Time-Variant Distributed Agent Matching Applications*

David Sarne[1] and Sarit Kraus[1,2]
[1]Department of Computer Science, Bar-Ilan University, Ramat-Gan, 52900 Israel
[2]Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742
{sarned, sarit}@macs.biu.ac.il

Abstract

The process of pair partnership formation is an important infrastructure for many plausible MAS applications. Each agent evaluates potential partner agents, where each potential match yields a different utility. Commonly, the utility associated with a given agent partner in such two-sided search processes may change over time. This change in the agent's future attractiveness to potential partners significantly increases the complexity of the agent's decision making process regarding the set of agents it is willing to partner with. In this paper we analyze the special dynamics and present equilibrium characteristics of such a model. The agents can gain a utility derived from the partner agent's type. However, as an agent has an incentive to extend its search for a better type partner, the benefit that can be offered to potential partners reduces as the search proceeds. We introduce a two-sided model which takes into consideration a continuous decrease in the agent's type and formulate the appropriate equilibrium equations. The suggested equilibrium analysis yields an algorithm for the calculation of the agents' equilibrium strategy. Special emphasis is placed on the scenario where an agent's attractiveness is influenced by an additional dimension other than just time. Simulation results are presented to illustrate the findings.

1. Introduction

Pair partnership formation is an important variant of the general coalition formation process. The main incentive for this process is similar to the one which drives the coalition formation: throughout a coalition, the agents (as a group) can operate more effectively and coordinate their activities [12], thus increasing the participants' benefits [2]. However in the pair partnership formation model, the agent is satisfied with only one partner for forming a coalition.

Typical applications that make use of size-two partnership formation process include buyer-seller, peer-to-peer media exchange, dual long distance call partnering termination-services, etc. In these types of applications the agent can gain utility only if it eventually partners with another agent. However, once a coalition is formed, adding additional agents does not produce any additional benefit.

As in the general coalition formation model, the key issue for the partnering process is for each agent to determine the set of agents it is willing to form a partnership with. This is where each agent is associated with a specific type that captures special characterizing properties. Two important factors significantly increase the complexity of the model. The first factor is the search costs (associated with the resources the agent has to spend as part of its search for partners - communication, CPU, memory, etc.), induced at each search stage [11]. The second is whether or not the agents types are constant (stationary) or changing over time (non stationary). The latter attribute, which will be discussed in this paper, is actually application-dependant. For example, while in a buyer-seller application, the seller's preferences may remain constant over time[1], the buyer agent's preferences may change as it reaches its purchase deadline. The main contribution of this paper is the formalization, analysis, and algorithm for finding the equilibrium strategies for partnership formation in a distributed environment where the agent type is: (a) changing over time, and (b) comprised of more than one attribute. Even though most partnership applications are associated with type changes, to the best of our knowledge no such model has been developed to date.

As a framework for our analysis, we introduce and utilize the dual-backup application, though the suggested model is general and applicable for any domain in which the agents types weaken constantly over time. The dual-backup application is a peer-to-peer based backup service. Here, agents, representing different servers search for similar agents to form a partnership for a mutual backup purpose.

Recall that for most corporations and small businesses, the general definition of "proper" backups requires redundancy. One must keep multiple copies of the same files at different locations. This is why backups are sent (either through the internet, intranet or any other secure link) to a remote server (either offsite or within the organization), and stored safely away from the clients' computers. Whether the backup server belongs to a service provider or is operated by the organization itself, the billing is mainly calculated based on actual usage[2]. An alternative for the above tradi-

* This research was supported in part by NSF under grant #IIS-0208608.

1 The seller's preferences may also change over time, i.e. when the seller considers interest over equity.

2 Usually around $50 per month per gig - see for example http://www.persontech.com/

tional costly backup method is the distributed dual backup application. Many intra-organizational servers occasionally have free resources (mostly disk space). Though these resources are not compatible for self backup (the source and backup files shouldn't reside on the same machine), they can be used for backing up other servers within the organization network and vice versa. Similar applications can be used for offsite dual-backup between organizations or different branches, assuming security issues can be resolved. Any volume of backup that can be obtained, in such a manner, can be deducted from the total backup volume directed to the external backup provider or the internal backup server.

Since the volume of free resources of each server largely varies over time, any server can supply only a short-term forecast of its availability for these dual-backup partnerships. Therefore, periodically, the server creates a new agent, equipped with its short term commitment details (file size and due date), offering its backup services from the current time until a pre-defined due-date. In exchange for the offered resources, the agent will try to obtain similar backup services from the agents representing the other servers. An exchange is considered rather than a direct sell, as selling the backup service is not supported by backoffice infrastructure.

A centralized partnerships allocation would require a central planner (broker) and complete visibility of available agents. It would also require a solution which would prevent a conflict of interests. However, in the absence of any of the above requirements (for example, when agents represent servers from different corporations or divisions) a distributed model should be considered.

In our model, we consider the utility an agent can gain out of a partnership with any given potential partner as the product of the file size (disk space) offered by the partner and the time left until the agent's due date[3]. An agent's type at any given time is determined by these two parameters. We envision an environment with numerous agents of different types. Each agent sequentially locates and interacts with other agents to form partnerships as described above. After reviewing the information regarding the current potential partner, the agent must make a decision whether to form a coalition with this partner or to keep searching for other potential partners. Obviously continuing the search will eventually yield a higher type partner. However, search is timely and the more selective the agent is, it will take it more time to find an appropriate partner. As the agent extends the search period, the time interval it can commit to (until its deadline) shortens, resulting with a decrease in its own type (and thus a decrease in its future attractiveness to other agents).

The context of this paper does not deal with the process of locating other agents (though search costs can be easily integrated into the model), but rather with finding the mechanisms used by the agent to determine which of the other agents it is willing to form a partnership with. The main challenge for each agent is to determine how its strategy will affect the other agent types' strategies towards equilibrium, considering the fact that the agent's own type changes over time. In the next section we address relevant multi-agent models and two-sided matching literature. In these models, search behavior is influenced mainly by explicit search costs rather than type change considerations. In section 3 we present the model and analyze the agent's equilibrium strategy. The influence of several environmental factors on the equilibrium strategies, carried out throughout simulations, as well as a comparison with the model of stationary agent types are given in section 4. We conclude and present directions for future work in section 5.

2. Related Work

The use of agents for matchmaking is not new [6]. Service matchmaking and brokering have been referred to in several systems and applications [15, 7]. Most of the proposed applications have been in the buyers and sellers domain [2]. Extended coalition formation models evolved as a group buying mechanism based on agent-mediated electronic commerce [5] and for large-scale electronic markets [8]. While some mechanisms assume that an agent can scan as many agents as needed, others use a central matcher or middle agents [3]. Few, have considered the problem of finding matches for cooperative tasks without the help of a predefined organization or central facilitator [10, 11]. However, to the best of our knowledge, a model in which agents types change over time (and the influence of this factor on the equilibrium strategies) has not been studied.

The concept of matching is an important function of many domains and processes, thus relevant research can also be found in economic and social studies (e.g. applications of students and colleges, workers and firms, marriageable men and women, etc.). A legacy matching application, was introduced by Gale and Shapley [4], emphasizing the stability of matching. Nevertheless, the model (as well as the rich literature that followed) assumes a relatively "free" search process, e.g. all parties involved have complete knowledge about the available options to choose among, allowing each searcher to consider all opportunities. More relevant to our model, is the "assortative matching" literature, which suggests models where preferences and options are at least partially unknown, and the search process is usually associated with costs. These models involved a decentralized search of more than two heterogeneous agent types, assuming an agent's utility is mainly derived from its partner's type. Such costless model was first introduced by Becker [1], and later on extended to include some search cost elements [13, 14]. Nonetheless, all the above mentioned models assume the searcher's type, in terms of the utility it can offer potential partners, remains constant over time. In these models the

3 This utility can also be expressed in monetary values, since this is the amount of backup volume the corporations can deduct from the alternative costly remote backup usage. However, if the agent's type is a function of more than one dimension, then monetary values are not applicable for describing the agent's type.

only incentives to limit search activity were the actual search costs incurred or the discounting of future revenues.

Variants of the model, incorporating strategy change over time (due to finite decision horizon [9] and environmental changes [16]) can be found in one-sided search applications. Here, the initial focus is on establishing optimal strategies for the searcher while assuming no mutual search activities. These models lack the equilibrium concept of the two-sided search and thus, are not compatible to our problem.

The transition from stationary to non-stationary agent types models adds several constraints that needs to be considered, as will be demonstrated in the next section. Furthermore, while economic models are more concerned with describing the equilibrium equations, MAS applications require also computational means (algorithms) for deriving the agent's policy for different settings.

3. The Model

Consider an environment with numerous heterogeneous agents, where each agent represents a server interested in a remote backup service. At any time t, each agent is characterized by a type, defined by the pair $(q, l(t))$, reflecting the volume of backup it is committed to supply. Here, q is the offered file size and $l(t)$ is a measure of availability, expressed as the time length left for this offer (the total time the server can commit for q, assuming the process will start at current time). The population of agent types is associated with a p.d.f. $f(q, l)$ and c.d.f. $F(q, l)$.

Each agent randomly becomes acquainted with other potential agent partners in a sequential process with an inter-arrival time, denoted by $\triangle$. Thus, if agent $(q, l(t))$ has an unsuccessful interaction (where no partnership is formed) at time t, its next interaction with another random potential partner agent will take place at time $t + \triangle$, however, in the latter interaction the agent's type will be reduced to $(q, l(t)-\triangle)$. Thus, the agent's type changes (decreases) constantly over time, and its decision horizon is limited - once the agent has exceeded its due date, it can't offer any backup services to other agents, and thus it ceases to exist. We assume that agents, while ignorant of other individual agents' types are acquainted with the overall distribution of agent types in the environment[4] and that this distribution remains constant over time (widely common assumptions in search models - see for example [13, 14]). Similarly, the interaction with other agents doesn't imply any new information about the environment structure.

A similar scenario can be found in other plausible applications. Consider, for example, a service provider that can produce a short term forecast for its unused bandwidth. The service provider can create an agent that will search for similar agents to exchange this bandwidth. Each potential partner agent will be characterized by the bandwidth and the due-date for which it can commit. Once a partnership is formed, each service provider will route some of its traffic to the remote destinations via the other service provider's network, instead of using its costly formal termination partners. Alternatively, consider content companies which pay for mirroring their content - these companies can reduce the amount of mirroring service consumed, by exchanging their idle servers resources with other different geographically located content companies for mirroring purposes.

4 There are several methods by which an agent can be acquainted with this distribution function: past experience (assuming the server frequently executes such agent), bayesian update through sampling, etc.

The limited decision horizon, and the constant change of type, suggests that the agent's strategy is non-stationary - an agent will be willing to accept different sets of agent types at different time points along its life cycle. An opportunity rejected in the past can't be recalled, as other agents change their type as well. At the encounter, both agents will reveal their current updated type. Then, each agent will make a decision whether to continue searching or to form a partnership with the current encountered agent. In the latter case, the new partnership will take effect only if both agents are willing to form it. Upon the creation of a new partnership, both agents will exchange technical parameters (servers' IP addresses, etc.) and each represented server will be able to use the other represented server's resources, up to a capacity of q for an l length of time. The agents are self-interested and therefore, given several alternatives, they will prefer to select the more beneficial ones. Thus the agent's strategy is reservation value based (following a reservation-value rule: accepting all offers greater than or equal to the reservation value, and rejecting all those less than this value), with a changing reservation value over time. Recall that the agent's utility is derived from the total backup volume offered, which is the product of the disk size and time length (the amount reduced from the alternative costly remote service). Thus the agent's reservation value should be related to this product. Namely, if the agent is willing to accept a specific agent of type (q_i, l_j), it will also be willing to accept any other agent of type (q_m, l_n) where $q_m l_n \geq q_i l_j$.

3.1. Agent's Strategy

As in most real-life applications, we expect agent types to be discrete. Thus the agents population can be described using the probability function $P(q, l)$. The variable l will be used to represent the number of time units of length $\triangle$, left at time t until the agent exceeds its expiration due-date ($l(t) = 1, ..., n$, where n is the highest type agent, and the product n$\triangle$ can be considered the longest forecast made by any of the servers). Since $\triangle$ is a parameter of the environment, the agent's decision at time point t of its search is exclusively a function of its type at this time $(q, l(t))$.

Consider an agent of type (q_i, l_i) at time t, using a reservation value x_i. The expected utility of this agent, denoted $V_{q_i,l_i}(x_i)$, can be described using the following equation:

$$\begin{aligned} V_{q_i,l_i}(x_i) = & E[q_j l_j \bullet 1[(q_i l_i \geq x_j) \cap ((q_j l_j \geq x_i))] + \\ & + V_{q_i,l_{i-1}} \bullet 1[(q_i l_i < x_j) \cup ((q_j l_j < x_i)]] \end{aligned} \quad (1)$$

Here $\bullet 1[(q_il_i \geq x_j) \cap ((q_jl_j \geq x_i))]$ represents the indicator of the event $(q_il_i \geq x_j) \cap ((q_jl_j \geq x_i))$.

Theorem 1 *Given the other agents' reservation values, the optimal strategy of type (q, l_i) agent is a reservation value x_i which equals the utility of agent type (q, l_{i-1}). Formally stated: $x_i = V_{q,l_{i-1}}(x_{i-1})$.*

Proof: Consider any agent at a given stage of its search, where its type is (q_i, l_i). After reviewing the current potential partner's type (q_j, l_j), the agent has two alternatives: to continue the search, resulting in an expected future total utility of $V_{q_i,l_{i-1}}$ or to accept the partnership. The latter option will result in a utility of q_jl_j, if the other agent agrees to form a coalition, otherwise the agent will have to keep searching with an expected future total utility of $V_{q_i,l_{i-1}}$. Therefore, the optimal reservation value is $x_i = q_jl_j$, where the agent is indifferent to both options:

$$V_{q_i,l_{i-1}} = x_i \bullet 1[q_il_i \geq x_j] + V_{q_i,l_{i-1}} \bullet 1[q_il_i < x_j] \quad (2)$$

Resulting with: $x_i = V_{q_i,l_{i-1}}$. □

Figure 1 illustrates this scenario. The inter arrival time is 10, though assuming an agent of type l_3 or l_2 didn't find a match, its next search stage will be as type l_2 and l_1 accordingly. The equilibrium reservation values and the obtained utilities are given in the figure. None of the agents have an incentive to deviate from this strategy.

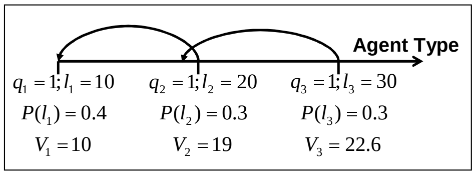

Figure 1: Simple equilibrium example

Notice that if the agents are subject to a search cost c (in terms of the resources associated with finding a potential partner agent and communication) then, the equation in Theorem 1 simply transforms into: $x_i = V_{q,l_{i-1}}(x_{i-1}) - c$, where the validity of the following analysis remains unchanged. For simplification, we'll proceed with the analysis assuming the agents are not subject to any search costs.

The agent types can be described over the bi-dimensional grid (see Figure (2)), where the horizontal axis represents the time interval l, and the vertical axis is the file size q. Each agent type (q_i, l_i) uses its own reservation value, x_i, accepting all other agents of type (q, l) satisfying $ql \geq x_i$.

Notice that even though the agent's attractiveness is the product q_il_i, two agents, offering the same product, may be using two different reservation values. This is simply because the agent's attractiveness may decrease in different rates (a parameter of q_i).

Theorem 2 *For any given q, the agents equilibrium reservation value increases with the agent's type. Formally stated: Given the two agents (q, l_i) and (q, l_j), where $l_i \geq l_j$ then the following holds: $x_i \geq x_j$*

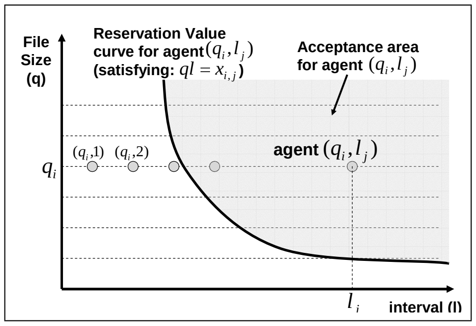
File Size (q)
Reservation Value curve for agent (q_i, l_j) (satisfying: $ql = x_{i,j}$)
Acceptance area for agent (q_i, l_j)
$(q_i,1)$
$(q_i,2)$
agent (q_i, l_j)
q_i
l_i
interval (l)

Figure 2: Agent type representation over a grid

Proof: Consider an agent of type (q, l_i). Assuming all other agents behave according to the theorem, then the utility of this agent can be written as:

$$V_{q_i,l_i}(x_i) = \sum_{j=1}^{n} \sum_{l=\frac{x_i}{q_j}}^{L_j(q_i)} q_jlP(q_j,l) + \quad (3)$$

$$+ V_{q_i,l_{i-1}}(1 - \sum_{j=1}^{n} \sum_{t=\frac{x_i}{q_j}}^{L_j(q_i)} P(q_j,l))$$

This is where $L_j(q_i) = max(l_k | x_{l_k} \leq q_il_i)$, the maximal time interval offered by any of the agent types (q_j, l), accepting agent type (q_i, l_i). We will prove that agent (q_i, l_i)'s strategy is to act according to the theorem. Utilizing Theorem 1, the following equation should hold:

$$x_i = \sum_{j=1}^{n} \sum_{l=\frac{x_{i-1}}{q_j}}^{L_j(q_{i-1})} q_jlP(q_j,l) + x_{i-1}(1 - \sum_{j=1}^{n} \sum_{l=\frac{x_{i-1}}{q_j}}^{L_j(q_{i-1})} P(q_j,l)) \quad (4)$$

Rearranging the equation, we obtain:

$$x_i - x_{i-1} = \sum_{j=1}^{n} \sum_{l=\frac{x_{i-1}}{q_j}}^{L_j(q_{i-1})} (q_jl - x_{i-1})P(q_j,l) \quad (5)$$

The right hand-side of the above equation is always positive, thus $x_i \geq x_{i-1}$.□

3.2. Equilibrium Analysis and Algorithm

For the purpose of simplification of the equilibrium discussion and notation of the algorithm that follows, a simpler scenario will be used. We'll consider each agent to be responsible for a fixed disk space size (for example 1Gb), for a variable amount of time (differing in the value of l). Thus, the agent's type can be exclusively represented by $l(t)$. At the end of the section we show that the suggested algorithm can be applicable in the general case as well. Equation (3) can be now formulated as:

$$V_{l_i}(x_i) = \sum_{l=x_i}^{L_i} lP(l) + V_{l_i-1}(1 - \sum_{l=x_i}^{L_i} P(l)) \quad (6)$$

$$= \sum_{y=1}^{i} \Big(\sum_{k=x_y}^{L_y} kP(k) \prod_{k=y+1}^{i} (1 - \sum_{j=l_k}^{L_k} P(j)) \Big)$$

where $l_i = 1, ..., n$ and $L_i = max(l_k | x_{l_k} \leq l_i)$.

We'll start by stating that an equilibrium will not necessarily exist, when there is a type change. A simple example for demonstrating this is given in figure (3) below:

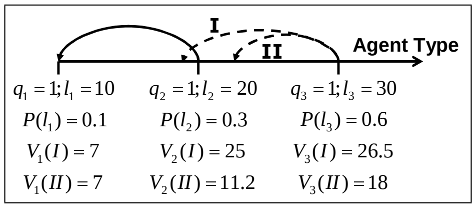

Figure 3: No equilibrium can be found

If agents of type l_3 will not accept agents of type l_2, then the maximum utility for agents of type l_2 is necessarily smaller than 20. However, for an individual agent of type l_3 it is beneficial to accept agent l_2 (utility of 20) rather than become this agent's type at the next search stage (expected utility of 11.2). Similarly, if all agents of type l_3 will accept agents of type l_2 then an individual agent of type l_3 can benefit by rejecting an agent of type l_2, hoping to partner with an agent of type l_3 at the next search stage. An equilibrium can be assured only if the agents types are defined over a continuous interval with a positive distribution function[5]. However in most domains types are discrete and each agent type has an actual probability, rather than a distribution function. Thus, any reservation value x_i within the continuous interval $[l_j, l_{j+1}]$ will yield agent l_i a similar utility as if using other values in this interval as a reservation value.

Thus, the equation $x_i = V_{q,l_i-1}(x_{i-1})$ as stated in Theorem 1 becomes a condition for equilibrium, when the types are discrete, formulated as:

$$\lfloor V_{q,l_i-1}(x_{i-1}) \rfloor \leq x_i \leq \lceil V_{q,l_i-1}(x_{i-1}) \rceil \qquad (7)$$

Obviously, a pure equilibrium for this problem (if exists) can be identified by solving a set of n complex equations, of the type similar to Equation (6), under the set of $n-1$ constraints of the type similar to Equation (7). Since each equation contains the reservation values as the sums upper bound, the solution method will involve solving n^n combinations of n equation sets, which can be reduced to $(n-1)!$ equations utilizing Theorem 2. However considering Theorem 1, a simple and efficient algorithm can be suggested for the task of calculating the equilibrium strategies.

Algorithm 1 *An algorithm for calculating the equilibrium reservation values $(x_1, ..., x_n)$.*
Input: $P[1:n]$ *- Vector of probabilities* $(\sum_{i=1}^{n} P[i] = 1)$.
$V[1:n]$ *- Vector of types.*
Output: $X[1:n]$ *- Vector of reservation values.*

```
01 init: done=false; i=n; U[]=V[];
     for (j=1;j ≤ n-1;j++) X[j+1]=j;
02 While $not((i == 1)\&\&(done))$ do {
03   PrevUtil=U[i]; U[i]=min(CalcUtil(i),V[i]);
04   if (i<n)&&($PrevUtil > U[i]$) then {
05     done=false;
06     if (X[i+1]>1)&&((V[X[i+1]]-$\lfloor U[i] \rfloor$)>$\triangle$) then {
07       X[i+1]−−;
08       for (k=2;k<=i;k++) {
09         X[k]=min(X[i+1],k-1);
10         if (k<X[i+1]) then U[k]=V[k];
11         else U[i]=CalcUtil(i);
       } }
12     i++;
     }
10   else $i--$;
11   if (i==n) then done=true;
   }
12 return(X[]);
```

The function *CalcUtil(i)* in the algorithm calculates $\sum_{j=X[i]}^{max(k|X[k]\leq i)} jP[j] + U[j-1](1 - \sum_{j=X[i]}^{max(k|X[k]\leq i)} P[j])$. Once the algorithm has terminated, the vector $X[]$ will hold a set of reservation values. If Equation (7) holds for these values (which can be easily checked) then this is the equilibrium strategy. Otherwise, the problem doesn't have an equilibrium.

The algorithm is based on a continuous update of the agent's estimation for the upper bound value of the utility that can be gained by becoming the next best type agent (the upper bound of the expected future utility if the current interaction will not yield a partnership). The vector U[] is used to keep track of the upper bound for the agent's utility, where U[i] is the current upper bound for the utility agent l_i can gain, given the current set X[i],...,X[n]. The initial utility's upper bound for each agent is set as its own type and from this point on, it can only decrease as the algorithm proceeds. Each agent i's initial reservation value is set as type l_{i-1} agent.

The algorithm will stop, once it has completed a full pass over all agent types, top down, and verified that the chain of utilities and reservation values is valid, in terms of the internal constraints that must hold. At each stage the current agent's utility upper bound is recalculated for type l_i and compared with the previous value. If the new value reflects a decrease (rather than stagnation), then agent type l_i's utility should be updated and so the variable i will be increased. Otherwise, the current utility for agent l_i can be supported, and the index i can be decreased. A reservation value change (decrease) will take place if the left-hand-side inequality of (7) does not hold for the current agent l_i. In this case, the agent's reservation value will be decreased to x[i]-1, and all the agents having a lower type will be initialized as before.

The following Lemma will be used to prove the correctness of the algorithm.

5 With the continuous type function, we can always maintain the equation $x_i = V_{q,l_{i-1}}(x_{i-1})$ by slightly increasing/decreasing an agent's reservation value.

Lemma 1 *In equilibrium (a) an agent's utility will never exceed its own type. Formally stated: $V_{l_i} < l_i$. (b) an agent's reservation value will always be smaller or equal to the next best agent type. Formally stated: $x_i \leq l_{i-1}$. (c) a decrease in an agent's reservation value, can only improve the expected utility related to the next best type agent. Formally stated:* $\frac{dV_{l_{i-1}}}{dx_i} \leq 0$.

Proof:
(a) Assume there is an agent of type l_i, for which $V_{l_i} \geq l_i$. Now, consider agent l_{i+1} evaluating agent l_i as a potential partner. In this scenario, agent l_{i+1}'s best strategy is to use $x_{i+1} > l_i$, and according to Theorem 2, the rest of the higher type agents will reject agent l_i as well. Being accepted only by lower type agents, agent l_i's maximum possible utility is now smaller than l_i which contradicts the assumption of the proof.
(b) Consider an agent of type l_i evaluating an agent of type l_{i-1}. Rejecting agent l_{i-1} will bring agent l_i to the next search stage as l_{i-1}. According to part (a) of the Lemma, the maximum utility that can be obtained by this type is smaller than l_{i-1}. Thus type l_i should have initially accepted type l_{i-1}.
(c) Sketch of proof: Consider an agent of type l_j which was formerly rejected by agents of type l_i, and using the new reservation value of l_i it is now accepted by this type. Obviously this agent type has improved its utility, and the increase will affect all other agents with a better type up to l_{i-1}. □

Theorem 3 *(a) Algorithm 1 will always terminate in finite time. (b) Upon termination of the algorithm, if X[] satisfies (7) then the problem has a pure equilibrium, specified by X[]. (c) Otherwise, there is no pure equilibrium for this environment settings.*

Proof: Notice that in the initial state of the algorithm (stage 01), each agent i is equipped with an upper bound (according to Lemma 1) for its equilibrium reservation value $X[i]$ and an upper bound measure for the utility that it can gain $U[i]$. Stages 02-11 keep updating these upper bounds, based on an optimistic evaluation of the highest lower type agent ($i - 1$). Thus, a decrease in the latter agent's upper bound evaluation, will unavoidably decrease the current agent i's upper bound as well. Whenever such a utility decrease violates Equation (7), it is obvious that no equilibrium can be maintained within the current subset values X[i,...,n]. Therefore the reservation value of agent i must be decreased further. In this case, all lower type agents $1, ..., i$ initialize their optimistic utility upper bound, which equals their own value type (steps 7-11). In any case where U[i] was decreased, the algorithm draws back to the former agent $i + 1$, in order to check if a further decrease is required in reservation values. Otherwise it can proceed towards the lowest type agent. Since a change in the agent's reservation value and utility can only cause a decrease in other upper agent types' reservation values and utilities, then eventually the process will end leaving every agent i's reservation value smaller than the expected utility of agent $i - 1$. At the end of the process, vector U[] needs to be re-calculated one more time using Equation (4). If the problem has a pure equilibrium strategy, then Equation (7) should hold for every X[i]. Otherwise, the only way to maintain Equation (7) is by increasing one of the upper types agents' reservation values. However, this can not be done because each agent's utility as stored in U[] at the end of the algorithm execution, is an upper bound for the equilibrium utilities.□

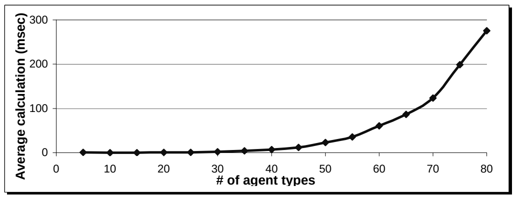

Figure 4: Average algorithm completion time (miliseconds)

The performance of the algorithm is highly correlated with the distribution function being used. Theoretically (worst-case scenario) the algorithm will need to perform (n-1)! type changes, each resulting from an average of n/2 simple utility update calculations. Practically, we found that the mechanisms used by the algorithm in order to eliminate non-relevant reservation values vector structures, suggests quite an impressive performance. The following graph presented in Figure (4) illustrates the algorithm performance as obtained in the simulations, for different numbers of agent types. Each point in the graph represents the average time it took to complete the algorithm execution. Each average was calculated over 10,000 different environments with randomly drawn agent type probability functions. The simulation was implemented using Java, running on a 2.4Ghz Windows based PC.

One may observe, that even for a relatively large number of agent types, each agent is able to calculate the equilibrium strategy in quite a short time. Moreover, since equilibrium is concerned, and the agent is familiar with the distribution function of agent types, the calculation can be performed offline. The results presented in Figure 4, demonstrate significant improvement in comparison to the method of solving the proposed complex equilibrium equation set.

Lastly, we would like to emphasize that the suggested algorithm can be adjusted to the general case where the agent's type is a function of two parameters (q_i, l_i). The bi-dimensional problem can be represented by a vector V[], where the agents are sorted according to the product $q_i l_i$ (if two agents will have the same product, they will be ordered according to the value of q). Initially, each agent (q_i, l_i)'s reservation value will be set to $x_{q_i,l_i} = q_i l_{i-1}$ and the upper bound utility will be calculated in the same manner suggested in the algorithm, also taking into consideration the other agent types above the reservation value.

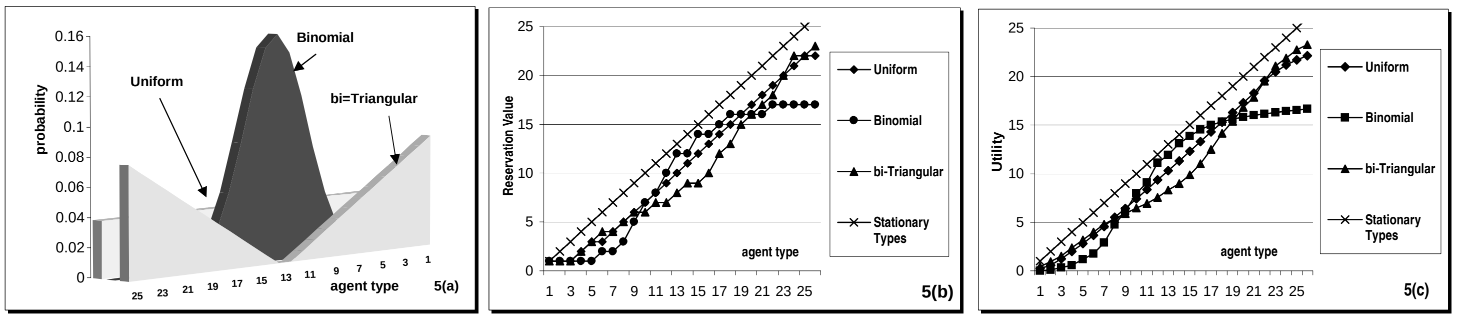

Figure 5: Equilibrium strategies by probability functions comparison

A reduction in the value of any member of the reservation values vector X[], will be to the highest lower type in the vector V[], having the next best value of the product ql. The appropriate algorithm for the general case, as well as an extended version of the paper can be found at http://www.cs.biu.ac.il/ sarit/Articles/TimeVariantFull.pdf.

3.3. Comparison with the stationary model

If the agent types would have remained constant over time, then we wouldn't have needed the bi-dimensional grid representation, and the agent's type could have been considered simply as the product ql. For the stationary agent type model with no search costs, Becker [1] showed that matched partners are identical (in type). If search costs are integrated into the stationary agent types model, a perfect segregation equilibrium will hold [14]: An interval of the 'highest' types agents join together only with each other, then the agents in the next highest type interval join together only with each other, and so on; there is no intermingling. The reason is that individuals in each class have the same opportunity set (namely, those willing to match with them), and thus make the same matching decisions. Therefore, for the stationary agent type model with search costs, the agent's problem would have become a simple problem of choosing the best reservation value, given the reservation value of upper types agents and assuming all agents of lower types will accept it. This process as described in [11] would have required n calculations.

However, given our non-stationary agent type characteristic, an agent must also consider the reflected change in the lower agents types' strategy, as part of its calculations[6]. Thus equilibrium strategy is distinguished from the above described stationary type equilibriums.

4. Equilibrium in different environments

For the purpose of demonstrating the dynamics of changes in the agent's policy, and the equilibrium calculation we used a simulation environment with varying numbers of agent types[7] and three different agent type probability functions (uniform, binomial and bi-triangular). The probability functions' parameters were all set to support an equal mean of $(\frac{l_1+l_n}{2})$.

Figure 5 uses 26 agent types to demonstrate the different affects of agent type probability functions (Figure 5-a) on the equilibrium strategy (Figure 5-b) and the perceived utility (Figure 5-c). The highest curves in Figures 5-b and 5-c are the theoretical reservation values and the perceived utilities that can be gained if the agent types remain unchanged. The three lower curves are related to our model, where the agent type decreases along time. One may see that the environment with a bi-triangular function produces the highest utilities for upper and lower type agents (in comparison to the center type agents). The same can be said for the reservation values. The intuitive explanation is that these agents have a good acceptance probability even as their type decreases, and thus can remain selective for a substantial time. The environment with the binomial probability function projects the opposite behavior, that can be explained in a similar manner. Notice however, that the center type agents' expected utility, when the agent types are characterized by a binomial probability function, is very close to the theoretical utility these agents could have obtained if their types remained stable. This is mainly because of the high probability for encountering a similar type agent. As expected, the utility of agents when uniformly distributed, is bounded by the utility of these agents in the environments of the other two probability functions.

Figure 6, demonstrates the obtained utility and the equilibrium reservation values for similar agents in different environments characterized by different granularity of agent types (different values of $\triangle$ - the inter-arrival time of potential partner agents). The agent's utility increases as the granularity increases. This can be explained as follows: since the agent has more opportunities to scan potential partnering agents, without a significant decrease in its type, it can become more selective and improve its expected utility.

6 This is simply because a possible future world state for this agent is to become of a lower type.

7 Agent types were set as $l_i = 1 + i\triangle$, $i = 1, ..., n$, where we had control over the value of n and $\triangle$.

Lastly, in Figure 7 we compare the equilibrium strategy and the perceived utilities in our model, with the known results for the stationary agent types model (with and without search costs). As suggested in section 3, the stationary model suggests identical types matching or perfect segregation equilibrium (depending on implementation of search costs). This is illustrated by the upper and middle curves (using search cost $c = 0.5$ for middle curve). In our model, as proved in the analysis given in section 3, the agents use monotonic decreasing matching sets (illustrated by the bottom curve).

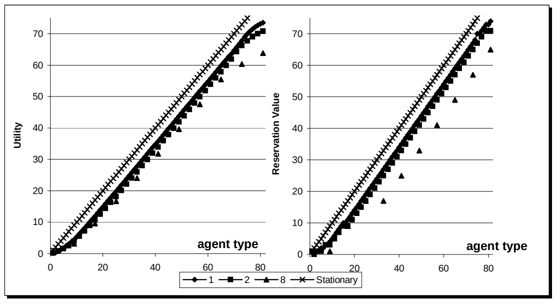

Figure 6: Effect of type granularity ($\triangle$)

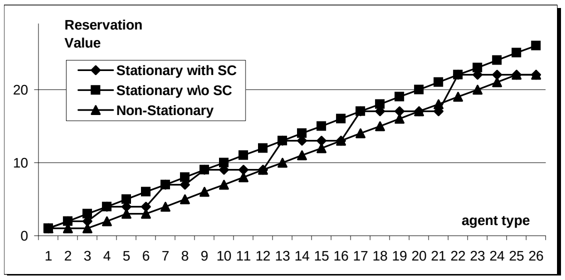

Figure 7: Stationary and non-stationary agent types

5. Conclusions

The non-stationary characteristic of agents types adds many complexities to the analysis of MAS two-sided search applications. Unlike stationary agent type models, where the agent's strategy is affected only by the distribution of higher type agents, a similar change in the non-stationary agent types model will affect all other agent types and consequently also affect the agent's future strategies. In this paper we introduce the dual-backup application which utilize this model, and analyze the appropriate equilibrium strategies.

Though the model was derived from the dual-backup application, it is appropriate for many other applications where the agent gains a utility only by strictly partnering with one other agent. The suggested analysis is also valid for the case where agent types are a function of an additional dimension, as well as for the case where search costs are integrated along the process. We show that generally, the equilibrium strategy structure differs from the one obtained in stationary agent types model. Specifically, the equilibrium shifts from identical types matching (for the stationary model with no search cost) or perfect segregation (for the stationary model with search costs) to monotonic decreasing matching sets.

An algorithm for identifying the agents equilibrium strategies is suggested, based on the comprehensive analysis. Empirically we show that the algorithm can efficiently determine the equilibrium strategies. We plan to extend the research to further improve the calculation algorithm, either by using heuristics or for specific probability functions. Throughout simulation, we manage to demonstrate a substantial affect of agent types probability functions and a moderate affect of the granularity of search intervals on the equilibrium strategies and the perceived utilities.

References

[1] G. Becker. A theory of marriage. *Journal of Political Economy*, 81:813–846, 1973.

[2] S. Breban and J. Vassileva. Long-term coalitions for the electronic marketplace. In *B. Spencer, ed., Proc. of E-Commerce Applications Workshop, Canadian AI Conference*, 2001.

[3] K. Decker, K. Sycara, and M. Williamson. Middle-agents for the internet. In *Proc. of IJCAI*, 1997.

[4] D. Gale and L. Shapley. College admissions and the stability of marriage. *American Math. Monthly*, 69:9–15, 1962.

[5] T. Ito, H. Ochi, and T. Shintani. A group-buy protocol based on coalition formation for agent-mediated e-commerce. *IJCIS*, 3(1):11–20, 2002.

[6] M. Klusch. Agent-mediated trading: Intelligent agents and e-business. *J. on Data and Knowledge Engineering*, 36(3), 2001.

[7] A. Kraft, S. Pitsch, and M. Vetter. Agent-driven online business in virtual communities. In *Proc. of HICSS-33*, Maui, Hawaii, January 2000.

[8] K. Lermann and O. Shehory. Coalition formation for large scale electronic markets. In *Proc. of ICMAS'2000*, pages 216–222, Boston, 2000.

[9] S. A. Lippman and J. J. McCall. The economics of job search: A survey. *Economic Inquiry*, 14:155–189, 1976.

[10] E. Ogston and S. Vassiliadis. Local distributed agent matchmaking. In *Proc. of the 9th International Conference on Cooperative Information Systems*, volume 2172, pages 67–79, 2001.

[11] D. Sarne and S. Kraus. The search for coalition formation in costly environments. In *Proc. of CIA2003*, pages 117–136, 2003.

[12] O. Shehory and S. Kraus. Methods for task allocation via agent coalition formation. *AIJ*, 15(3):218–251, 1998.

[13] R. Shimer and L. Smith. Assortative matching and search. *Econometrica*, 68(2):343–370, 2000.

[14] L. Smith. The marriage model with search frictions. *Journal of Political Economy*, 2004. to appear.

[15] K. Sycara, S. Widoff, M. Klusch, and J. Lu. Larks: Dynamic matchmaking among heterogeneous software agents in cyberspace. *Autonomous Agents and Multi-Agent Systems*, 5:173–203, 2002.

[16] G. J. van den Berg. Nonstationarity in job search theory. *Review of Economic Studies*, 57:255–277, 1990.