Analysing Partner Selection through Exchange Values

Maira Ribeiro Rodrigues and Michael Luck

School of Electronics and Computer Science, University of Southampton, Southampton, UK
mrm03r,mml@ecs.soton.ac.uk

Abstract. Dynamic and resource-constrained environments raise interesting issues for partnership formation and multi-agent systems. In a scenario in which agents interact with each other to exchange services, if computational resources are limited, agents cannot always accept a request, and may take time to find available partners to delegate their needed services. Several approaches are available to solve this problem, which we explore through an experimental evaluation in this paper. In particular, we provide a computational implementation of Piaget's exchange-values theory, and compare its performance against alternatives.

1 Introduction

In many disciplines, like medicine and biology [1], researchers are discovering the advantages of collaborative research, in which different types of information and tools are exchanged in order to improve individual or global results. In an ideal situation, in which computational resources are plenty, individuals can collaborate with each other by performing services or giving access to information to any suitable (authorised) requester. However, when there are limited computational resources in the system, the participants must choose which requests to accept. Such collaborative systems are also dynamic, in the sense that participants change their needs and provided services over time due, for example, to changes in research interests, and in the incorporation or development of new tools and information.

Such dynamic and resource-constrained environments raise interesting issues for partnership formation. First, because computational resources are limited, participants cannot always accept a request and may take time to find available partners to delegate their needed services. Second, because the participants may change their needed and provided services over time, maintaining a collaborative interaction becomes more complicated, since the link between clients and services is not permanent.

As a consequence of participants having to choose which requests to accept and to whom to send requests that have a higher chance of being accepted, they need some criteria to guide their decision over partners. This guidance primarily involves two things: first, sending requests to those participants that are more likely to perform the service and, second, using a strategy for accepting or refusing requests from other participants that favours the possibility of future interactions.

One way to select between interaction partners is to restrict the collaboration to situations in which a service dependency is observed. In this case, the concept of dependence [2] can be used to motivate the exchange of services between participants and to influence their decisions about interactions. Although this dependence approach is

powerful, we believe it is more suitable for situations in which the link between service provision and service request is static. Since dependence relations are built upon the services each individual needs and provides, if they change over time, the dependence network could lose accuracy, yet continual updates are likely to be expensive.

In response, we propose to address the problems of partnership formation and partner selection by using a system of exchange values, based on Piaget's theory [3]. Here, exchange values are formally defined as *qualitative values* that individuals associate with their interactions with other individuals, indicating the effort, cost, satisfaction, and benefit to each individual of the received or performed services and, thus, can influence the behaviour of interacting individuals. Piaget defines two main functions for exchange values in social interactions: as a means for individual decision-making, and as a regulation tool for guaranteeing the continuity of social interactions, since exchange values imply *moral* and *legal commitments* made by the agents during their interactions.

The application of exchange values for supporting the exchange of services between agents was first proposed in [4], in which the social reasoning mechanism is based on Piaget's system. We believe this approach is suitable for addressing the problem of partner selection in dynamic and resource-constrained environments for two main reasons: first, exchange values provide a system of credits and debits that are seen as moral commitments between agents, which motivates and favours the chances of future interactions (e.g., a credit that is gained by performing a service can be charged in the future) and, second, the set of exchange values of each individual (agent) is not related to the specific services they perform, but to the results of past interactions.

Other notions of values have appeared in the multi-agent systems literature. Antunes and Coelho [5], for example, use a notion of "multiple values" to improve agents' decision-making process by explicitly including these values in their agent architecture, the BVG architecture. Although this effort relates in one way or another to the concept of values, it either does not encompass the idea of exchange values, or does not have the moral commitment implications (that suggest reciprocity and potentially regulation) of the notion of values in Piaget's approach.

In this paper, we investigate these approaches to interaction, through an implemented experimental testbed with a multi-agent simulation system. In particular, there has not been an implementation of the system of exchange values, so the results are novel and valuable. The key contribution is thus the computational model of the exchange values approach, and the associated experimental analysis of the performance of this approach in comparison to alternatives.

The paper begins with an introduction to the different social approaches we consider, followed by a description of the scenario we use, as well as the details of the strategies used (dependence-based and value-based) in a computational context. We end by presenting the experimental simulations and results obtained.

2 Background

This section describes two theoretical approaches that can be used to address the problem of partnership formation: dependency theory and Piaget's system of exchange values, both as a basis for decision making about interaction partners.

2.1 Dependence Theory

The concepts of *social dependence* and *social power* in multi-agent systems were first proposed in [2], with the basic idea that one of the fundamental notions of social interaction is the *dependence relations* between agents. This represents the situation in which an individual needs a resource (e.g., an object, a task to be performed, a piece of information) that he does not possess or have access to, but which can be provided by another individual, giving the latter influence over the former. The basic definition of social dependence is that an agent x depends on an agent y regarding an action a needed for achieving a goal g, if x is not capable of performing a but y is. In this case, agent y's action a is viewed by agent x as a resource for achieving g.

Associated with this is the notion of *power of influence* of an agent over another. Dependence relations can be associated with power of influence in the sense that when agent x depends on agent y for achieving a goal, it becomes susceptible to the influence of the latter agent. In this view, agents structure their interactions to take advantage of situations in which they can exert influence to ensure the compliance of agents with their requests. Consequently, the formalisation of these relations allows agents to achieve better interactions since an agent is able to control its interactions according to its own interests, by obtaining power through dependence relations.

In this way, some models for coalition formation based on dependence relations have been proposed [6] [7], in which agents try to form coalitions by identifying dependencies with other agents in the society.

2.2 Piaget's Theory of Exchange Values

Piaget's theory of exchange values studies and formalises the dynamics of values in such a way that they can form an exchange system [3], specifically in the context of the exchange of *services* between individuals, i.e., actions that an individual performs *on behalf* of another individual. The values that arise from such exchanges are called *exchange values*, and can be seen as *moral values*, concerning *moral debits* (the obligation to perform new services in return for services previously received) and *moral credits* (the right to demand the performance of new services in return for services previously given).

One important characteristic of this notion of exchange values is that they are of a qualitative nature, with quantitative values appearing only in the particular case of the modelling of exchanges involving economic values.

Piaget's theory assumes two conditions for the existence of an exchange value system: (i) the individuals involved in an interaction must share a *common scale of values*, to ensure the compatibility of their evaluations of performed and received actions (services); and (ii) there must be conservation of the exchange values in time, otherwise, if values suffer depreciation, the continuity of the interactions and the functioning of the whole society can be risked. If these two conditions hold, a system of exchange values can be seen as a mechanism for regulating (or coordinating) the social exchange of services between agents, guaranteeing their continuity (and thus the continuity of the society).

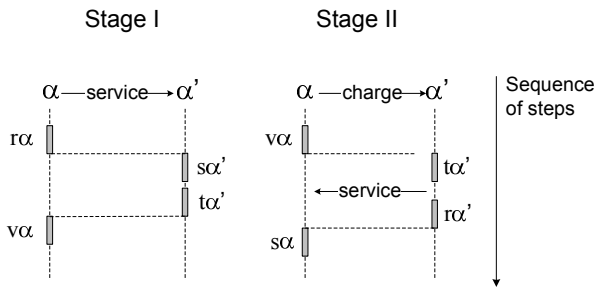

Fig. 1. The two stages of exchange.

Complete exchanges between individuals occur in two stages, whose basic forms are as follows: (I) an individual, say α, performs an action on behalf of another, say α', acquiring some credit for that action; and, (II) α charges his credit, asking α' to perform some action for him, in return. When α' performs that action, the exchange is complete. The event sequence for the two stages of a basic social exchange between α and α' is shown in Figure 1.

Piaget's theory identifies four types of exchange values involved in the exchanges between two individuals: *renouncement value*, *satisfaction value*, *acknowledgement value*, and *reward value*. These values are determined by the individuals over time, and can be positive, negative, or null, which, as a consequence, can provide a debit or credit in relation to others.

The first stage of an exchange consists of four steps, as shown on the left of Figure 1:

1. α performs a service on behalf of α' and associates with this action a renouncement value (r_α), representing the effort, or *investment*, in performing the action;
2. α' receives the performed action, and associates a satisfaction value ($s_{\alpha'}$) with the received action;
3. α' then acknowledges a debt with α through an acknowledgement value ($t_{\alpha'}$), which may be used in the future;
4. α feels valued with the recognition of α', and associates to this valuation a reward value (v_α), which may also be used in the future.

At the end of the first stage, α' has acquired a debt ($t_{\alpha'}$) with α, and α has acquired a credit (v_α) with α'. Note that since these are from the individual perspective of each, they may not be equal, and may not be known by the other. In this stage, $r_\alpha$and $t_{\alpha'}$are negative, while $s_{\alpha'}$and $v_\alpha$are positive; a negative $t_{\alpha'}$is viewed as debit for α', and a positive $v_\alpha$is viewed as a credit for α. Later on, α can charge the credit with α' by requesting performance of some service that benefits α in return. This begins the second stage of the exchange process, which follows the steps shown on the right of Figure 1. Here, $r_{\alpha'}$and $v_\alpha$are negative, while $t_{\alpha'}$and $s_\alpha$are positive. In this way, we say that the first stage represents an *accumulation* of exchange values, and the second stage

represents their *realisation*, since the debt $t_{\alpha'}$ and the credit v_α are first generated, and the positive value of $t_{\alpha'}$ means α' has actually repaid its debt, and the negative value of v_α means that α has charged its credit.

If the amount assigned for each exchange value in one stage of the exchange is the same, i.e., if they directly correspond, the system is said to be in *equilibrium* with respect to that exchange. This equilibrium situation can be represented by the following logical implication:

$$\textbf{Implication I} \qquad (r_\alpha = s_{\alpha'}) \wedge (s_{\alpha'} = t_{\alpha'}) \wedge (t_{\alpha'} = v_\alpha) \Rightarrow (v_\alpha = r_\alpha) \qquad (1)$$

From this, we conclude that, if the amounts assigned to the exchange values that result from the evaluations of the performed or received service are equivalent, then α is valued by α' proportionally to the service that was provided.

It is important to notice that all four values are related, because they all originate from the evaluation of the exchanged service. However, because this evaluation is subjective, and consequently individuals will use different, personal criteria in the evaluation (e.g., according to their character, personality, personal experience, etc), these values need not correspond. In summary, perfect or fair judgement results in a direct correspondence between the exchange values and, consequently, in exchanges in equilibrium, while imperfect or unfair judgement results in a divergence of the exchange values and, consequently, there is no equilibrium.

3 Scenario

In an effort to experiment with these models, we have adopted the context of a dynamic collaborative research environment (as in Bioinformatics), in which researchers exchange information and tools in order to complete experiments or validate hypotheses. The environment is dynamic because participants are likely to change their needs and capacities over time due, for example, to changes in research interests, the appearance of new theories, or development of new tools. In this kind of environment, researchers are motivated to collaborate, but not to be strictly benevolent. In other words, researchers do not need to collaborate with everyone. Consequently, when a researcher needs a service from others, he or she must choose those agents that are likely to perform the service and, at the same time, adopt a *criterion* for accepting or refusing requests from other researchers in a way that favours the possibility of future interactions. In this scenario, we assume that each researcher is represented by an agent that takes decisions about partners on his/her behalf.

The scenario thus consists of a group (society) of agents, each of which has a list of services they need to execute, and a list a services they can provide (i.e., the agents are service clients and providers at the same time); agents are supposed to exchange services in order to achieve individual goals. Since participants in a collaborative research environment are likely to change their needs over time, the agents that model them also change their lists of needed services at a regular rate.

More specifically, the environment imposes a restriction on computational resources by limiting agent capacity for service provision. This is done by assigning to every service provider (or agent) a capacity indicating the maximum number of services they

can perform at one time. Consequently, when capacity is full, an agent cannot accept more requests to perform a service. In addition, we assume that agents are not capable of performing their needed services (since we want to evaluate service exchange), but they always find interaction partners. High level relations between agents like goals or organisations are not considered, just relations between provided and needed services, so relationships like mutual or reciprocal dependence are omitted.

3.1 Strategies

A decision making process towards the selection of partners, and the continuity of interactions, must establish not only the process of *choosing a partner*, but also the process of *analysing the received requests*, since the acceptance or refusal of a request will influence the chance of future interactions. In this context, an agent's decision over partners has two different contexts: when the agent receives a request message and has to decide whether to accept it; and when the agent must choose agents to send a service request, giving preference to those more likely to accept the request.

A common way to select between interaction partners is to restrict the collaboration to situations in which dependencies are observed. For example, an agent A is likely to collaborate with agent B by performing a service on its behalf, only if B is capable of performing a service that A needs. In the same way, if B needs to request a service from others, it will give preference to those with which it identifies a dependence (or that are seen as more likely to collaborate with it). In this case, the *criterion* used in the selection process is the *dependence*. We call this approach the *dependence-based approach* for decision making regarding interaction partners.

Another way of selecting between partners is to use exchange values as a basis for decision making. This uses the basic exchange values system, according to which the agent's decision is not based on the services each agent can perform, but on the results of past interactions and expectations of future interactions. It implements a system of credits and debits, derived from the performed and received services, and is used to choose those agents that are more likely to perform tasks, as well as to choose whether to collaborate with other agents. In this case, the *criterion* used is the set of *exchange values*. We call this approach the *value-based approach* for decision making. In what follows, we refer to the former of these types of agents, which employ dependence-based reasoning as *DAgents*, and the latter value-based agents as *VAgents*.

To analyse requests for a service, an agent considers: information on past interactions and on services provided and by whom; current capacity for service provision; and preferences and policies regarding interactions. To analyse possible partners to send service requests, an agent considers information on past interactions and services provided; and preferences and policies regarding interactions.

VAgents must maintain an individual *history of interactions* in which they participate, with the resulting exchange values, and *state of exchange values*, which is the set of exchange values (r,s,t,v) determined through interactions with other agents. Both the history and the state of exchange values are updated after every interaction. *DAgents* keep a record of the agents for which they performed a service, a *service log*, in order to derive information on dependences.

Table 1. Dependence-based algorithm for sending service requests.

For each sn_i in S_n do
search the resulting L_{pp} for sn_i;
order L_{pp} giving priority to agents in L_{DepOn};
send request for first agent in L_{pp};
if request refused then
repeat
send request to next agent in L_{pp} ;
until request accepted OR $L_{pp} = \phi$
if $L_{pp} = \phi$ AND request not accepted then
go to line 4 and repeat the process.

Information on past interactions and on services is used according to each approach. For example, the dependence-based approach uses information about service provision to infer the dependence between agents, while the value-based approach uses information on past interactions related to the history and state of exchange values.

In the context of this paper, we assume the requests are analysed as soon as they arrive, and that when the agent's current capacity is less than half full it will always accept requests. However, when the current capacity is equal to or more than half the maximum, the agent's decides whether to accept the request based on a set of preferences or policies that are different for each approach. For example, the dependence-based approach uses policies related to dependence relations, while the value-based approach uses policies related to the state and history of exchange values.

In this scenario, agents request needed services according to the initial configuration, one after another, and must deal with any request messages that arrive. To decide to which agent a request is sent, and from which agents a request should be accepted, the agents take into account the limitations imposed by the environment, and also the criteria defined by each approach for partner selection, dependence-based or value-based.

3.2 Dependence-based Agents

DAgents use information on dependencies to choose between interaction partners based on the assumption that partners with which a dependence is identified are considered to be more likely to accept requests. Conversely, requesters in a dependence relation can influence the decision when analysing incoming service requests.

Formally, if we take D_{ag} to be a dependence-based agent, L_{pp} the list of possible partners for D_{ag} when requesting a service sn_i from the needed services set S_n, sr_i a service being requested from D_{ag}, L_{DepOn} the agents that depend on D_{ag} to perform a service, and L_{IsDep} the list of agents on which D_{ag} is dependent, we can define two algorithms to represent D_{ag}'s decision making. The algorithm in Table 1 describes the decision process for sending requests for services, and the algorithm in Table 2 describes the decision process for analysing incoming requests.

In this approach, preference is given to agents with which a dependence relation is observed, since they are more likely to accept a request. In order to determine the current

dependence relation, DAgents search the service log for agents for which they have previously performed a service (so that these agents depend on them for that service). When resources are limited (current capacity is almost full), requests cannot always be accepted, we assume that DAgents prefer to collaborate only with agents they depend on for performing a service, as shown in Table 2.

3.3 Value-based Agents

As described above and proposed in [4], agents using the exchange values approach store four types of values: renouncement (r), satisfaction (s), acknowledgement (t) and reward (v). For each interaction in which a Vagent (e.g, Ag1) participates, it stores an array of exchange values (V_{Ag1Ag2}, which contains the accumulated values for $(r; s; t; v)$) associated with the other agent involved (Ag2). Along the various exchange processes, those values can suffer variations and can be decreased or increased. So, we define $\Delta_I V_{Ag1Ag2}$ and $\Delta_{II} V_{Ag1Ag2}$ to be the vectors indicating the change to exchange values when Ag1 interacts with Ag2, and the same for the change to exchange values of the partner Vagent ($\Delta_I V_{Ag2Ag1}$). After each interaction, VAgents calculate the variations in exchange values and update the state and history of exchange values with these variations.

For VAgents, we take a simplified value set for exchange values, namely integers (though other models are possible), and define each increase or decrease during a single step of an exchange process to be by an integral number of units, $+n$ or $-n$. We also simplify the way values are calculated, by assuming the values are always in equilibrium, and that the provider communicates his *r* value for the current exchange to the receiver, together with the service completion notification message. The receiver than assigns that value to his *s* and *t* values.

As before, if we take V_{ag} to be a VAgent, L_{pp} to be the list of possible partners for V_{ag} when requesting a service sn_i from the needed services set S_n, sr_i a service being requested from V_{ag}, $L_{HasCred}$ the list of partner agents with which V_{ag} has credits, and L_{HasDeb} the list of agents with which V_{ag} has debts, we can specify two algorithms to represent the V_{ag}'s decision making. In Table 3, we show how to send service requests and in Table 4, we show how to analyse incoming requests.

Table 2. Dependence-based algorithm for analysing incoming requests.

For each sr_i do
  if $(current\_capacity < maximum\_capacity/2)$ then
    accept request;
  if $(current\_capacity = maximum\_capacity)$ then
    refuse request;
  if $(current\_capacity >= maximum\_capacity/2)$ then
    if requesting agent is in L_{IsDep} then
      accept request;
    else **refuse** request.

Table 3. Value-based algorithm for sending requests for services.

For each sn_i in S_n do
  search the resulting L_{pp} for sn_i;
  order L_{pp} giving priority to agents in $L_{HasCred}$;
  send request for first agent in L_{pp};
  if request refused then
    repeat
      if refusing agent is in $L_{HasCred}$ then
        devalue refusing agent;
      send request to next agent in L_{pp} ;
    until request accepted OR $L_{pp} = \phi$
  if $L_{pp} = \phi$ AND request not accepted then
    go to line 4 and repeat the process.

Table 4. Value-based algorithm for analysing incoming requests.

For each sr_i do
  if $(current\_capacity < maximum\_capacity/2)$ then
    accept request;
  if $(current\_capacity = maximum\_capacity)$ then
    refuse request;
  if $(current\_capacity >= maximum\_capacity/2)$ then
    if sender is charging a credit then
      accept request;
    else if sender has negative exchange history then
        refuse request;
      else **accept** request.

In this approach, preference is given to agents with which an agent has credits, since they are more likely to accept a request because they have a *commitment* with the requester, and are motivated to collaborate and pay their debts (a service was received, and a service is provided in return). Since the algorithm was defined in such a way that an agent first selects the possible partners for a needed service (that is, those that have the capability to provide it), and then uses the information about debts to order them, the situation in which an agent is not able to pay a debt because it is not able to perform the requested service is not considered.

If a request charging a credit is refused, it indicates that the other agent did not keep its commitment, so it is devalued (the refusing agent's credit value suffers a negative variation). This indicates that the refusing agent is a poor partner for collaborative interactions. When resources are limited and requests cannot always be accepted, we assume that a VAgent prefers to collaborate with agents with which it has debts, since they will try to keep their commitments by paying their debts and completing the exchange, or with agents with a good collaboration history (and do not have a negative

exchange result in the history of exchanges). In summary, collaboration is modelled by seeking complete exchanges, where services are received and performed in return.

4 Experiments

In seeking to determine which approach performs better in the changing environment, we have implemented an experimental testbed and undertaken some experiments to compare the success of each approach in exploring collaborative situations and finding interaction partners in dynamic, resource-constrained environments.

To do this, we measure the number of request messages that an agent must send until the service request is accepted (the *effort*), and the number of services an agent can delegate (the *task completion* or accomplishment). It is expected that agents with good performance will have a low value for effort (indicating that they were successful in choosing partners more likely to perform a service), and a high value for task completion (indicating that they were able to delegate more services). In summary, it is desirable that agents find partners quickly (and hence reduce the number of messages in the system), but also that the reasoning process does not consume too much time and consequently reduce the number of delegated services. Since we have restricted the scenario, assuming that there is always a provider for every available service, we guarantee that the task completion measure only represents tasks that could not be completed because partners did not accept them.

4.1 Simulation Configuration

A simulation in our experiments consists of a number of agents requesting services from, and performing services for, others at the same time, for a predetermined duration. For the agents' service configuration, each agent has a list of services it needs to request from other agents, and a list of services it can provide. The number of needed and provided services is the same for all agents, but there is no relationship between the number of needed services and the number of provided services. In addition, all available services are allocated to at least one provider, and no agent can provide and request the same service. This kind of *service configuration* is shown in Table 5 where, for example, Vagent1 needs services $s1, s2, s2, s1$ in order, and can provide $s4$ and $s3$ (in any order, and possibly concurrently). The service configuration is changed by randomly replacing half the services in the needed services list of all agents in the system. Each simulation has a basic simulation configuration (e.g., duration of the simulation,

Table 5.

Agent	**Needs**	**Provides**
$Vagent1$	$S1, S2, S2, S1$	$S4, S3$
$Vagent2$	$S3, S4, S5, S3$	$S1, S2$
$Dagent3$	$S1, S3, S5, S4$	$S2, S6$
...		

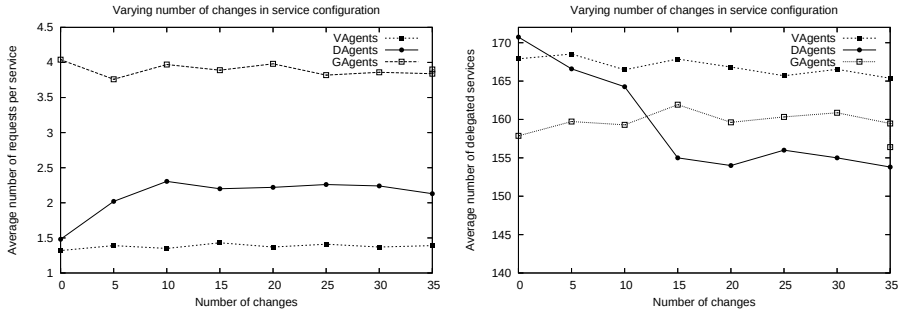

Fig. 2. Varying changes in the agents service configuration.

total number of agents, total number of available services, maximum capacity for providing services at the same time, frequency of changes in service configuration, etc), and performed three different experiments in order to analyse agent behaviour under different conditions. For example, if we want to vary the number of agents, we perform multiple simulation runs, each with a different value for the number of agents.

Also, to compare agent behaviour in uniform environments and heterogeneous environments, we defined some simulations to run with only one type of agent (single), and others with both types of agents (mixed). As a baseline, we also used a generic agent (GAgent) with random partner selection.

4.2 Results

First, we varied the number of changes in the service configuration, to see if there is a difference in the performance of each approach when there is no change at all in the environment, and when the changes are more frequent. For this experiment, we defined simulations, each with single groups of 30 VAgents, DAgents and GAgents.

The results, in Figure 2, show that VAgents generally performed better, in both effort and task completion, and also had stable behaviour in relation to the number of requests per service (effort) and delegated services (task completion), despite the changes. DAgents had an increase of effort and a decrease in the number of tasks completed in response to the increase in the number of changes in service configuration.

In the second experiment, we varied the number of agents in the system, to observe the behaviour of the agents when we scale up the system. Here, we also simulated single groups of VAgents, DAgents and GAgents and used a frequency of 25 changes per simulation. The results in Figure 3 again show that VAgents perform better with a convergence to stable behaviour in terms of effort (in requets) but with a decrease in task completion as the number of agents increases. We also observe that all approaches showed a decrease in their number of completed tasks in response to the increase in the number of agents. Interestingly, for this experiment we also measured the number of requests sent by the agents, to investigate if the decrease in task completion was due to an increase in the number of requests that could be responsible for making agents busier. We found that for GAgents, the number of request messages sent increased, but

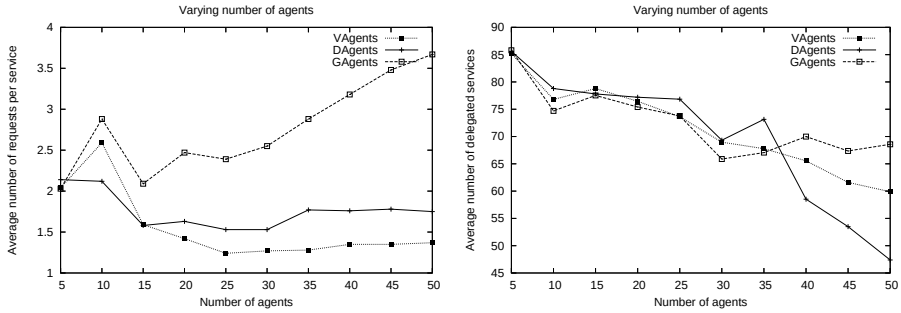

Fig. 3. Varying number of agents in the system.

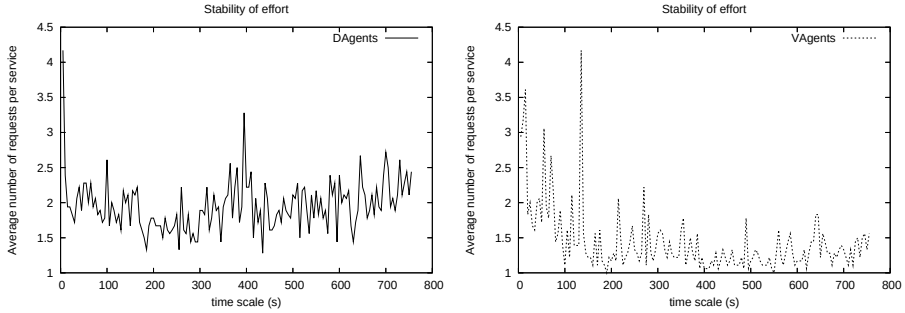

Fig. 4. Stability of effort during the simulation.

for the DAgents and VAgents the number decreased. This indicates that the latter spend more time in reasoning than in sending messages but, since the difference between task completion of VAgents and GAgents was relatively close, and the effort for VAgents was much better, we can conclude that as the number of agents increases the lower effort of VAgents compensates for the reasoning time. In comparison with DAgents, VAgents also performed better, stabilising at lower effort level and diverging from DAgents with better task completion as the number of agents increase.

Finally, we mixed the three types of agents in the same simulation, using the same number of agents of each type, keeping the parameters fixed to observe if their effort stabilises over time, indicating that they can adapt to the heterogeneous environment. In this experiment, we mixed VAgents, DAgents and GAgents, 16 of each type, and used a frequency of 25 changes per simulation. The results for DAgents and VAgents in Figure 4, show that although DAgents do not have very high values for effort, they cannot achieve stable behaviour in effort. By contrast, VAgents have slightly higher values for effort at the start of the simulation, but achieved fairly stable behaviour towards the end, with very small variations in the number of requests per service.

4.3 Discussion

The results in the previous section showed that in most cases VAgents performed better in the changing environment. However, we note that initial configurations can distort this as, for example, shown on the right side of Figure 2.

In the results for the second experiment (in Figure 3), the peaks we observe in the graphs are due to the distribution of services. Since each new agent added to the system has a different set of provided and needed services, a biased combination can either cause some agent to be busy if the new agents request services that were already scarce in the environment, or cause a more equal distribution of provided services if the new agents provide services that were scarce in the previous environment. Even though different service distributions could give different graphs, the results are not influenced, since the same configuration is used for all types of agents, and it is relative comparison between them that identifies which approach performs better.

In summary, the results show that each approach has different advantages, and they are not contradictory, but complementary. For example, one possibility of combining both approaches is changing the algorithm for sending requests in a way that it first selects the agents with dependencies as possible partners, and than orders this set according to the exchange values approach, or vice-versa. The dependence approach could also be used in cases in which an agent has no credits or debits with the possible partners (as a result of not having encountered them previously).

5 Conclusions and Future Work

In this paper, we have described an experimental testbed with a multi-agent simulation system to simulate the process of interaction partner selection in dynamic and resource-constrained environments. To model agent decision making in relation to interactions and interaction partners, we used two different approaches: a dependence-based approach and an exchange values-based approach. These address the issue of finding a partner more likely to accept requests in contrast to alternative approaches that seek the partner with the highest utility (as proposed in [8, 9]).

We described the experiments undertaken to test the behaviour of agents using both approaches in the task of finding interaction partners, and specifically to analyse the behaviour of agents using exchange values in their reasoning, since there is no previous empirical evidence of this approach. The results showed that agents using exchange values-based decision-making were more stable in their behaviour, and needed less effort to find interaction partners than the agents using a dependence-based approach, despite the changes in the environment. Also, by implementing a generic agent type that chooses interaction partners randomly, and comparing it to the other two types of agents, we observe that a strategy aimed at choosing interaction partners can significantly reduce the effort to find interaction partners, which can be very important when dealing with systems with resource limitations and dynamic behaviour.

Apart from these advantages, we believe that applying the system of exchange values in multi-agent systems can help more generally in regulating interactions, and motivating continuity of interactions, since exchange values imply moral and legal commitments, made by the agents during their interactions. Future work is to analyse the

combination of both approaches, dependence-based and exchange values-based (as proposed in [4, 10]) for the purpose of partnership formation. We also aim to extend agent reasoning and partner selection to include the notion of evaluation of provided and received services, so that the best interaction partner for delegating a service in terms of performance satisfaction can also be considered.

Acknowledgements: The first author is supported by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) of the Brazilian Ministry of Education. We would also like to thank Antonio Carlos da Rocha Costa (UCPel, Brazil) for his valuable comments regarding the Piaget's theory of exchange values.

References

1. Overbeek, R., Disz, T., Stevens, R.: The SEED: A peer-to-peer environment for genome annotation. Communications of the ACM **47** (2004) 47–50
2. Castelfranchi, C.: Social Power: A point missed in multi-agent, DAI and HCI. In Demazeau, Y., Mller, J., eds.: Decentralized A.I. Elsevier Science, Amsterdam (1990) 49–62
3. Piaget, J.: Sociological Studies. Routlege, London (1973)
4. Rodrigues, M.R., da Rocha Costa, A.C., Bordini, R.H.: A system of exchange values to support social interactions in artificial societies. In: Second International Joint Conference on Autonomous Agents and Multiagent Systems, Melbourne (2003) 81–88
5. Antunes, L., Coelho, H.: Decisions based upon Multiple Value: The BVG agent architecture. In Barahona, P., Alferes, J.J., eds.: Proc. of the 9th Portuguese Conference on Artificial Inteligence, EPIA. Progress in A.I. Volume 1695 of Lecture Notes in Artificial Intelligence., Portugal, Berlin, Springer-Verlag (1999) 297–311
6. Sichman, J.S., Conte, R., Castelfranchi, C., Demazeau, Y.: A social reasoning mechanism based on dependence networks. In: European Conference on Artificial Intelligence. (1994) 188–192
7. David, N., Sichman, J.S., Coelho, H.: Agent-based simulation with coalitions in social reasoning. In Moss, S., Davidson, P., eds.: MultiAgent-Based Simulation. Volume 1979 of Lecture Notes in Artificial Intelligence., Germany, Springer-Verlag (2001) 245–266
8. Saha, S., Sen, S., Dutta, P.S.: Helping based on future expectations. In: Second International Joint Conference on Autonomous Agents and Multiagent Systems, Melbourne (2003) 289–296
9. Sarne, D., Kraus, S.: Time-variant distributed agent matching applications. In: Third International Joint Conference on Autonomous Agents and Multiagent Systems, New York New York (2004) 168–175
10. Rodrigues, M.R., da Rocha Costa, A.C.: Using qualitative exchange values to improve the modelling of social interactions. In: Second International Joint Conference on Autonomous Agents and Multiagent Systems. Multiagent-based Simulation, Melbourne (2003)