Proceedings of the 6th WSEAS Int. Conf. on EVOLUTIONARY COMPUTING, Lisbon, Portugal, June 16-18, 2005 (pp55-60)

Investigation of Matchmaking and a Genetic Algorithm for Multilateral and Integrative E-Negotiations

SIMONE A. LUDWIG[1], LARRY RAISANEN[1], S.M.S. REYHANI[2]
[1]Department of Computer Science, Cardiff University, Cardiff CF24 3AA, UK
[2]Department of Information Systems and Computing, Brunel University, Uxbridge, Middlesex UB8 3PH, UK

Abstract: - 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. In this paper we use multilateral and integrative e-negotiations to investigate our approach which attempts to find the best buyer-seller pairs, for an equal number of buyers and seller, using either matchmaking or a well-tested genetic algorithm: NSGA-II. The goal is to match as many buyers and sellers as closely as possible on five objectives (i.e., quality, quantity, price, delivery and payment) that vary randomly between a given range for buyers and are fixed for sellers. Experiments are performed and results are discussed for both approaches. The main finding is that there is a trade-off between solution quality and execution time: The genetic algorithm is capable of finding higher quality solutions than matchmaking when a suitable population size is employed, but matchmaking's execution time is significantly faster. This allows in turn to predict which technique to use depending on quality and speed in an e-negotiation scenario.

Key-Words: - E-negotiation, Matchmaking, Genetic Algorithm, NSGA-II.

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 colour 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 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 regarding their preferences. Automatic negotiation plays the most important role among processes in an E-marketplace as it seeks to maximise 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 minimise 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].
In general, 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 if this transaction is performed and therefore is similar to the reservation price in an auction [4].
This paper compares the approach of matchmaking in negotiations to the use of a genetic algorithm to find the best matches. The results reveal the point at which the genetic algorithm outperforms matchmaking given a certain number of buyers and sellers. In the next section we discuss different negotiations processes. Section 3 describes the two approaches and outlines the application scenario. In section 4 the experiment setup is described and the results delineated. Finally, in section 5 we summarise the finding and draw conclusions.

2 Negotiation Processes

An E-marketplace offers many participants the chance to take certain actions, such as buying, selling, or simply browsing available products. When an enterprise needs to locate new or existing partners, it must first determine the strategies it is going to establish regarding a collaboration with its partners. Many attributes can be used to classify negotiations; however, Jelassi and Foroughi categorised them into distributive versus integrative negotiations and bilateral versus multilateral negotiations [5].

2.1 Distributive versus Integrative

In distributive negotiations one issue is subject to negotiation and the parties involved have opposing interests. One party tries to minimise (to give as little as possible) and the other party tries to maximise (to receive as much as possible). Distributive negotiations are also characterised as 'win-lose' negotiations. The more one party gets, the less the other party gets. In integrative negotiations multiple issues are negotiated and the parties involved have different preferences towards these issues. Two parties may want, for example, to buy a company, but one is interested primarily in the human capital whereas the other is interested in the patent portfolio. These variant valuations can be exploited to find an agreement with joint gains for both parties. If their preferences are the same across multiple issues, the negotiation remains distributive until opposing interests are identified. In such a case, both parties can realise gains; thus, another name for this class of negotiations is 'win-win' negotiations [6].

2.2 Bilateral versus Multilateral

This aspect of negotiations refers to the number of parties participating in the negotiation. Only two parties participate in bilateral negotiations, whereas in multilateral negotiations parties may be either one-to-many or many-to-many. In addition, parties involved in multilateral negotiations can typically inspect offers from other parties (unless the offers are intentionally sealed). Similarly, multilateral negotiations are also characterised as public competitive negotiations, whereas bilateral negotiations have a private character and are therefore often referred to as cooperative negotiations.

2.3 Summary

Combinations of attributes within these two classifications can be used for a high-level design of negotiation protocols. A protocol for negotiations defines the rules by which parties come to agreement. A classification is not necessarily persistent during a real negotiation process. An integrative negotiation can be reduced to a distributive negotiation if only one issue is subjected to discussion and all other issues are temporarily fixed. On the other hand, a distributive negotiation can be extended to an integrative negotiation by adding issues to the discussion [7].

3 Approach

Our approach is based on finding the best buyer-seller pairs with the highest match score in multilateral and integrative negotiations. Let us assume we have a single product which many sellers offer and many buyers want but with different negotiation values for attributes such as quality, quantity, etc. The aim is to find an algorithm which matches buyers and sellers with each other, achieving the best match score for each of them - and as a whole.
We compare two different approaches: matchmaking and a genetic algorithm.

3.1 Matchmaking

Matchmaking is concerned with matching buyers with sellers based on a range of negotiation attributes. We assume that sellers have a fixed value (S_i) for all the attributes whereas the buyers have a range $[BU_i \ldots BL_i]$. The match value for each attribute is calculated as:

$$MV_i = \begin{cases} 0 & \text{for } BU_i \le S_i \text{ or } BL_i > S_i \\ \omega_i \cdot \left(\dfrac{BU_i - S_i}{BU_i - BL_i} \right) & \text{for } BL_i \le S_i < BU_i \end{cases} \quad (1)$$

whereby BU_i represents the buyer's upper value for attribute i; BL_i is the buyer's lower value for attribute i; S_i represents the seller's value for attribute i; ω_i is the weight value for attribute i.
The overall match score is the sum of all match values divided by the number of negotiation attributes:

$$MS = \frac{\sum_{i=1}^{n} MV_i}{n} \quad (2)$$

The matchmaking algorithm (Fig. 1) implemented uses equations (1) and (2) to calculate the match score for each buyer and seller. As each buyer might match several sellers the assignment of one buyer with one seller is carried out as follows: The algorithm takes the first buyer and determines the seller with the highest match score. This seller is removed from further consideration, as a buyer for his goods has been found. Then, the best match score to the remaining sellers is determined for the second buyer; that seller is then removed, and so on until each buyer is matched to a seller.

```
Calculation of Match Score:
for all buyers do
  for all sellers do
    for all negotiation attributes do
      calculate_MatchValue()
    end for
    calculate_MatchScore()
    store_Vec_Seller_MatchScore()
  end for
  store_Vec_Buyer_And_Vec_Seller_MatchScore()
end for
return Vec_Buyer_And_Vec_Seller_MatchScore

Best Match Assignment for Buyer-Seller Pair:
for all buyers do
  for all sellers matching a buyer do
    if matchScore is higher than previous &
       seller is not taken then
      assign_Buyer_Seller_MatchScore()
    end if
  end for
  store_Buyer_Seller_MatchScore()
end for
return Vec_Buyer_Seller_MatchScore
```

Fig. 1: Matchmaking Algorithm

The main strength of this approach is that each buyer is only considered once, while the main weakness is that buyers later in the list are very likely to get assigned with a seller having a worse match score. This is where the genetic algorithm can potentially enhance the assignment procedure, as it does not make matches in a linear fashion.

3.2 Genetic Algorithm

A genetic algorithm (GA) is a heuristic used to find approximate solutions to difficult-to-solve problems by applying the principles of evolutionary biology to computer science. GAs use biologically-derived techniques such as inheritance, mutation, natural selection, and recombination (or crossover) [8]. GAs are typically implemented as a computer simulation in which a population of solutions (or individuals) to an optimization problem evolve towards better solutions. This is possible as each solution is a chromosome which can undergo genetic modification.
In this study, each chromosome is a randomly generated permutation of integers, where each integer represents a given buyer. The length of the chromosome is therefore equal to the number of buyers in the given test circumstance. Given the sellers are a static integer permutation from 1 to *N*, each chromosome determines which buyer is matched to which seller. The number of individuals in the population is an alterable test parameter.
The process starts with a population of completely randomly generated individuals. In each generation, the fitness of each population member is evaluated. The fittest individuals, in terms of best match score - for example, form an archive population, where the best solutions found so far are saved. As even the quality of solutions here can range widely - particularly in earlier generations, members compete in binary tournaments, with winners forming a mating pool. Two parents are randomly selected from the pool, and undergo cycle crossover [9] and mutation to form two children. This is repeated until the new population of size *N* is filled. The new population is evaluated; its members compete for inclusion in the archive, and the process repeats until either a set number of generations are completed, stagnation, or termination criteria is met.

3.2.1 NSGA-II

The genetic algorithm used here is NSGA-II. NSGA-II is a fast elitist non-dominated sorting genetic algorithm. For an excellent overview of evolutionary optimization techniques, the interested reader is referred to Deb [10], and for a full description of NSGA-II to Deb et al. [11]. In brief, in NSGA-II 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 is employed here to search for better match sequences, which guide the evolutionary process toward solutions with better objective values. In this paper, NSGA-II is allowed to continue until the best objective measures in the archive do not improve for 50 generations, after which the non-dominated solutions are saved.

Non-dominated solutions are desirable in the sense that it is impossible to find another solution in the set which improves the value on any objective (i.e., number of buyers and sellers which match on all five objectives or match function score) without simultaneously degrading the quality of the other objective, and is formally defined as follows:

Definition 1 *(Non-dominated) Let* $o_1, o_2, ..., o_n$ *be objective functions which are to be maximized. Let* S *be the set of obtained solutions.* $s \in S$ *is dominated by* $t \in S$ *(denoted* $t \succ s$*) if* $\exists j$, $j \in \{1,...,n\}$, *such that* $o_j(t) > o_j(s)$ *and* $\forall i$, $1 \le i \le n$, $o_j(t) \ge o_i(s)$. *A non-dominated solution is therefore any solution* $s \in S$ *which is not dominated by any other* $t \in S$.

The set of all possible non-dominated solutions from the entire search space constitute the *Pareto front.*

Solutions which lie on the Pareto front represent the best trade-off between the number of total matches and match function score. Due to search space size, it is only feasible to approximate the true Pareto front for a non-trivial problem scenario. As the best permutation could be any ordering, the search space size here is 2^N, where N is the number of buyers and sellers.

4 Experiments and Results

The ability of two algorithms was compared on how effectively they matched buyers to an equal number of sellers on five objective measures with random ranges. The five objective measures or negotiation attributes were: quality, quantity, price, delivery and payment. The number of buyers and sellers were set at 20, 40, 60, 80, and 100. The number of population members in NSGA-II was set to 100, 250, 500, 1000, and 1500. For each trial of NSGA-II, the matchmaking algorithm was run using the same randomly generated data set. Ten trials at each potential combination were run, which led to a total of 250 trials (5 buyer/seller sizes times 5 population sizes times 10 trials). In addition to match function scores, a performance measurement was taken (i.e., execution time in seconds) and rate of convergence for NSGA-II was measured.

Fig. 2 shows the match score distribution for NSGA-II and the matchmaking for 20 buyers and sellers.

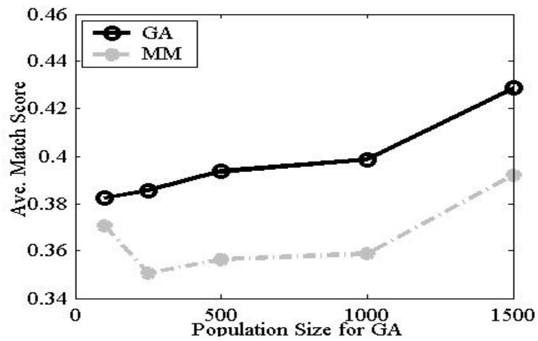

Fig. 2: NSGA-II vs. Matchmaking for 20 Buyers and Sellers

We see that with 20 buyers and sellers, NSGA-II outperformed matchmaking at all population sizes starting at 100. In Fig. 3 we see the match score distribution for 40 buyers and sellers. Here NSGA-II outperformed matchmaking starting at a population size of 500.

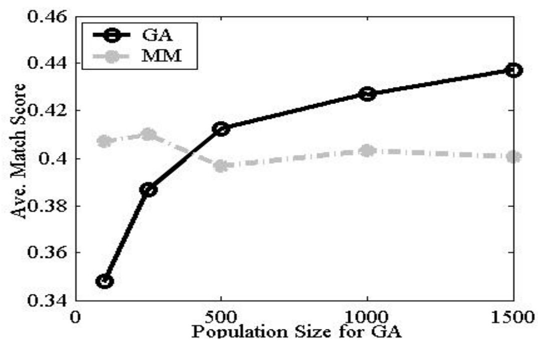

Fig. 3: NSGA-II vs. Matchmaking for 40 Buyers and Sellers

Fig. 4 shows the distribution for 60 buyers and sellers, where NSGA-II outperformed matchmaking starting at a population size of 1000. For 80 buyers and sellers, NSGA-II outperformed matchmaking at a population size of 1500 (see Fig. 5), and at population size of 2000 for 100 buyers and sellers (see Fig. 6).

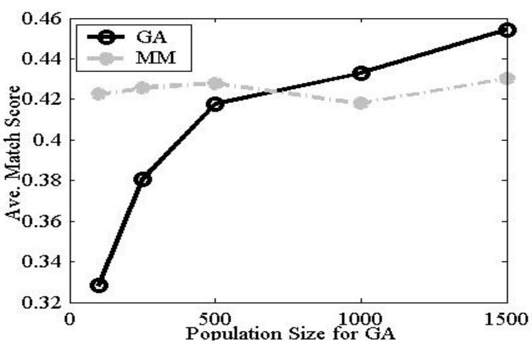

Fig. 4: NSGA-II vs. Matchmaking for 60 Buyers and Sellers

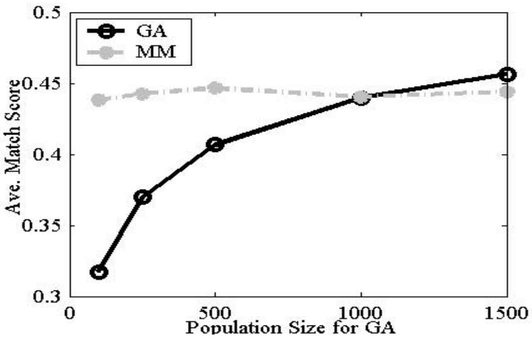

Fig. 5: NSGA-II vs. Matchmaking for 80 Buyers and Sellers

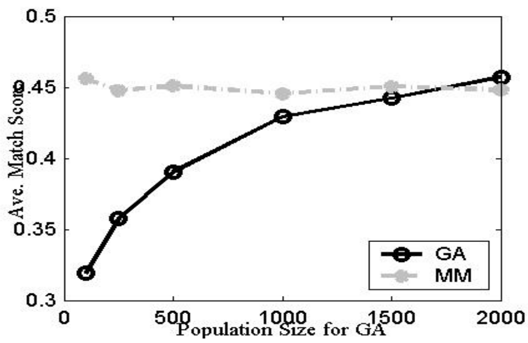

Fig. 6: NSGA-II vs. Matchmaking for 100 Buyers and Sellers

The average time to complete each trial can be seen in Table 1 for NSGA-II - in relation to the number of buyers and sellers, while the overall average time in seconds for matchmaking was 0.09 seconds. Therefore, while matchmaking is faster than NSGA-II, NSGA-II is still reasonably fast considering high solution quality is very desirable.

The rate at which NSGA-II converged was similar to the average execution time in seconds, with an average convergence in 38.58 generations with 20 buyers and sellers, 72.94 for 40, 91.48 for 60, 104.46 for 80, and 127.53 for 100. In GA terms, this is reasonably fast, and suggests the chosen representation was suitable.

We can see from the overall comparison of the two approaches that the matchmaking returns similar values repeatedly given the same number of sellers and buyers. This is expected, as the search is linear, whereas NSGA-II shows an exponential distribution, as it contends with a growing search space.

Number of Sellers and Buyers	**Population Size (Outperform point)**	**Execution Time for NSGA-II in seconds**
20	100	30.993
40	400	48.687
60	700	64.834
80	1000	76.197
100	1750	91.843

Table 1: Average Measurement Results

Because of the nature of results (i.e., they are suitable to linear regression), it is possible to accurately estimate the point at which the GA will outperform matchmaking, and how long it will take to perform the search, given a certain number of buyers and sellers.

Table 1 shows the average measurement results in numerical format, while Fig. 7 and 8 show them graphically. In particular, the relationship between the number of buyers and sellers and the population size where the GA outperforms matchmaking is provided. Fig. 7 shows this relation and a linear regression and equation for determining additional scenarios. For example, if there were 200 buyers and sellers, the GA can be expected to outperform matchmaking given a population size of 3,520.

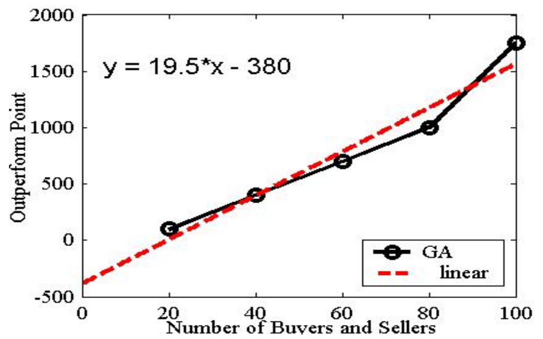

Fig. 7: Outperform Point vs. Number of Buyers and Sellers

In Table 1 the execution time in comparison with the number of sellers and buyers is also shown. Fig. 8

shows this relation and a linear regression and equation for determining additional scenarios. For example, if there were 200 buyers and sellers, the GA can be expected to outperform matchmaking in 168 seconds.

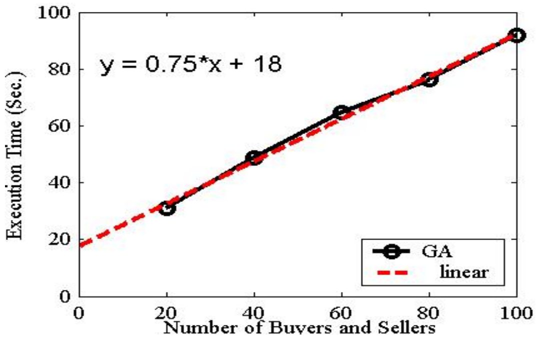

Fig. 8: Execution Time vs. Number of Buyer and Sellers

5 Conclusion

This paper investigated multilateral and integrative negotiations using two approaches to find the best matches for a single product depending on a variable but equal number of buyers and sellers. The two approaches chosen were matchmaking and a GA: NSGA-II. The measurements revealed that NSGA-II outperforms matchmaking when a suitable population size is used. Two equations were derived from the results which allow good estimations of both the size of the population to use and the average speed of execution. In general, larger number of buyers and sellers means greater population sizes and more time for execution. In conclusion, considering matchmaking is commonly used in e-negotiation situations, evidence here suggests that if high solution quality is paramount, GAs can do a better job in a reasonable amount of time.

References:

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

[2] R.H. Guttman and P. Maes. Cooperative vs. Competitive Multi-Agent Negotiations in Retail Electronic Commerce. *Cooperative Information Agents II, Springer-Verlag*, July 1998.

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

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

[5] T. Jelassi and A. Foroughi. Negotiation support systems: an overview of design issues and existing software. *Decision Support Systems* 5: pp. 167-181, June 1989.

[6] H. Raiffa. *The art and science of negotiation*. Harvard University Press 1982.

[7] M. Ströbel. Effects of Electronic Markets on Negotiation Processes. Proceedings of the ECIS, Vienna, Austria, 2000.

[8] M. Mitchell. An Introduction to Genetic Algorithms (Complex Adaptive Systems), The MIT Press, ISBN 0-262-63185-7, 1998.

[9] T. Starkweather, S. McDaniel, K. Mathias, D. Whitley and C. Whitley. A comparison of genetic sequencing operators. In Rick Belew and Lashon Booker, editors, Morgan Kaufman. Proceedings of the Fourth International Conference on Genetic Algorithms, pages 69-76, San Mateo, CA, 1991.

[10] K. Deb. Multi-objective optimization using evolutionary algorithms. Wiley, Chichester, England, 2001.

[11] K. Deb, S. Agrawal, A. Pratap and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In Lecture Notes in Computer Science, volume 1917, pages 848-849, 2000.