First of all, as the GA algorithm always outperformed the PSO algorithm, the PSO algorithm will be extended to include the selection of Pareto fronts, which will most likely increase the match quality. The Pareto front works as follows: the most fit individuals from the union of archive and child populations are determined by a ranking mechanism (or crowded comparison operator) composed of two parts. The first part 'peels' away layers of non-dominated fronts, and ranks solutions in earlier fronts as better. The second part computes a dispersion measure, the crowding distance, to determine how close a solution's nearest neighbors are, with larger distances being better. It will be employed to search for better match sequences, which will guide the evolutionary process toward solutions with better objective values.

Secondly, different matchmaking functions will be tested, e.g. to allow a buyer to specify a request with an upper and lower limit of the attributes.

The third direction will be to investigate whether the GA and PSO algorithms can be made to perform faster by distribution and parallelization in order to achieve execution times closer to the Random approach.

References

Ströbel, M. and Stolze, M. (2002). A Matchmaking Component for the Discovery of Agreement and Negotiation Spaces in Electronic Markets. *Group Decision and Negotiation*, 11:165-181.

Guttman, R.H. and Maes, P. (1998). Cooperative vs. Competitive Multi-Agent Negotiations in Retail Electronic Commerce. *Cooperative Information Agents II, Springer*.

Hoffner, Y. and Schade, A. (1998). Co-operation, Contracts, Contractual Matchmaking and Binding. *Proceedings EDOC 98*, IEEE, Piscataway, pp. 75-86.

Ströbel, M. (2001). Communication design for electronic negotiations on the basis of XML schema. *Proceedings of the tenth international conference on World Wide Web*, Hong Kong, pp.9-20, May 01-05.

Stroebel, M. and Weinhardt, C. (2003). "The Montreal Taxonomy for Electronic Negotiations." *Group Decision and Negotiation*, 12(2): 143-164.

Kumar, M. and Feldman, S. (1998). Internet auctions. In Proceedings of the 3rd USENIX Workshop on Electronic Commerce, Boston, USA, pp. 49–60.

Koppius, O. (1998). Electronic multidimensional auctions: Trading mechanisms and applications. In V. Homburg, M. Janssen, and M. Wolters (Eds.), Edispuut Workshop’98: Electronic Commerce — Crossing Boundaries, Rotterdam, The Netherlands.

Koppius, O., Kumar, M. and van Heck, E. (2000). Electronic multidimensional auctions and the role of information feedback. In H. R. Hansen, M. Bichler, and H. Mahrer (Eds.), Proceedings of the 8th European Conference on Information Systems (ECIS’2000), Volume 1, Vienna, Austria, pp. 461–468. Association for Information Systems.

Bichler, M., Kaukal, M., and Segev, A. (1999). Multi-attribute auctions for electronic procurement. In Proceedings of the 1st IBM IAC Workshop on Internet Based Negotiation Technologies, Yorktown Heights, USA, pp. 154–165.

Bichler, M. (2000). Trading financial derivatives on the web - an approach towards automating negotiations on OTC markets. Information Systems Frontiers 1 (4), 401–414. Kluwer Academic Publishers.

Bichler, M. and Werthner, H. (2000). Classification framework for multidimensional, multi-unit procurement auctions. In Proceedings of the DEXA Workshop on Negotiations in Electronic Markets, Greenwich, U.K., pp. 1003–1009.

Stroebel, M. (2001). Design of roles and protocols for electronic negotiations. Electronic Commerce Research Journal, Special Issue on Market Design 1(3), 335–353.

Lo, G. and Kersten, G. (1999). Negotiation in electronic commerce: Integrating negotiation support and software agent technologies. In Proceedings of the 29th Atlantic Schools of Business Conference.

Benyoucef, M., Alj, H., Vezeau, M., and Keller, R. K. (2001). Combined negotiations in e-commerce: Concepts and architecture. Electronic Commerce Research Journal - Special issue on Theory and Application of Electronic Market Design 1 (3), 277–299.

Kuokka, D. and Harada, L. (1996). Integrating information via matchmaking. Journal of Intel ligent Information Systems 6 (2-3), 261–279.

Subrahmanian, V. S., Bonatti, P., Dix, J., Eiter, T., Kraus, S., Ozcan, F., and Ross, R. (2000). Heterogenous Agent Systems. MIT Press. ISBN: 0262194368.

Sycara, K., Widoff, S., Klusch, M. and Lu J. (2002). Larks: Dynamic matchmaking among heterogeneous software agents in cyberspace. Autonomous Agents and Multi-Agent Systems 5, 173203.

Veit, D., Müller, J. P., Schneider, M., and Fiehn, B. (2001). Matchmaking for autonomous agents in electronic marketplaces. In *Proceedings of the Fifth international Conference on Autonomous Agents* (Montreal, Quebec, Canada). AGENTS '01. ACM, New York, NY, 65-66.

Holland, J.H. (1975). Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor.

Kennedy, J. and Eberhart, R.C. (1995). Particle swarm optimization, Proceedings of IEEE International Conference on Neural Networks.

Clerc, M. (2004). Discrete particle swarm optimization – illustrated by the traveling salesmen problem, New Optimization Techniques in Engineering, Springer, 2004.

Kuhn, H.W. (2005). The Hungarian method for the assignment problem. Naval Research Logistics, 52(1).

Kuhn, H.W. (1955). The hungarian method for solving the assignment problem. Naval Research Logistics, 2:83.

Munkres, J. (1957). Algorithms for the Assignment and Transportation Problems. Journal of the Society for Industrial and Applied Mathematics, 5:32, 1957.

Bourgeois, F. and Lassalle, J.C. (1971). An extension of the munkres algorithm for the assignment problem to rectangular matrices. Commun. ACM, 14(12).

Nedas, K. (2008). Munkres' (Hungarian) Algorithm, Java implementation, last retrieved on March 2009 from http://konstantinosnedas.com/dev/soft/munkres.htm.