中央研究院
資訊科學研究所
Institute of Information Science, Academia Sinica • Taipei, Taiwan, ROC

TR-IIS-06-013

Collaborative Assignment Using BDI Multiagent Negotiation

Kiam Tian Seow, Kwang Mong Sim and Yuan Chia Kwek

January 25, 2007 || Technical Report No. TR-IIS-06-013

http://www.iis.sinica.edu.tw/LIB/TechReport/tr2006/tr06.html

Collaborative Assignment Using BDI Multiagent Negotiation

Kiam Tian Seow, Kwang Mong Sim and Yuan Chia Kwek

***Abstract*— In this report, we propose a distributed agent model that embodies belief-desire-intention (BDI) reasoning and negotiation for addressing the linear assignment problem (LAP) collaboratively. In resource allocation, LAP is viewed as seeking a concurrent allocation of one different resource for every task to optimize a linear sum objective function. The proposed model provides a basic agent-based foundation needed for efficient resource allocation in a distributed environment. A distributed agent algorithm realizing the BDI negotiation model is developed and examined both analytically and experimentally. The significance of the model and its algorithm is also discussed in relation to existing multiagent work.**

***Index Terms*— Intelligent Agents, BDI Negotiation Model, Reasoning Systems, Reasoning Control, Collaborative Linear Assignment Problem, Decision Support.**

I. Introduction

Efficient resource allocation is a basic problem inherent in a variety of real world applications. In this report, we study *linear assignment*, an important subclass of this problem, but in the modern context of multiagent systems [1], [2], and the significant role *collaborative negotiation* plays in a new distributed agent-based approach to the problem.

Classical assignment problems deal with the question of how to assign N distinct elements in a set to N distinct elements in another on a one-to-one basis in the best possible way; underlying the *assignment* is a combinatorial structure, with an objective function modelling the *best way* [3]. Assignment problems are distinguished by their different objective functions. Of fundamental interest is the linear (sum) assignment problem (LAP) which attracted decades of active research [3]. In a LAP instance, a table or matrix of assignment values is given; cast in the important context of concurrent resource allocation, each value indicates the A-QoS (application quality-of-service) of one resource for one task (e.g., see Table I, where the A-QoS of resource r_0 for task t_0 has a value of 14, and so on). The objective of LAP is to maximize the sum total A-QoS of the concurrent allocations[1] of one different resource for every task.

For decades, solving LAP has traditionally been done in a centralized fashion, producing many centralized algorithms [3] as highly efficient solutions. However, with modern advancement in communication and networking technologies, creating an ubiquitous (distributed) environment, it is becoming practically more effective or feasible to deploy multiple problem solvers cooperating for a wider variety of application problems that were hitherto not possible. In exploiting this infrastructure, we are increasingly seeing the need to have computational entities call agents [2, Ch.1] that go beyond being just *algorithmic*. Existing centralized LAP solutions were never intended for online deployment in such an environment.

Motivated by the naturalness and ease by which many a variety of applications can be characterized in terms of distributed interacting agents that co-operate [4], we propose a multiagent perspective to LAP, leading to a new collaborative problem solving approach. This distributed agent approach involves different task agents capable of interacting collaboratively with one another to select different resources during problem solving. The objective of LAP becomes the joint (social) goal of these task agents, and we call the resulting problem a collaborative LAP (CLAP). Importantly, this new perspective admits a logical or physical distribution

K.T. Seow is with the Division of Computing Systems, School of Computer Engineering, Nanyang Technological University, Republic of Singapore 639798. Email: asktseow@ntu.edu.sg

K.M. Sim was with the Institute of Information Science (IIS), Academia Sinica, Taipei, Taiwan (until August 2005). He gratefully acknowledges funding from IIS, Academia Sinica for supporting K.T. Seow's visit at IIS. He is currently with the Department of Computer Science, Hong Kong Baptist University, Kowloon Tong, Kowloon, Hong Kong. Email: bsim@comp.hkbu.edu.hk

[1]In this study, *allocating a resource to a task* is used interchangeably with *assigning a task to a resource*.

of information and processing that characterizes a distributed resource allocation (or task assignment) environment.

In addressing CLAP as a distributed agent problem, a task agent attempting to reach an optimal assignment faces the basic issue of deciding what action to perform. In so doing, each agent needs to reason about its beliefs and preferences as well as its collaborating agents', mediating through negotiative interactions among the agents during problem solving. In assignment, an agent's beliefs refer to its local information obtained that lead the agent to believe whether or not there are alternative resource selections that will help move closer towards the joint goal, and its preferences refer to the ordering of all its resource exchange options with different agents, each leading to an incremental social gain towards achieving the joint goal. The agent, possessing only partial A-QoS information, would need to determine its preferences through some form of reasoning that entails negotiation, by exchanging A-QoS information believed to be useful for determining its own preferences or the preferences of its collaborating agents. Conceptually the mediation also requires a simple arbitration agent, and a collaborative agent's preferences can naturally be viewed as the agent's *desires* (generated motivations), of which its *intention* (decisive stance) represents perhaps the best (local) desire that it commits to in an arbitrary round of negotiation. The result is a novel Belief-Desire-Intention (BDI) negotiation model for CLAP. The proposed model extends existing LAP solutions from *centralized algorithmic processing* to *distributed agent reasoning*.

Following, the key contribution in this report is a basic BDI negotiation model for CLAP, by which a distributed agent algorithm is developed (Sections II and III) that we examine both analytically and experimentally (Sections IV and V). Discussions in relation to existing multiagent research efforts examine the significance of the work (Section VI).

II. The Linear Assignment Problem (LAP)

A. Problem Formulation

Let $T = \{t_0, t_1, \cdots, t_{|T|-1}\}$ and $R = \{r_0, r_1, \cdots, r_{|R|-1}\}$ denote a set of tasks and resources respectively; and

$$d_{ij} = d[t_i, r_j], \text{ for } t_i \in T, r_j \in R$$

be a measure of the A-QoS that a resource $r_j \in R$ can offer to a task $t_i \in T$ upon allocation.

Assume $|T| \leq |R|$. Then formally, the objective of the $|T| \times |R|$ LAP is to find the particular (total) assignment mapping

$$\begin{aligned} \Pi : T \rightarrow R \text{ such that for } t_i, t_j \in T, \\ i \neq j \text{ implies } \Pi(t_i) \neq \Pi(t_j) \end{aligned} \quad (1)$$

and the total quality of service (total A-QoS)

$$S_{tot} = \sum_{i=0}^{|T|-1} d[t_i, \Pi(t_i)] \quad (2)$$

is maximized over all possible permutations of Π. Each permutation represents an assignment (or allocation) set.

$\Pi(t) \in R$ is referred to as a resource selection by task $t \in T$ (under an arbitrary permutation of Π). Intuitively, Π (1) specifies that no two different tasks select the same resource, and every task in T selects only one resource in R. An assignment set (or simply assignment) corresponds to one permutation of Π (1); and can also be equivalently represented as containing elements of the form $(t, \Pi(t)) \in T \times R$.

Note that if $|T| = |R|$, then every resource in R is selected (by one different task in T).

B. A Simple Example

Consider the following 3×3 LAP example, with

$$T = \{t_0, t_1, t_2\} \text{ and } R = \{r_0, r_1, r_2\}.$$

The individual A-QoS values are tabulated in T-R Table I.

TABLE I

A 3×3 LAP MATRIX

	r_0	r_1	r_2
t_0	{14}	5	8
t_1	2	6	{4}
t_2	8	{7}	3

For this example, the optimal solution consists of the assignment set $\{(t_0, r_0), (t_1, r_2), (t_2, r_1)\}$, with a maximal total A-QoS of 14 + 4 + 7 = 25.

Existing algorithms such as [5] can yield such optimal solutions. These are, however, minimization algorithms, so to solve our A-QoS maximization problem, they need to operate on a T-R table of *negated* A-QoS values.

LAP is fundamental in a variety of resource allocation applications. For example, in a multiple sensor-multiple target tracking problem, T represents a set of tracks and R represents a set of sensors. In the example, the A-QoS value $d[t_i, r_j]$ represents the effectiveness of the resource $r_j \in R$ for task $t_i \in T$, and the goal is to find a set of allocations (i.e., a permutation of Π (1)) that maximizes the total A-QoS (2).

C. CLAP: A Distributed LAP

In this section, we propose a new collaborative agent approach to LAP. This distributed approach involves different task agents capable of interacting collaboratively with one another to select different resources during problem solving.

The basic approach is to decompose the T-R table row-wise, such that each task agent represents (i.e., has the responsibility of selecting a resource $r \in R$ for) a task $t_i \in T$ and has A-QoS knowledge of task $t_i \in T$ only, namely, the individual A-QoS values $d[t_i, r]$ for all resources $r \in R$. For the example of Section II-B, three task agents are needed; each agent represents a different $t_i \in T$ and only has A-QoS knowledge contained in the t_i-row of Table I. A task agent representing task $t \in T$ is called a t-agent; for convenience, where it is clear in the context, we simply use the terms *task* $t \in T$ or *agent* $t \in T$ to refer to a t-*agent*.

The goal of LAP becomes the joint goal of these task agents, and we call the resulting problem a collaborative LAP (CLAP). In attempting to reach an optimal assignment solution, the basic issue a task agent faces is deciding what resource exchange option to propose, as detailed in Section III.

III. BASIC BDI NEGOTIATION MODEL FOR CLAP

A. An Overview

The proposed model divides the reasoning process into negotiation rounds, and in each round, performs negotiative means-end reasoning, where the *end* is to increase the social value, i.e., the total allocated A-QoS (2), using the *means* of resource exchange between two task agents. In each round, each task agent locally accesses and directly acts only on its own row of A-QoS data, and determines its *belief* set - the information or evidence that indicates all the possible options - the alternative resources - a task agent can exchange its current resource selection for to achieve its end. Every task agent then begins *negotiating* by communicating with one another to acquire A-QoS data from any task agent whose current resource selection is in the agent's belief set. In collaborating, any such agents will respond with the required A-QoS values, using which the agent would deliberate to determine its own *desire* set - the means of exchanging its current resource selection for options (that survive the deliberation) with the respective agents (currently holding on to these options). As a final step in a negotiation round, the agent will select the best (local) desire - the one that offers a net exchange gain that is the highest from the agent's perspective - as its *intention*, which it would then use as the basis for a resource exchange proposal. All the agents' resource exchange intentions (or the lack thereof) would undergo arbitration to decide which two agents to proceed with the resource exchange, before negotiation is concluded, and the next round begins. The negotiation process terminates when simultaneously, all task agents have no (more) intention to exchange resources.

B. BDI Concept Formalization

To formally ground the BDI concepts for CLAP, the following CLAP-specific data structures are formally defined in such a way that they can be naturally interpreted as a task agent's beliefs, desires and intentions computed in an arbitrary round of collaborative negotiation. In these definitions, the current resource selections of all agents refer to those made under an arbitrary permutation of Π (1).

Definition 1 (Belief Set B_i*):* Given that an agent $t_i \in T$'s current resource selection is $r^i \in R$. Then its (current) belief set B_i is given by

$$\mathsf{B}_i = \{r \in R \mid d[t_i, r] > d[t_i, r^i]\} \tag{3}$$

If $\mathsf{B}_i \neq \emptyset$, this means that agent $t_i \in T$ has at least one alternative resource selection $r \in \mathsf{B}_i$ that may lead to increase in total A-QoS (2) (when made in exchange with an agent whose current selection is $r \in R$).

Definition 2 (Desire Set D_i*):* Given that an agent $t_i \in T$'s current resource selection is $r^i \in R$ and its belief set is B_i, $\mathsf{B}_i \neq \emptyset$. An arbitrary agent $t_j \in T$ whose current resource selection is $r^j \in R$ is said to accept agent $t_i \in T$'s beliefs B_i if $r^j \in \mathsf{B}_i$. To generate the desired exchange options or desires D_i, agent $t_i \in T$ broadcasts its beliefs B_i and current selection $r^i \in R$, and an arbitrary agent $t_j \in T$ who accepts the beliefs would respond with a pair of A-QoS values $d[t_j, r^j]$ and $d[t_j, r^i]$, so that for each of the $|\mathsf{B}_\mathsf{i}|$ responses received, the corresponding resource exchange option $[(t_i, r^j), (t_j, r^i), \rho] \in \mathsf{D}_\mathsf{i}$ (i.e., is agent $t_i \in T$'s desire) if $\rho > 0$, where ρ is defined by

$$\rho = -d[t_i, r^i] + d[t_i, r^j] - d[t_j, r^j] + d[t_j, r^i] \quad (4)$$

If $\rho > 0$, it means that there is a net exchange gain if agent $t_i \in T$ gives up its current selection $r^i \in R$ and selects $r^j \in R$, and in exchange, agent $t_j \in T$ gives up its current selection $r^j \in R$ and selects $r^i \in R$. Thus, any desire $d \in \mathsf{D}_i$, when carried out, will definitely lead to an increase in total A-QoS without violating Π (1). Quite naturally, it provides the motivation for agent $t_i \in T$ to want to exchange its current resource selection.

Definition 3 (Intention I_i): Given that an agent $t_i \in T$'s desire set is D_i, $\mathsf{D}_\mathsf{i} \neq \emptyset$. Then, agent $t_i \in T$'s intention I_i is given by

$$\begin{aligned} &I_i = [(t_i, r^j), (t_j, r^i), \rho] \in \mathsf{D}_\mathsf{i}, \text{ for which} \\ &\rho = \max\{\rho' \mid [-, -, \rho'] \in \mathsf{D}_\mathsf{i}\ \} \end{aligned} \quad (5)$$

Agent $t_i \in T$'s decisive stance or intention has to be I_i since it is the best exchange option that the agent can propose. It is said to have no intention if either $\mathsf{B}_i = \emptyset$ or $\mathsf{D}_\mathsf{i} = \emptyset$.

Finally, in the role of arbitration, an intention with the highest exchange gain, i.e., one that contributes to the highest increase (in total A-QoS) if carried out, is selected from all the agents' intentions $I_i \in \mathsf{I}$ gathered.

With the above formalization, a distributed agent algorithm that realizes the BDI negotiation model is proposed in the next section. This algorithm is referred to as a Multi-Agent Assignment Algorithm (MA^3), and handles the simple role of arbitration through a dedicated agent.

C. Distributed Agent Algorithm

MA^3 assumes that $|T| = |R| = N$, and consists of an arbitration agent (or arbiter) and a team of t-agents, $t \in T$. Agent $t \in T$ only has A-QoS knowledge of the task it represents, i.e., $d[t, r]$ for all $r \in R$. Each task agent initially selects a resource $r \in R$ according to (a permutation of) $\Pi : T \rightarrow R$ (1). The arbiter then initiates negotiation.

	r_0	r_1	r_2
t_0	{14}	5	8
t_1	2	{ 6}	4
t_2	8	7	{3}

Table (a)

$\overset{\text{MA}^3}{\Longrightarrow}$

	r_0	r_1	r_2
t_0	{14}	5	8
t_1	2	6	{4}
t_2	8	{7}	3

Table (b)

Fig. 1. Example to illustrate MA^3

1) Algorithmic Details: The generic BDI reasoning mechanism of a task agent and the simple role of the arbitration agent in an arbitrary round of collaborative negotiation can now be described as follows:

MA^3 : Collaborative (Task) Agent

---

1) If agent believes that there are alternative resource selections which may lead to increase in total A-QoS, it would, based on its (local) beliefs, generate the desired exchange options or desires, from which the best option will be chosen as its intention.
2) Agent submits its intention (or the lack thereof) to the arbitration agent.
3) Concurrent with Step 1 and Step 2, it responds to any requesting task agent whose beliefs it accepts, by sending to the requesting agent the A-QoS values as required for computing the requesting agent's desire.
4) Agent changes its resource selection (and then acknowledges it), proceeds to next round of negotiation or quit, as decided by the arbitration agent.

MA^3 : Arbitration Agent

---

1) Agent first receives the intentions (or the lack thereof) of all the task agents.
2) If agent sees that all task agents have no intention to exchange, it terminates the negotiation by telling all task agents to quit.
3) Otherwise, it
   a) selects an intention with the highest exchange gain and instructs the two agents concerned to proceed with the resource exchange;
   b) receives acknowledgement of resource exchange made as instructed (from the two agents concerned), before telling all task agents to proceed to next round of negotiation.

---

2) An Example: To illustrate the working mechanisms of the proposed MA^3, consider the earlier example problem presented in Section II-B. Fig. 1 shows two assignment tables for the problem, in which the resource selection of each agent $t_i \in \{t_0, t_1, t_2\}$ is represented by enclosing the corresponding A-QoS value within $\{\}$.

Referring to Fig. 1, Table (a) represents a randomly selected (initial) assignment and Table (b) represents a solution assignment. The following illustrates how the solution can be obtained by collaborative negotiations.

- Round 1
  - Agent t_0 selects r_0 and agent t_1 selects r_1 (as initialized). Both agents believe that they have the best selection because their belief sets are empty, hence no desire, and therefore no intention.
  - Agent t_2 selects r_2 (as initialized) but believes that there are alternative resource selections that may increase the total A-QoS, namely resources r_0 and r_1. It therefore generates its desired exchange options as follows:
    * for exchange with agent t_0, the exchange gain is
    $$-d[t_2, r_2] + d[t_2, r_0] - d[t_0, r_0] + d[t_0, r_2]$$
    which is equal to $-3 + 8 - 14 + 8 = -1 \leq 0$.
    * for exchange with agent t_1, the exchange gain is
    $$-d[t_2, r_2] + d[t_2, r_1] - d[t_1, r_1] + d[t_1, r_2]$$
    which is equal to $-3+7-6+4 = 2 > 0$.
    * Hence, its only desire is $[(t_2, r_1), (t_1, r_2), 2]$ and is therefore also its intention.
  - To get the required pairs of A-QoS values $\{d[t_0, r_0],$
  $d[t_0, r_2]\}$ and $\{d[t_1, r_1], d[t_1, r_2]\}$ for computing its desire set as done above, agent t_2 broadcasts its belief set $\{r_0, r_1\}$ and current selection $r_2 \in R$. The respective agents whose current resource selection is in agent t_2's belief set respond with those values. In subsequent rounds, such broadcasts and responses are deemed understood and will not be mentioned again.
  - Agents t_0, t_1 and t_2 send their intentions (or the lack thereof) to the arbitration agent.
  - The arbitration agent tells agent t_1 to change its resource selection to r_2 and agent t_2 to change it to r_1.
  - Once both agents t_1 and t_2 inform the arbitration agent that they have changed the selections as instructed, the arbitration agent tells all agents to proceed to next round of negotiation.
- Round 2
  - Agent t_0 selects r_0 and believes that it has the best selection because its belief set is empty, hence no desire, and therefore no intention.
  - Agent t_1 selects r_2 but believes that an alternative resource selection r_1 may increase the total A-QoS. It therefore generates its desired exchange options as follows:
    * for exchange with agent t_2, the exchange gain is
    $$-d[t_1, r_2] + d[t_1, r_1] - d[t_2, r_1] + d[t_2, r_2]$$
    which is equal to $-4+6-7+3 = -2 \leq 0$.
    * Hence, it has no desire, and therefore no intention.
  - Agent t_2 selects r_1 but believes that an alternative resource selection r_0 may increase the total A-QoS. It therefore generates its desired exchange options as follows:
    * for exchange with agent t_0, the exchange gain is
    $$-d[t_2, r_1] + d[t_2, r_0] - d[t_0, r_0] + d[t_0, r_1]$$
    which is equal to $-7 + 8 - 14 + 5 = -8 \leq 0$.
    * Hence, it has no desire, and therefore no intention.
  - Agents t_0, t_1 and t_2 send their lack of intentions to the arbitration agent.
  - The arbitration agent tells all agents t_0, t_1 and t_2 to quit. The final resource selections of the agents yield the solution as shown in Table (b) of Fig. 1.

IV. Theoretical Analysis

In this section, we present an analysis of the $N \times N$ CLAP via a formulation of an assignment reachability graph for the problem. Using the properties of the reachability graph, basic properties about the proposed MA^3, and hence the BDI negotiation model for CLAP, are formalized.

A. The Assignment Reachability Graph

We model the possible sequential execution of desires in a reachability graph as follows:

For a set of tasks T and a set of resources R, for which $|T| = |R| = N \geq 2$, let

$$\mathcal{G} \stackrel{\text{def}}{=} (V, D, \delta, V_o) \tag{6}$$

represent an assignment reachability graph (ARG) in which

1) V denotes a (nonempty) finite set of states uniquely characterizing the permutations of Π (1), and we write $\Pi(t)|_v$ to denote the resource selection of task $t \in T$ in state $v \in V$. $|V| = N!$. The total A-QoS (2) in a state $v \in V$ (i.e., a permutation of Π) is denoted by $|v|$ and given by

$$|v| = \sum_{i=0}^{(N-1)} d[t_i, \Pi(t_i)|_v].$$

2) $D \subseteq V \times V$ denotes a finite set of desires.
3) $\delta : D \times V \rightarrow V$ is a state transition function (due to resource exchange between two arbitrary tasks $t_i, t_j \in T$), such that $\delta(e_{ij}, v) = v' \in V$ iff $\Pi(t_i)|_{v'} = \Pi(t_j)|_v$ and $\Pi(t_j)|_{v'} = \Pi(t_i)|_v$ and the magnitude of $e_{ij} \in D$, $\triangle e_{ij}|_v > 0$, is defined by

$$\begin{aligned}\triangle e_{ij}|_v &= \{-d[t_i, \Pi(t_i)|_v] + d[t_i, \Pi(t_j)|_v]\} + \\ &\quad \{-d[t_j, \Pi(t_j)|_v] + d[t_j, \Pi(t_i)|_v]\} > 0.\end{aligned}$$

We can interpret $\triangle e_{ij}|_v$ as the increase in total A-QoS if task t_i and task t_j exchange their resource selections held in state $v \in V$, i.e., $\Pi(t_i)|_v$ and $\Pi(t_j)|_v$, respectively.
4) $V_o \subseteq V$ denotes a finite set of terminal states such that for $v_o \in V_o$, $\delta(e, v_o)$ is not defined for any $e \in D$.

Let D^* contain all possible finite sequences, or strings, over D, plus the null string ε. Then, definition of δ can be extended to D^* as follows:

$$\delta(\varepsilon, v) = v,$$

$$(\forall e \in D)(\forall w \in D^*), \delta(we, v) = \delta(e, \delta(w, v)).$$

For a string $s \in D^*$, $|s|$ denotes the length of the string, i.e., the number of elements of set D in string s. $|s| = 0$ if $s = \varepsilon$.

We conclude this section with the following definition.

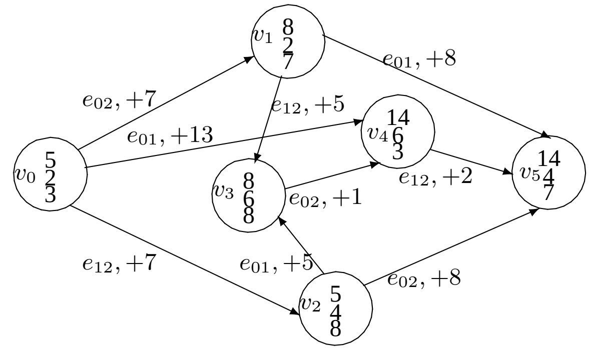


Fig. 2. Assignment reachability graph: An example

Definition 4 (Max-Transition): At an arbitrary state $v \in V$, a max-transition is a desire $e_{max} \in D$ for which

$$\triangle e_{max}|_v = \max\{\triangle e|_v \mid \delta(e, v) \in V\}. \tag{7}$$

A max-transition $e_{max} \in D$ is unique if $\triangle e_{max}|_v > \triangle e|_v$ for all $e \in D$, $\delta(e, v) \in V$ for which $e \neq e_{max}$. A max-transition represents the *best* intention that the arbitration agent selects to conclude a round of negotiation.

1) An Example: Note that an ARG $\mathcal{G}$ can be naturally described by a directed-transition graph. Fig. 2 shows an ARG for the simple example presented in Section II-B.

To interpret Fig. 2 correctly, it is necessary to take note of the following: For this example, as seen in T-R Table I, there are no equal A-QoS values in each row. Thus, for simplicity, in each state $v_i \in V$, we list only the A-QoS values such that the top most value is due to a resource selection by agent t_0, the next is due to that by agent t_1 and so on. In a state $v_i \in V$, one can easily determine from T-R Table I which resource has been selected by which agent. For instance, in state $v_0 \in V$, the top most value listed is 5 which is due to the resource selection by task t_0, thus we know that task t_0 selects resource r_1 in state $v_0 \in V$, since $d[t_0, r_1] = 5$ as in Table I. For $\delta(e_{ij}, v) = v'$, the information associated with e_{ij} at state $v \in V$ is represented as $e_{ij},\ \triangle e_{ij}|_v$.

In the illustration of the proposed MA^3 in Section III-C.2 that uses the same example problem, we note that the negotiation starts from state $v_4 \in V$ of the ARG in Fig. 2, since it is an equivalent (and unique) representation of Table (a) in Fig. 1. In the illustration, after round 1, the arbitration agent approves the only and best intention proposed, as represented by transition e_{12} that leads state $v_4 \in V$ to state $v_5 \in V_o$ representing Table (b) of Fig. 1. But although these agents have reached a terminal state of the ARG, the task agents only know about this in another round (i.e., round 2) of negotiation, when the arbitration agent receives the lack of intentions by all agents and informs them to terminate negotiation.

2) Properties of ARG: Below, we establish some basic properties of an ARG $\mathcal{G}$ (6).

Property 1: If $e \in D$ and $\delta(e, v_1) = v_2 \in V$, then $|v_2| > |v_1|$.

Proof : If $\delta(e, v_1) = v_2$, and arbitrarily $e = e_{ij} \in D$, then

$$|v_1| = \{\sum_{all\ \ k\in\{0,\cdots,N-1\}-\{i,j\}} d[t_k, \Pi(t_k)|_{v_1}]\} + \{d[t_i, \Pi(t_i)|_{v_1}] + d[t_j, \Pi(t_j)|_{v_1}]\} \quad (8)$$

$$|v_2| = \{\sum_{all\ \ k\in\{0,\cdots,N-1\}-\{i,j\}} d[t_k, \Pi(t_k)|_{v_2}]\} + \{d[t_i, \Pi(t_i)|_{v_2}] + d[t_j, \Pi(t_j)|_{v_2}]\} \quad (9)$$

But there is no change in resource selections among tasks in $T - \{t_i, t_j\}$ under transition $e_{ij} \in D$, thus we can let

$$\begin{aligned} c &= \sum_{all\ \ k\in\{0,\cdots,N-1\}-\{i,j\}} d[t_k, \Pi(t_k)|_{v_1}] \\ &= \sum_{all\ \ k\in\{0,\cdots,N-1\}-\{i,j\}} d[t_k, \Pi(t_k)|_{v_2}]. \quad (10) \end{aligned}$$

Therefore, we have

$$\begin{aligned} |v_2| &= c + \{d[t_i, \Pi(t_i)|_{v_2}] + d[t_j, \Pi(t_j)|_{v_2}]\} \\ & \quad \text{By combining Eqs. (9) and (10)} \\ &= c + \{d[t_i, \Pi(t_j)|_{v_1}] + d[t_j, \Pi(t_i)|_{v_1}]\} \\ & \quad \text{Because } \Pi(t_i)|_{v_2} = \Pi(t_j)|_{v_1} \text{ and } \Pi(t_j)|_{v_2} = \Pi(t_i)|_{v_1} \text{ by definition of } \delta(e_{ij}, v_1) = v_2 \\ &= c + \{d[t_i, \Pi(t_i)|_{v_1}] + d[t_j, \Pi(t_j)|_{v_1}]\} + \\ & \quad \{-d[t_i, \Pi(t_i)|_{v_1}] + d[t_i, \Pi(t_j)|_{v_1}]\} + \\ & \quad \{-d[t_j, \Pi(t_j)|_{v_1}] + d[t_j, \Pi(t_i)|_{v_1}]\} \\ &= |v_1| + \{-d[t_i, \Pi(t_i)|_{v_1}] + d[t_i, \Pi(t_j)|_{v_1}]\} + \\ & \quad \{-d[t_j, \Pi(t_j)|_{v_1}] + d[t_j, \Pi(t_i)|_{v_1}]\} \\ & \quad \text{By combining Eqs. (8) and (10)} \\ &= |v_1| + \triangle e|_{v_1} \\ & \quad \text{By definition of } e \in D \\ &> |v_1| \\ & \quad \text{Because } \triangle e|_{v_1} > 0 \text{ by definition.} \end{aligned}$$

Hence the property. ∎

Property 2: ARG $\mathcal{G}$ is acyclic.

Proof : If $\delta(w, v) = v'$ for $w \in D^* - \{\varepsilon\}$, then by applying Property 1 recursively over $w \in D^*$, $|v'| > |v|$. Clearly, there is no string $x \in D^*$ such that $\delta(x, v') = v$, for it would contradict the fact that $|v'| > |v|$. Hence the property. ∎

Property 3: $V_o \neq \emptyset$ (i.e., given an arbitrary $v \in V$, $\exists w \in D^* : \delta(w, v) \in V_o$).

Proof : The property follows from Property 2 and the finiteness of $|V|$. ∎

B. Properties of Algorithm

We now present some basic properties of MA^3.

Theorem 1: MA^3 always terminates in a finite number of negotiation rounds.

Proof : Starting from an arbitrary state $v \in V$ of an ARG $\mathcal{G}$ that is not a terminal state, MA^3 traverses one max-transition $e \in D$ (as defined by Definition 4) in one round of negotiation such that by Property 3, it will eventually enter a terminal state $v_o \in V_o$. In a terminal state, a final round of negotiation proceeds during which the arbitration agent receives the lack of intentions by all agents and informs them to terminate negotiation. Hence the result. ∎

Theorem 2: Given a random initial assignment, MA^3 does not guarantee an optimal solution.

Proof : This can be established by a counter example. Consider the problem instance in Fig. 3(a). As shown in Fig. 3(b), this instance's ARG indicates that $V_o = \{v_4, v_5\}$, with $|v_4| = 18$ (optimal value) and $|v_5| = 17$. This means that if MA^3 should start at state $v_5 \in V$, it will end in the same state because it is a terminal state, but $v_5 \in V_o$ does not represent an optimal solution because $|v_5| < |v_4|$. Hence the result. ∎

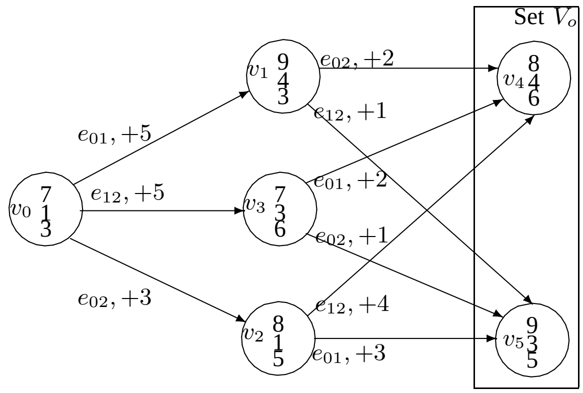
In general, MA^3 cannot guarantee optimality of a solution it reaches since the terminal state set V_o for a problem's ARG can possibly have more than one element of *different values*, and a string of max-transitions x the algorithm traces from an arbitrary state $v \in V$ cannot guarantee leading to a state $\delta(x, v) \in V_o$ which is the optimal.

Theorem 3: The number of negotiation rounds that MA^3 takes to converge to a solution depends on the initial assignment.

Proof : From an arbitrary initial solution state $v \in V$, MA^3 traverses a max-transition $e \in D$ (as defined by Definition 4) of a problem instance's ARG $\mathcal{G}$ following every negotiation round, until it reaches a terminal state $v_o \in V_o$. Thus, starting

	r_0	r_1	r_2
t_0	9	8	7
t_1	1	3	4
t_2	6	3	5

(a) Problem instance



(b) Problem's ARG

Fig. 3. An example that shows MA^3's no guarantee of optimality of solution

from different initial solution states (i.e., different initial assignments) may take different number of negotiation rounds to reach a terminal state. Hence the result. ■

To illustrate, consider a problem instance in Fig. 3(a). As shown in Fig. 3(b), if MA^3 starts from state $v_4 \in V$, it will take 2 negotiation rounds to reach terminal state $v_4 \in V$; if it starts from $v_0 \in V$, it will take 3 negotiation rounds to reach the same state.

V. EXPERIMENTAL SIMULATIONS

In this section, we present an empirical study of MA^3 to assess its performance. The performance is assessed primarily by the solution quality produced and the implementation-independent negotiation speed. The solution quality is measured (and graded) in terms of the various extents (in %) that a solution produced deviates from the optimal one, and the negotiation speed is measured in terms of the number of negotiation rounds needed to converge to a solution. The 'profile' of the performance is gathered from the various probabilities of interest defined, which include those of converging to these 'graded' solutions, and those of the algorithm running at various defined speed levels.

A. Simulation Results & Discussion

To conduct the study, we first prototyped a simulator for MA^3. The simulator consists of a centralized program running on an Intel® Pentium® personal computer with a 1.6GHz CPU and 128MB (RAM) memory. For an $N \times N$ problem instance, the program generates and inputs each of the $N!$ initial assignment solutions to a reasoning mechanism which computes the agents' resource selections which would have resulted from the distributed agent algorithm MA^3. The centralization is aimed at simplifying the code that, importantly, automates and speeds up the experimental (running and data collection) process, but with all the features of the original algorithm retained, except for its distribution.

In principle, MA^3 can handle an arbitrary problem size N. But for a complete simulation, the number of simulation runs required is $N!$ per problem instance. Clearly for a big N, it can become intractably time consuming to simulate for a large number of problem instances. For experimental purposes, we limit $N = 10$, requiring 10! (or 3,628,800) simulation runs per problem instance. This was manageable when we ran the simulator prototype for 100 *randomly generated* 10×10 problem instances. Despite this limit, we note that the simulation results can also provide a base reference for addressing large problem instances decomposed into smaller subproblems for MA^3. Problem decomposition, however, is usually done based on application-specific criteria that are beyond the scope of this report.

For all the simulation results tabulated in this report, the average value of each variable Z, $Z \in \{\epsilon_{wc}, n_{max}, P_x, P_{wc}, P_{vhi}, P_{hi}, P_{lo}\}$, denoted $\overline{Z}$, is computed using the respective formulae in Appendix . For the definition of each variable Z, see Appendix .

Table II presents the average empirical results based on 100 10×10 problem instances. The experimental results reported herein update and extend the preliminary ones presented in the conference version [6] of this report, which were generated based only on several hundreds of initial assignments.

Notice that the ratio of the maximum number of negotiation rounds n_{max} (required to produce a solution) and the problem size 10 approximated to 1.5. We also see that the worst-case solutions

TABLE II

MA^3 : EXPERIMENTAL RESULTS (AVERAGES)

Worst-Case		Solution Quality						Negotiation Speed	
ϵ_{wc}	n_{max}	P_0	P_5	P_{10}	P_{15}	P_{20}	P_{wc}	P_{hi}	P_{lo}
11.199 %	15	0.2702	0.8974	0.9973	1.0000	1.0000	0.0019	0.1078	0.0426

produced were *good enough*[2] in the sense that they were either equal to or within 20% of the optimal, as verified against the solutions produced by an existing centralized algorithm [5]. In fact, for all the problem instances, the probability of converging to a solution that is within 10% of the optimal was a high score of over 97%. Besides, the probability of converging to a worst-case solution was very low, and so were the respective probabilities of convergence at high speed and at low speed. The latter probabilities, i.e., P_{hi} and P_{lo}, mean that there was a high probability of running at intermediate speed, i.e., in X_m negotiation rounds, where $0.5N < X_m \leq N$ for $N = 10$.

On the average, the worst-case solution produced was definitely good enough as its deviation from the optimal was a good average of $11.199\% \leq 20\%$. From these simulations, we infer that, given a $N \times N$ CLAP instance where $2 \leq N \leq 10$, MA^3 will *certainly* converge to a good enough solution. It will *almost certainly* run at intermediate speed, regardless of its efficient implementation, whenever the initial assignment is randomly selected. How the negotiation can be sped up (without degrading solution quality) is an issue for future work.

VI. RELATED WORK

A. BDI Models

Among the agent architectures/models (see [2, Ch. 1]), the BDI model [7], [8] is one of the best known and studied model of practical reasoning. Based on a philosophical model of human practical reasoning, originally developed by M. Bratman [9], the basic model guides us to develop an agent to decide moment by moment which action to take in the furtherance of a goal. We adapt this model, motivated by its appropriateness in allowing us to conceptualize and metaphorically describe an agent's reasoning mechanism, moment by moment, in terms of the agent' mental attitudes B, D and I to solve CLAP. However, two aspects clearly differentiate our work from existing BDI models. In the first is our approach to modelling. Existing BDI models are developed without concisely formulating the problems they attempt to solve while in our work, the BDI model is developed with a clear formulation of the problem it addresses, namely, CLAP. In the second, each *moment* is not a moment of reasoning in reaction to changes in its environment, but a negotiation round of collaborative reasoning - in fact, existing BDI models give no architectural consideration to explicitly multiagent aspects of behaviour [10] that is essential for addressing CLAP.

B. The Auction Algorithm

Deserving special mention for applying a different metaphor of negotiation to the assignment problem is the auction algorithm [11]. Each iteration in auctioning proceeds as follows:

- Bid on behalf of the persons (or tasks in our context) for objects (or resources in our context). The bid of each person is the *object with the highest net value*. The net value is the magnitude of the difference between the benefit (or A-QoS value in our context) of assigning the object to this person and the object's latest price.
- Assign each object to a *person with the highest bid for it*, after unassigning the object from a different person (assigned with it at the start of the iteration), and adjust its price accordingly.

The algorithm terminates once every object has been bidded for at least once; for actual details, see [11]. There have been several multiprocessor- and shared memory-based implementations [3] of the algorithm to exploit the inherently parallelizable phases of bidding and assigning. But, to the best of our knowledge, there is no reported implementation of the algorithm as a system of software agents performing bidding and assigning to solve

[2]It seems reasonable to use 'good enough' as a qualitative reference here since, applying *Pareto's 80/20 rule*, 80% of CLAP applications can tolerate a 20% deviation from their optimal solutions.

the $N \times N$ CLAP. A multiagent implementation, if it exists, is at best a system of N task agents negotiating with N resource agents, possibly with a simple agent keeping track of the bid receipt of each resource agent and informing all agents when the termination condition holds. This contrasts with our proposed algorithm of N task agents negotiating among themselves for N resources in the presence of an arbitration agent.

C. Automated Negotiation

In distributed problem solving, negotiation is one way that agents use in cooperation to solve a problem. The MA^3 task agents are said to be in negotiation since they work out, communicatively, to reach an agreement that is acceptable by all agents; this form of negotiation is said to be *collaborative* since the agreement that is reached is due to every task agent's attempt in reasoning to contribute to achieving their social (sum-total maximization) goal.

In general, an automated means (or reasoning mechanism) of negotiation for an agent can be implemented as a *negotiation protocol* within which a *decision making model (or logic)* over the *objects of negotiation* resides. In the literature on general negotiation frameworks, agents that can negotiate with exact knowledge of each other's cost and utility functions, or such knowledge learnt in the initial step of interaction, have been proposed [12], [13]. There are agents that negotiate using the unified negotiation protocol in worth-, state-, and task-driven domains where agents look for mutually beneficial deals to perform task distribution [14], [15]. In negotiation via argumentation (NVA), the agents negotiate by sending each other proposals and counter-proposals. In [16], these proposals are accompanied by supporting arguments (explicit justifications) formulated as logical models. In [17], the distributed constraint satisfaction problem (DCSP) algorithm [18], [19] provides the computational model, extended with the supporting arguments (accompanying the proposals) formulated as local constraints. In [20], agents can conduct NVA in which an agent sends over its inference rules to its neighbour to demonstrate the soundness of its arguments. Finally, there are also negotiating agents that incorporate AI techniques (e.g. [21]) and auction mechanisms (e.g. [22]). For a good albeit not exhaustive review of agent negotiation, see [23].

The proposed MA^3 differs from existing work on negotiation in that it employs a new BDI negotiation model for CLAP (Section III).

D. Task Allocation

Also related are distributed approaches to task allocation [24], [25]. If agents are appointed as collaboration agents negotiating for tasks instead, then CLAP that they address becomes a task allocation problem which can be viewed as a special version of the task allocation problem formally stated in [24, p. 68, Definition 1]. In [25], the idea of *coalition formation* through which agents arrive at task allocation solutions by themselves has been utilized. Independently in [24], the idea of *contracting* to address the same problem has been used. Ours uses a different idea of *BDI reasoning*, and one direction for future work is to extend the basic foundation developed in this report to address the more general problem of task or resource allocation.

VII. Conclusions

This report has proposed a BDI negotiation model for the $N \times N$ CLAP, and investigated an algorithm MA^3 that embodies the BDI model concepts and offers a novel approximate solution to the problem.

This research should be of theoretical interest to multiagent researchers since to the best of our knowledge, this is perhaps the first effort that develops a BDI negotiation model with a clear formulation of the problem it addresses, namely, CLAP. The research should also be of practical interest to application researchers as the MA^3 proposed extends existing LAP algorithms from centralized processing to distributed agent reasoning. It is hoped that this report can shed new light on adopting agent approaches for solving traditional combinatorial problems in general, and inspire others to do so.

The proposed MA^3 does not guarantee an optimal solution unless the initial assignment set is *properly selected*. In general, whether an optimal solution can be obtained or not depends on the selection of the initial state (i.e., the initial assignment set). Future work includes developing heuristics to select an initial assignment set, on which MA^3 might produce a solution better than *good enough*, and in fewer rounds of negotiation.

APPENDIX

1) n : number of negotiation rounds.
2) n_{max} : maximum n, obtained when algorithm terminates, and is the largest of all n obtained from each of the $N!$ different initial assignments simulated for a $N \times N$ problem instance.
3) α : total A-QoS value (2).
4) α_{opt} : optimal α.
5) α_{wc} : worst-case α, obtained when algorithm terminates, and is the worst of all α's computed based on the simulation of $N!$ different initial assignments generated for a $N \times N$ problem instance.
6) ϵ : error, given by $\left(|\frac{\alpha-\alpha_{opt}}{\alpha_{opt}}| \times 100\%\right)$.
7) ϵ_{wc} : worst-case ϵ, where $\alpha = \alpha_{wc}$.
8) P_x : probability that an initial assignment can lead to a solution with total A-QoS α within $x\%$ of optimal, i.e., $\alpha \in [\frac{(100-x)}{100}\alpha_{opt}, \alpha_{opt}]$.
9) P_{wc} : probability that an initial assignment can lead to a worst-case solution, i.e., with $\alpha = \alpha_{wc}$.
10) P_{vhi} : probability that the algorithm runs at very high speed, i.e., the number of negotiation rounds it can take to reach a solution is not more than the greatest integer $\leq 0.3N$.
11) P_{hi} : probability that the algorithm runs at high speed, i.e., the number of negotiation rounds it can take to reach a solution is not more than the greatest integer $\leq 0.5N$.
12) P_{lo} : probability that the algorithm runs at low speed, i.e., the number of negotiation rounds it can take to reach a solution exceeds N.

All the probabilities of interest defined above are computed using formula $\left(\frac{\beta}{\gamma}\right)$, where integer β is the number of initial assignments satisfying the associated conditions *upon termination of algorithm*, and integer γ is the total number of different initial assignments input for simulation. $\gamma = N!$.

Let N_p' denote the sample size, label N-Y identify an $N \times N$ CLAP instance, and $Z[Y]$ refer to the value of variable Z for a $N \times N$ CLAP instance labelled N-Y. Then the averaging formulae used in Section V to determine the performance of a proposed algorithm on a 'per problem instance basis', are as follows: For $M \in \{\epsilon_{wc}, n_{max}, n_{hmax}\}$,

$$\overline{M} = \left(\frac{1}{N_p'}.\sum_{Y=1}^{N_p'} M[Y]\right), \text{ and } \quad \overline{P_j} = \left(\frac{1}{N_p'}.\sum_{Y=1}^{N_p'} P_j[Y]\right),$$

where $P_j[Y] = \left(\frac{\beta[Y]}{\gamma[Y]}\right)$, $j \in \{0, 5, 10, 15, 20, wc, vhi, hi, lo\}$.

Let N_p be the number of $N \times N$ problem instances used in the simulation. Then, in the formulae for M, $N_p' = N_p$; in the formula for P_j,

- if $j \in \{hi, lo\}$, $N_p' = N_p$;
- if $j \in \{0, 5, 10, 15, 20, wc\}$, $N_p' \leq N_p$ is the number of problem instances N-Y's for which $\alpha_{wc}[Y] < \alpha_{opt}[Y]$. It is more representative to average the $P_j[Y]$'s of instances N-Y's for which several different solutions other than the optimal one can be obtained via extensive simulation of its $N!$ possible initial assignments. In our simulations, for these P_j's, $N_p = 100$, but $N_p' = 99$; this is because out of the 100 randomly generated 10×10 problem instances, the instance 10-3 has $\alpha_{opt}[3] = \alpha_{wc}[3]$.

REFERENCES

[1] M. Silva, A. Giua, and J. M. Colom, Eds., *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004)*. IEEE Computer Society, USA, July 2004.

[2] G. Weiss, Ed., *Multiagent System : A Modern Approach to Distributed Artificial Intelligence*. The MIT Press, London, U.K, 1999.

[3] R. E. Burkard and E. Cela, "Linear assignment problems and extensions," in *Handbook of Combinatorial Optimization, Vol.4*, P. M. Pardalos and D. Z. Du, Eds. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1999, pp. 75–149.

[4] N. R. Jennings, K. Sycara, and M. Wooldridge, "A roadmap of agent research and development," *Autonomous Agents and Multi-Agent Systems*, vol. 1, no. 1, pp. 7–38, 1998.

[5] R. Jonker and A. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," *Computing*, vol. 38, no. -, pp. 325–340, 1987.

[6] K. T. Seow and K. Y. How, "Collaborative assignment : A multiagent negotiation approach using BDI concepts," in *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'02)*, Palazzo Re Enzo, Bologna, Italy, July 2002, pp. 256–263.

[7] M. E. Bratman, D. J. Israel, and M. E. Pollack, "Plans and resource-bounded practical reasoning," *Computational Intelligence*, vol. 4, no. 4, pp. 349–355, 1988.

[8] M. P. Georgeff and A. L. Lansky, "Reactive reasoning and planning," in *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI'87)*, Seattle, WA, USA, July 1987, pp. 677–682.

[9] M. E. Bratman, *Intentions, Plans and Practical Reason*. Harvard University Press, Cambridge, M.A, 1987.

[10] M. P. Georgeff, B. Pell, M. E. Pollack, M. Tambe, and M. Wooldridge, "The Belief-Desire-Intention model of agency," in *Intelligent Agents V: Agent Theories, Architectures, and Languages (ATAL): Proceedings of the 5th International Workshop, ATAL'98*, J. P. Müller and A. S. R. Munindar Singh, Eds. Paris, France: Springer Verlag, London, UK, July 1998, pp. 1–10.

[11] D. P. Bertsekas, "The auction algorithm : A distributed relaxation method for the assignment problem," *Annals of Operations Research*, vol. 14, pp. 105–123, 1988.

[12] S. Kraus, “Beliefs, time and incomplete information in multiple encounter negotiations among autonomous agents,” *Annals of Mathematics and Artificial Intelligence*, vol. 20, no. 1-4, pp. 111–159, 1997.

[13] S. Kraus, J. Wilkenfeld, and G. Zlotkin, “Multiagent negotiation under time constraints,” *Artificial Intelligence*, vol. 75, no. 2, pp. 297–345, 1995.

[14] J. S. Rosenschein and G. Zlotkin, “Designing conventions for automated negotiation,” *AI Magazine*, vol. 15, no. 3, pp. 29–46, 1994.

[15] G. Zlotkin and J. S. Rosenschein, “Mechanism design for automated negotiation, and its application to task oriented domains,” *Artificial Intelligence*, vol. 86, no. 2, pp. 195–244, 1996.

[16] S. Kraus, K. Sycara, and A. Evenchik, “Reaching agreements through argumentation : a logical model and implementation,” *Artificial Intelligence*, vol. 104, no. 1-2, pp. 1–69, 1998.

[17] H. Jung, M. Tambe, and S. Kulkarni, “Argumentation as distributed constraint satisfaction : Applications and results,” in *Proceedings of the 5th International Conference on Autonomous Agents*. Montreal, Quebec, Canada: ACM Press, July 2001, pp. 324–331.

[18] M. Yokoo, E. H. Durfee, T. Ishida, and K. Kuwabara, “The distributed constraint satisfaction problem : Formalization and algorithms,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 10, no. 5, pp. 673–685, September/October 1998.

[19] M. Yokoo and K. Hirayama, “Distributed constraint satisfaction algorithm for complex local problems,” in *Proceedings of the Third International Conference on Multiagent Systems (ICMAS-98)*, 1998, pp. 372–379.

[20] C. Sierra, N. R. Jennings, P. Noriega, and S. Parsons, “A framework for argumentation-based negotiation,” in *Proceedings of The Fourth International Workshop on Agent Theories, Architectures and Languages*, Rhode Island, USA, 1997, pp. 167–182.

[21] A. Chavez and P. Maes, “Kasbah : An agent marketplace for buying and selling goods,” in *Proceedings of the First International Conference on Practical Application of Intelligent Agents and Multi-Agent Technology*, London, UK, April 1996, pp. 75–90.

[22] C. Boutilier, M. Goldszmidt, and B. Sabata, “Sequential auctions for the allocation of resources with complementarities,” in *Proceedings of the 8th International Joint Conference on Artificial Inteligence*, 1999, pp. 527–534.

[23] N. R. Jennings, P. Faratin, A. R. Lomuscio, S. Parsons, C. Sierra, and M. Wooldridge, “Automated negotiation : Prospect, methods and challenges,” *International Journal of Group Decision and Negotiation*, vol. 10, no. 2, pp. 199–215, 2001.

[24] T. W. Sandholm, “Contract types for satisficing task allocation: I theoretical results,” in *Proceedings of AAAI Spring Symposium Series: Satisficing Models*, Sanford University, Stanford, CA, USA, March 1998, pp. 68–75.

[25] O. Shehory and S. Kraus, “Task allocation via coalition formation among autonomous agents,” in *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, Montreal, Canada, August 1995, pp. 655–661.