Issues in Automated Negotiation and Electronic Commerce: Extending the Contract Net Framework

Tuomas Sandholm and Victor Lesser *
{sandholm, lesser}@cs.umass.edu
University of Massachusetts at Amherst
Computer Science Department
Amherst, MA 01003

Abstract

In this paper we discuss a number of previously unaddressed issues that arise in automated negotiation among self-interested agents whose rationality is bounded by computational complexity. These issues are presented in the context of iterative task allocation negotiations. First, the reasons why such agents need to be able to choose the stage and level of commitment dynamically are identified. A protocol that allows such choices through conditional commitment breaking penalties is presented. Next, the implications of bounded rationality are analyzed. Several tradeoffs between allocated computation and negotiation benefits and risk are enumerated, and the necessity of explicit local deliberation control is substantiated. Techniques for linking negotiation items and multiagent contracts are presented as methods for escaping local optima in the task allocation process. Implementing both methods among self-interested bounded rational agents is discussed. Finally, the problem of message congestion among self-interested agents is described, and alternative remedies are presented.

1 Introduction

The importance of automated negotiation systems is likely to increase [Office of Technology Assesment (OTA), 1994]. One reason is the growth of a fast and inexpensive standardized communication infrastructure (EDI, NII, KQML [Finin *et al.*, 1992], Telescript [General Magic, Inc., 1994] etc.), over which separately designed agents belonging to different organizations can interact in an open environment in real-time, and safely carry out transactions [Kristol *et al.*, 1994; Sandholm and Lesser, 1995d]. Secondly, there is an industrial trend towards *agile enterprises*: small, organizational overhead avoiding enterprises that form short term alliances to be able to respond to larger and more diverse orders than they individually could. Such ventures can take advantage of economies of scale when they are available, but do not suffer from diseconomies of scale. This concept paper explores the implications of performing such negotiations where agents are *self-interested* (SI) [1] and must make negotiation decisions in real-time with *bounded or costly computation resources.*

We cast such negotiations in the following domain independent framework. Each agent has a (possibly empty) set of tasks and a (possibly empty) set of resources it can use to handle tasks. These sets change due to domain events, e.g. new tasks arriving or resources breaking down. The agents can subcontract tasks to other agents by paying a compensation. This subcontracting process can involve breaking a task into a number of subtasks handled by different agents, or clustering a number of tasks into a supertask. A task transfer is profitable from the global perspective if the contractee can handle the task less expensively than the contractor, or if the contractor cannot handle it at all, but the contractee can. So, the problem has two levels: a *global task allocation problem*, and each agent's *local combinatorial optimization problem* defined by the agent's current tasks and resources. The goal of each agent is to maximize its *payoff* which is defined as its income minus its costs. Income is received for handling tasks, and costs are incurred by using resources to handle the tasks. We restrict ourselves to domains where the feasibility and cost of handling a task do not depend on what other agents do with their resources or how they divide tasks among themselves, but do depend on the other tasks that the agent has [2]. The global solution can be evaluated from a social welfare viewpoint according to the sum of the agents' payoffs.

Reaching good solutions for the global task allocation problem is difficult with SI agents, e.g. because they may not truthfully share all information. The problem is further complicated by the agents' bounded rationality: local decisions are suboptimal due to the inability

*This research was supported by ARPA contract N00014-92-J-1698. The content does not necessarily reflect the position or the policy of the Government and no official endorsement should be inferred. T. Sandholm also funded by a University of Massachusetts Graduate School Fellowship, Leo and Regina Wainstein Foundation, Heikki and Hilma Honkanen Foundation, and Ella and George Ehrnrooth Foundation.

[1] In domains where agents represent different real world organizations, each agent designer will want its agent to do as well as it can without concern for other agents. Conversely, some domains are inherently composed of benevolent agents. For example, in a single factory scheduling problem, each work cell can be represented by an agent. If the cells do not have private goals, the agents should act benevolently.

[2] Such domains are a superset of what [Rosenschein and Zlotkin, 1994] call Task Oriented Domains, and intersect their State Oriented and Worth Oriented Domains.

to precisely compute the value associated with accepting a task. This computation is especially hard if the feasibility and cost of handling a task depend on what other tasks an agent has. These problems are exacerbated by the uncertainty of an open environment in which new agents and new tasks arrive - thus previous decisions may be suboptimal in light of new information.

The original contract net protocol (CNP) [Smith, 1980] did not explicitly deal with these issues, which we think must be taken into account if agents are to operate effectively in a wide range of automated negotiation domains. A first step towards extending the CNP to deal with these issues was the work on TRACONET [Sandholm, 1993]. It provided a formal model for bounded rational (BR) self-interested agents to make announcing, bidding and awarding decisions. It used a simple static approximation scheme for *marginal cost*[3] calculation to make these decisions. The choice of a contractee is based solely on these marginal cost estimates. The monetary payment mechanism allows quantitative tradeoffs between alternatives in an agent's negotiation strategy. Within DAI, bounded rationality (approximate processing) has been studied with cooperative agents, but among SI agents, perfect rationality has been widely assumed, e.g. [Rosenschein and Zlotkin, 1994; Ephrati and Rosenschein, 1991; Kraus *et al.*, 1992]. We argue that in most real multiagent applications, resource-bounded computation will be an issue, and that bounded rationality has profound implications on both negotiation protocols and strategies.

Although the work on TRACONET was a first step towards this end, it is necessary—as discussed in the body of this paper—to extend in significant ways the CNP in order for bounded rational self-interested (BRSI) agents to deal intelligently with uncertainty present in the negotiation process. This new protocol represents a family of different protocols in which agents can choose different options depending on both the static and dynamic context of the negotiation. The first option we will discuss regards commitment. We present ways of varying the stage of commitment, and more importantly, how to implement *varying levels of commitment* that allow more flexible local deliberation and a wider variety of negotiation risk management techniques by allowing agents to back out of contracts. The second option concerns local deliberation. Tradeoffs are presented between negotiation risks and computation costs, and an approximation scheme for marginal cost calculation is suggested that dynamically adapts to an agent's negotiation state. The third set of options has to do with avoiding local optima in the task allocation space by *linking negotiation items* and by *contracts involving multiple agents.* The fourth set of options concerns *message congestion management.* We present these choices in terms of a new protocol for negotiation among BRSI agents, that, to our knowledge, subsumes the CNP and most—if not all—of its extensions.

[3]The *marginal cost* of adding a set of tasks to an agent's solution is the cost of the agent's solution with the new task set minus the cost of the agent's solution without it.

2 Commitment in negotiation protocols

2.1 Alternative commitment stages

In mutual negotiations, *commitment* means that one agent binds itself to a potential contract while waiting for the other agent to either accept or reject its offer. If the other party accepts, both parties are bound to the contract. When accepting, the second party is sure that the contract will be made, but the first party has to commit before it is sure. Commitment has to take place at some stage for contracts to take place, but the choice of this stage can be varied. TRACONET was designed so that commitment took place in the bidding phase as is usual in the real world: if a task is awarded to him, the bidder has to take care of it at the price mentioned in the bid. Shorter protocols (commitment at the announcement phase[4]) can be constructed as well as arbitrarily long ones (commitment at the awarding phase or some later stage).

The choice of commitment stage can be a static protocol design decision or the agents can decide on it dynamically. For example, the focused addressing scheme of the CNP was implemented so that in low utilization situations, contractors announced tasks, but in high utilization mode, potential contractees signaled availability—i.e. bid without receiving announcements first [Smith, 1980; Van Dyke Parunak, 1987]. So, the choice of a protocol was based on characteristics of the environment. Alternatively, the choice can be made for each negotiation separately before that negotiation begins. We advocate a more refined alternative, where agents dynamically choose the stage of commitment of a certain negotiation during that negotiation. This allows any of the above alternatives, but makes the stage of commitment a negotiation strategy decision, not a protocol design decision. The offered commitments are specified in *contractor messages* and *contractee messages*, Fig. 1.

2.2 Levels of commitment

In traditional multiagent negotiation protocols among SI agents, once a contract is made, it is binding, i.e. neither party can back out. In cooperative distributed problem solving (CDPS), commitments are often allowed to be broken unilaterally based on some local reasoning that attempts to incorporate the perspective of common good [Decker and Lesser, 1995]. A more general alternative is to use protocols with continuous levels of commitment based on a monetary penalty method, where commitments vary from unbreakable to breakable as a continuum by assigning a commitment breaking cost to each commitment separately. This cost can also increase with time, decrease as a function of acceptance time of the offer, or be conditioned on events in other negotiations or the environment. Using the suggested message types, the level of commitment can also be dynamically negotiated over on a per contract or per task set basis.

[4]With announcement phase commitment, a task set can be announced to only one potential bidder at a time, since the same task set cannot be exclusively awarded to many agents.

Among other things, the use of multiple levels of commitment allows:

- a low commitment search focus to be moved around in the global task allocation space (because decommitting is not unreasonably expensive), so that more of that space can be explored among SI agents which would otherwise avoid risky commitments[5],
- flexibility to the agent's local deliberation control, because marginal cost calculation of a contract can go on even after that contract has already been agreed upon,
- an agent to make the same low-commitment offer (or offers that overlap in task sets) to multiple agents. In case more than one accepts, the agent has to pay the penalty to all but one of them, but the speedup of being able to address multiple agents in committal mode may outweigh this risk,
- the agents with a lesser risk aversion to carry a greater portion of the risk. The more risk averse agent can trade off paying a higher price to its contractee (or get paid a lower price as a contractee) for being allowed to have a lower decommitting penalty, and
- contingency contracts by conditioning the payments and commitment functions on future negotiation events or domain events. These enlarge the set of mutually beneficial contracts, when agents have different expectations of future events or different risk attitudes [Raiffa, 1982].

The advantages of such a leveled commitment protocol are formally analyzed in [Sandholm and Lesser, 1995a], and are now reviewed. Because the decommitment penalties can be set arbitrarily high for both agents, the leveled commitment protocol can always emulate the full commitment protocol. Furthermore, there are cases where there is no full commitment contract among two agents that fulfills the participation constraints (agent prefers to agree to the contract as opposed to passing) for both agents, but where a leveled commitment contract does fulfill these constraints. This occurs even among risk neutral agents, for example when uncertainty prevails regarding both agents' future offers received, and both agents are assigned a (not too high or low, and not necessarily identical) decommitment penalty in the contract. Among risk neutral agents, this does not occur if only one of the agents is allowed the possibility to decommit (other agent's decommitment penalty is too high), or only one agent's future is uncertain. If the agents have biased information regarding the future, they may perceive that such a contract with a one-sided decommitment possibility is viable although a full commitment contract is not. In such cases, the agent whose information is biased is likely to take the associated loss while the agent with unbiased information is not.

Figure 1 describes the message formats of the new contracting protocol. A negotiation can start with either a contractor or a contractee message, Fig. 2. A contractor message specifies exclusive alternative contracts that the contractor is willing to commit to. Within each alternative, the tasks can be split into disjoint task sets by the sender of the message in order for the fields (a) - (d) to be specific for each such task set - not necessarily the whole set of tasks. Each alternative has the following semantics. If the contractee agrees to handle all the task sets in a manner satisfying the minimum required task descriptions (a) (which specify the tasks and constraints on them, e.g. latest and earliest handling time or minimum handling quality), and the contractee agrees to commit to each task set with the level specified in field (d), then the contractor is automatically committed to paying[6] the amounts of fields (b), and can cancel the deal on a task set only by paying the contractee a penalty (c)[7]. Moreover, the contractor is decommitted

```
CONTRACTOR MESSAGE:
0. Negotiation identifier
1. Message identifier
2. In-response-to (message id)
3. Sender
4. Receiver
5. Terminate negotiation
6. Alternative 1
  6.1. Time valid through
  6.2. Bind after partner's decommit
  6.3. Offer submission fee
  6.4. Required response submission fee
  6.5. Task set 1
    (a) (Minimum) specification of tasks
    (b) Promised payment fn. to contractee
    (c) Contractor's promised commitment fn.
    (d) Contractee's required commitment fn.
  6.6. Task set 2
      ...
  6.i. Task set i-4
7. Alternative 2
    ...
j. Alternative j-5
```

```
CONTRACTEE MESSAGE:
0. Negotiation identifier
1. Message identifier
2. In-response-to (message id)
3. Sender
4. Receiver
5. Terminate negotiation
6. Alternative 1
  6.1. Time valid through
  6.2. Bind after partner's decommit
  6.3. Offer submission fee
  6.4. Required response submission fee
  6.5. Task set 1
    (a) (Maximum) specification of tasks
    (b) Required payment fn. to contractee
    (c) Contractor's required commitment fn.
    (d) Contractee's promised commitment fn.
  6.6. Task set 2
      ...
  6.m. Task set m-4
7. Alternative 2
    ...
n. Alternative n-5
```

```
PAYMENT/DECOMMIT MESSAGE:
0. Negotiation id
1. Message id
2. Accepted offer id
3. Acceptance message id
4. Sender
5. Receiver
6. Message type
      (payment/decommit)
7. Money transfer
```

Figure 1: *Contracting messages of a single negotiation.*

[5]For example, an agent can accept a task set and later try to contract the tasks in that set further separately. With full commitment, an agent needs to have standing offers from the agents it will contract the tasks to, or it has to be able to handle them itself. With the variable commitment protocol, the agent can accept the task set even if it is not sure about its chances of getting it handled, because in the worst case it can decommit.

[6]Secure money transfer can be implemented cryptographically e.g. by electronic credit cards or electronic cash [Kristol *et al.*, 1994].

[7]The "Bind after partner's decommit" (6.2) flag describes whether an offer on an alternative will stay valid according to its original deadline (field 6.1) even in the case where the contract was agreed to, but the partner decommitted by paying the decommitment penalty.

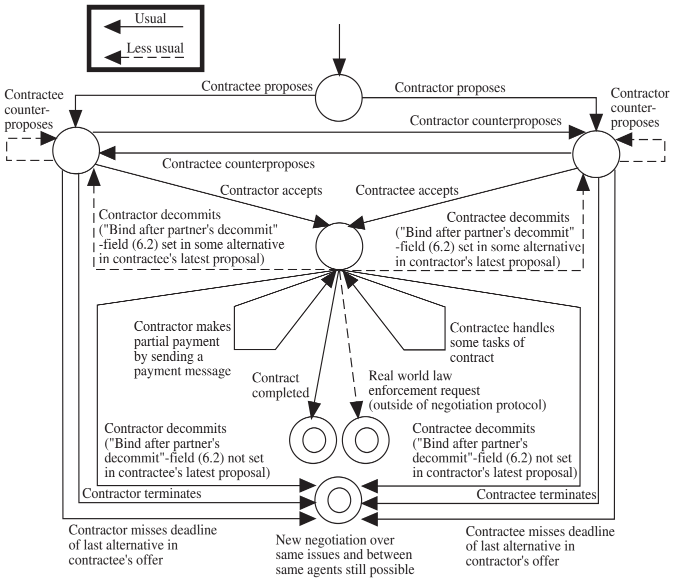

Figure 2: *State transition diagram of a single negotiation.*

from all the other alternatives it suggested[8]. If the contractee does not accept any of the alternatives, the contractor is decommitted from all of them. Fields (b), (c) and (d) can be functions of time, of negotiation events, or of domain events, and these times/events have to be observable or verifiable by both the contractor and the contractee. A contractee can accept one of the alternatives of a contractor message by sending a contractee message that has task specifications that meet the minimal requirements (a), and payment functions that meet the required payment functions (b), and commitment functions (c) for the contractee that meet the required commitment functions, and commitment functions (d) for the contractor that do not exceed the contractor's promised commitment. A contractor message can accept one of the alternatives of a contractee message analogously. An agent can entirely terminate a negotiation by sending a message with that negotiation's identifier (field 0), and the terminate-flag (field 5) set.

Alternatively, the contractee can send a contractee message that neither accepts the contractor message (i.e. does not satisfy the requirements) nor terminates the negotiation. Such a message is a *counterproposal*, which the contractor then can accept, terminate the negotiation, or further counterpropose etc. *ad infinitum* [9]. The CNP did not allow counterproposing: an agent could bid to an announcement or decide not to bid. A contractor had the option to award or not to award the tasks according to the bids. Counterproposing among cooperative agents was studied in [Moehlman *et al.*, 1992; Sen, 1993]. Our counterproposing mechanism is one way of overcoming the problem of lacking truthful abstractions of the global search space (defined by the task sets and resource sets of all the agents) in negotiation systems consisting of SI agents.

There are no uncommittal messages such as announcements used to declare tasks: all messages have some commitment specification for the sender. In early messages in a negotiation, these commitment specifications can be too low for the partner to accept, and counterproposing occurs. Thus, the level and stage of commitment are dynamically negotiated along with the negotiation of taking care of tasks.

The presented negotiation protocol is a strict generalization of the CNP, and can thus always emulate it. Moreover, there are cases where this protocol is better than the CNP—due to reasons listed earlier. Yet, the development of appropriate negotiation *strategies* for this protocol is challenging—e.g. how should an agent choose commitment functions and payment functions?

2.3 Decommitting: replies vs. timeouts

The (6.1) field describes how long an offer on an alternative is valid. If the negotiation partner has not answered by that time, the sender of the message gets decommitted from that alternative. An alternative to these strict deadlines is to send messages that have the (b) field be a function of the time of response (similarly for (c) and (d) fields). This allows a contractor to describe a payment that decreases as the acceptance of the contractor message is postponed. Similarly, it allows a contractee to specify required payments that increase as the acceptance of the contractee message is postponed. This motivates the negotiation partner to respond quickly, but does not force a strict deadline, which can inefficiently constrain that agent's local deliberation scheduling. Both the strict deadline mechanism and this time-dependent payment scheme require that the sending or receival time of a message can be verified by both parties.

An alternative to automatic decommitment by the deadline is to have the negotiation partner send a negative reply (negotiation termination message) by the deadline. These forced response messages are not viable among SI agents, because an agent that has decided not to accept or counterpropose has no reason to send a reply. Sending reply messages also in negative cases allows the offering agent to decommit before the validity time of its offer ends. This frees that agent from considering the effects of the possible acceptance of that offer on the marginal costs of other task sets that the agent is negotiating over. This saved computation can be used to negotiate faster on other contracts. Thus, an agent considering sending a negative reply may want to send it in cases where the offering agent is mostly negotiating with that agent, but not in cases, where the offering agent is that agent's competing offerer in most other negotiations.

[8] Another protocol would have offers stay valid according to their original specification (deadline) no matter whether the partner accepts, rejects, counterproposes, or does none of these. We do not use such protocols due to the harmfully (Sec. 3) growing number of pending commitments.

[9] An agent that has just (counter)proposed can counterpropose again (dotted lines in Fig. 2). This allows it to add new offers (that share the "In-response-to"-field with the pending ones), but does not allow retraction of old offers. Retraction is problematic in a distributed system, because the negotiation partner's acceptance message may be on the way while the agent sends the retraction.

3 Implications of bounded rationality

Interactions of SI agents have been widely studied in microeconomics [Kreps, 1990; Varian, 1992; Raiffa, 1982] and DAI [Rosenschein and Zlotkin, 1994; Ephrati and Rosenschein, 1991; Kraus *et al.*, 1992; Durfee *et al.*, 1993], but perfect rationality of the agents has usually been assumed: flawless deduction, optimal reasoning about future contingencies and recursive modeling of other agents. Perfect rationality implies that agents can compute their marginal costs for tasks exactly and immediately, which is untrue in most practical situations. An agent is bounded rational, because its computation resources are costly, or they are bounded and the environment keeps changing—e.g. new tasks arrive and there is a bounded amount of time before each part of the solution is used [Garvey and Lesser, 1994; Sandholm and Lesser, 1994; Zilberstein, 1993; Simon, 1982; Good, 1971]. Contracting agents have the following additional real-time pressures:

- A counteroffer or an acceptance message has to be sent by a deadline (field 6.1) - otherwise the negotiation terminates, Fig. 2. If the negotiation terminates, the agent can begin a new negotiation on the same issues, but it will not have the other agent's commitment at first.
- Sending an outgoing offer too late may cause the receiving agent to make a contract on some of the same tasks with some other agent who negotiated earlier—thus disabling this contract even if the offer makes the deadline. In case this deadline abiding offer is an acceptance message—as opposed to a counteroffer—the partner has to pay the decommitment penalty that it had declared.
- The (b)-(d) fields can be functions of response time, Fig. 1. An agent may get paid less for handling tasks (or pay more for having tasks handled) or be required to commit more strongly or receive a weaker commitment from the negotiation partner if its response is postponed.
- The agent's cost of breaking commitments (after a contract is made) may increase with time.

This problem setup leads to a host of local deliberation scheduling issues. An agent has to decide *how much computation* it should allocate to refine its marginal cost estimate of a certain task set. With a bounded CPU, if too much time is allocated, another agent may win the contract before the reply is sent, or not enough time remains for refining marginal costs of other task sets. If too little time is allocated, the agent may make an unbeneficial contract concerning that task set. If multiple negotiations are allowed simultaneously, the agent has to decide on *which sets of tasks* (offered to it or potentially offered by it) its bounded computation should be focused—and *in what order*. It may want to ignore some of its contracting possibilities in order to focus more deliberation time to compute marginal costs for task sets of some selected potential contracts. So, there is a tradeoff of getting more exact marginal cost estimates and being able to engage in a larger number of negotiations.

The CNP did not consider an agent's risk attitude toward being committed to activities it may not be able to honor, or the honoring of which may turn out unbeneficial. In our protocol, an agent can take a risk by making offers while the acceptance of earlier offers is pending. Contracting during pending commitments speeds up the negotiations because an agent does not have to wait for results on earlier commitments before carrying on with other negotiations. The work on TRACONET formalized the questions of risk attitude in a 3-stage (announce-bid-award) full-commitment protocol, and chose a risk taking strategy where each agent ignored the chances of pending commitments being accepted in order to avoid computations regarding these alternative future worlds. This choice was static, but more advanced agents should use a risk taking strategy where negotiation risk is explicitly traded off against added computation regarding the marginal cost of the task set in the alternative worlds, where different combinations of sent pending offers are accepted.

There is a tradeoff between accepting or (counter) proposing early on and waiting:

- A better offer may be received later.
- Waiting for more simultaneously valid offers enables an agent to identify and accept synergic ones: having more options available at the decision point enables an agent to make more informed decisions.
- Accepting early on simplifies costly marginal cost computations, because there are fewer options to consider. An option corresponds to an item in the power set of offers that an agent can accept or make.
- By waiting an agent may miss opportunities due to others making related contracts first.

An agent should anticipate future negotiation and domain events in its strategy [Sandholm and Lesser, 1995b].[10] It suffices to take these events into account in marginal cost estimation: this will cause the agent to anticipate with its domain solution. The real marginal cost of a task set is the difference in the *streams of payments and domain costs* when an agent has the task set and when the agent does not have it. This marginal cost does not necessarily equal the cost that is acquired statically at contract time (before the realization of unknown future negotiation events and domain events) by taking the difference of the cost of the agent's optimal solution with the task set and the optimal solution without it. Furthermore, for BR agents, the marginal cost may change as more computation is allocated to the solution including the task set or the solution without it. In general, the marginal cost of a task set depends on which other tasks the agent has. Therefore, theoretically, the marginal cost of a task set has to be computed in all of the alternative future worlds, where different combinations of pending,

[10] The agent can believe that domain events occur to the agent society according to some distribution and that in steady state these events will affect (directly or by negotiation) the agent according to some distribution. E.g. the agent assumes that future tasks end up in its task set according to a distribution. On another level, an agent can try to outguess the other agents' solutions so that it can use the others marginal costs as a basis for its own marginal cost calculation. On a third level, the agent can model what another agent is guessing about yet another agent, and so on *ad infinitum*. There is a tradeoff between allocating costly computation resources to such recursive modeling and gaining domain advantage by enhanced anticipation.

to-be-sent, and to-be-received offers have been accepted, different combinations of old and to-occur contracts have been broken by decommitting (by the agent or its partners), and different combinations of domain events have occurred. Managing such contingencies formally using probability theory is intractable: costs of such computations should be explicitly traded off against the domain advantage they provide. An agent can safely ignore the chances of other agents decommitting only if the decommitment penalties are high enough to surely compensate for the agent's potential loss. Similarly, an agent has to ignore its decommitting possibilities if its penalties are too high. The exponential number of alternative worlds induced by decommitting options sometimes increases computational complexity more than the benefit from the gradual commitment scheme warrants. Moreover, the decommitting events are not independent: chains of decommitting complicate the management of decommitment probabilities. Thus, decommitment penalty functions that increase rapidly in time may often be appropriate for BR agents.

Because new events are constantly occurring, the deliberation control problem is stochastic. An agent should take the likelihood of these events into account in its deliberation scheduling. The performance profile of the local problem solving algorithm should be conditioned on features of the problem instance [Sandholm and Lesser, 1994], on performance on that instance so far [Sandholm and Lesser, 1994; Zilberstein, 1993], and on performance profiles of closely related optimizations (related calculations of marginal costs). These aspects make exact decision theoretic deliberation control infeasible: approximations are required. The need for this type of deliberation control has not, to our knowledge, been well understood, and analytically developing a domain independent control strategy that is instantiated separately (using statistical methods) for each domain would allow faster development of more efficient automated negotiators across multiple domains.

4 Linking negotiation items

In early CNP implementations, tasks were negotiated one at a time. This is insufficient, if the cost or feasibility of carrying out a task depend on the carrying out of other tasks: there may be local optima, where no transfer of a single task between agents enhances the global solution, but transferring a larger set of tasks simultaneously does. The need for larger transfers is well known in centralized iterative refinement optimization [Lin and Kernighan, 1971; Waters, 1987], but has been generally ignored in automated negotiation. TRACONET extended the CNP to handle task interactions by having the announcer *cluster tasks into sets to be negotiated atomically.* Alternatively, the bidder could have done the clustering by counterproposing. Our protocol generalizes this by allowing either party to do the clustering, Fig. 1, at any stage of the protocol.

The equivalent of large transfers can be accomplished by smaller ones if the agents are willing to take risks. Even if no small contract is individually beneficial, the agents can sequentially make all the small contracts that sum up to a large beneficial one. Early in this sequence, the global solution degrades until the later contracts enhance it. When making the early commitments, at least one of the two agents has to risk taking a permanent loss in case the partner does not agree to the later contracts. Our protocol decreases such risks as much as preferred by allowing breaking commitments by paying a penalty. The penalty function may be explicitly conditioned on the acceptance of the future contracts, or it may specify low commitment for a short time during which the agent expects to make the remaining contracts of the sequence.

Sometimes there is no task set size such that transferring such a set from one agent to another enhances the global solution. Yet, there may be a beneficial *swap* of tasks, where the first agent subcontracts some tasks to the second and the second subcontracts some to the first. Swaps can be explicitly implemented in a negotiation protocol by allowing some task sets in an alternative (Fig. 1) to specify tasks to contract in and some to specify tasks to contract out. In the task sets added to implement swaps, "Minimum" in field (a) should be changed to "Maximum" and vice versa. In field (b), "Promised payment fn. to contractee" should be changed to "Required payment fn. from contractee" and "Required payment fn. to contractee" should be changed to "Promised payment fn. from contractee". Alternatively, in protocols that do not explicitly incorporate swaps, they can be made by agents taking risks and constructing the swap as a sequence of one way task transfer contracts. Here too, the decommitment penalty functions can be conditioned on later contracts in the sequence or on time to reduce (or remove) risk.

5 Mutual vs. multiagent contracts

Negotiations may have reached a local optimum with respect to each agent's local search operators and mutual contract operators (transfers and swaps of any size), but solution enhancements would be possible if tasks were transferred among more than two agents, e.g. agent A subcontracts a task to C and B subcontracts a task to C. There are two main ways to implement such deals[11]:

1. Explicit multiagent contracts. These contract operators can be viewed as atomic operators in the global task allocation space. First, one agent (with an incomplete view of the other agents' tasks and resources) has to identify the beneficiality of a potential multiagent contract. Alternatively, the identification phase can be implemented in a distributed manner. Second, the protocol has to allow a multiagent contract. This can be done e.g. by circulating the contract message among the parties and agreeing that the contract becomes valid only if every agent signs.

2. Multiagent contracts through mutual contracts. A multiagent contract is equivalent to a sequence of mutual contracts. In cases where a local optimum with respect to mutual contracts has been reached,

[11] Sathi et al. [Sathi and Fox, 1989] did this by having a centralized mediator cluster several announcements and bids from multiple agents into atomic contracts. That is unreasonable if decentralization is desired.

the first mutual contracts in the sequence will incur losses. Thus, one or more agents have to incur risk in initially taking unbeneficial contracts in unsure anticipation of more than compensatory future contracts. Our protocol provides mechanisms for decreasing this risk, either by conditioning the decommitment penalty functions on whether the contracts with other agents take place, or by choosing the penalties to be low early on and increase with time. In the limit, the penalty is zero (theoretically possibly even negative) for all contracts in the sequence if some contract in it is not accepted. The problem with contingency contracts is just the monitoring of the events that the contract (penalty) is contingent on: how can the contractee monitor the contractor's events and vice versa?

Sometimes an agent can commit to an unprofitable early contract in the sequence without risk even with constant high decommitting penalties. E.g. if an agent has received committal offers on two contracts, it can accept both without risk—assuming that decommitment penalties for the two senders are so high that they will not decommit. Even though the agent may have some offers committed simultaneously, the likelihood of having all the necessary offers committed simultaneously decreases as the number of mutual contracts required in the multiagent contract increases. Sometimes there is a loop of agents in the sequence of mutual contracts, e.g. say that the only profitable operator is the following: agent A gives task 1 to agent B, agent B gives task 2 to agent C, and agent C gives task 3 to agent A. In such cases it is impossible to handle the multiagent contract as separate mutual contracts without risk (without tailoring the decommitment penalty functions). A negotiating agent should take the possibilities of such loops into account when estimating the probabilities of receiving certain tasks, because the very offering or accepting of a certain task may directly affect the likelihood of getting offers or acceptances for other tasks.

6 Message congestion: Tragedy of the commons

Most distributed implementations of automated contracting have run into message congestion problems [Smith, 1980; Van Dyke Parunak, 1987; Sandholm, 1993]. While an agent takes a long time to process a large number of received messages, even more messages have time to arrive, and there is a high risk that the agent will finally be saturated. Attempts to solve these problems include focused addressing [Smith, 1980], audience restrictions [Van Dyke Parunak, 1987; Sandholm, 1993] and ignoring incoming messages that are sufficiently outdated [Sandholm, 1993]. Focused addressing means that in highly constrained situations, agents with free resources announce availability, while in less constrained situations, agents with tasks announce tasks. This avoids announcing too many tasks in highly constrained situations, where these announcements would seldom lead to results. In less constrained environments, resources are plentiful compared to tasks, so announcing tasks focuses negotiations with fewer messages. Audience restrictions mean that an agent can only announce to a subset of agents which are supposedly most potential.

Focused addressing and audience restrictions are imposed on an agent by a central designer of the agent society. Neither is viable in open systems with SI agents. An agent will send a message whenever it is beneficial to itself even though this might saturate other agents. With flat rate media such as the Internet, an agent prefers sending to almost everyone who has non-zero probability of accepting/counterproposing. The society of agents would be better off by less congested communication links by restricted sending, but each agent sends as long as the expected utility from that message exceeds the decrease in utility to that agent caused by the congesting effect of that message in the media. This defines a *tragedy of the commons* [Turner, 1992; Hardin, 1968] (n-player prisoners' dilemma). The tragedy occurs only for low commitment messages (usually early in a negotiation): having multiple high commitment offers out simultaneously increases an agent's negotiation risk (Sec. 2.2) and computation costs (Sec. 3).

The obvious way to resolve the tragedy is a use-based communication charge. Another is mutual monitoring: an agent can monitor how often a certain other agent sends low commitment messages to it, and over-eager senders can be punished. By mutual monitoring, audience restrictions can also be implemented: if an agent receives an announcement although it is not in the appropriate audience, it can directly identify the sender as a violator. Our protocol allows an agent to determine in its offer (field 6.4) a processing fee that an accepting or counterproposing agent has to submit in its response (field 6.3) for the response to be processed. This implements a self-selecting dynamic audience restriction that is viable among SI agents.

7 Conclusions

We introduced a collection of issues that arise in automated negotiation systems consisting of BRSI agents. Reasons for dynamically chosen commitment stage and level were given and a protocol that enables this was presented. The need for explicit local deliberation scheduling was shown by tradeoffs between computation costs and negotiation benefits and risk. Linking negotiation items and multiagent contracts were presented as methods to avoid local optima in the global task allocation space, and their implementation among BRSI agents was discussed. Finally, message congestion mechanisms for SI agents were presented.

Negotiations among BRSI agents also involve other issues (detailed in [Sandholm and Lesser, 1995b] due to limited space here) such as: insufficiency of the Vickrey auction to promote truth-telling and stop counterspeculation, usefulness of long term strategic contracts, tradeoffs between enforced and unenforced contracts [Sandholm and Lesser, 1995d], and knowing when to terminate the negotiations when an optimum with respect to the current tasks and resources has been reached or when further negotiation overhead outweighs the associated benefits. Coalition formation among BRSI agents has been studied in [Sandholm and Lesser, 1995c].

References

[Decker and Lesser, 1995] Keith Decker and Victor R Lesser. Designing a family of coordination algorithms. In *1st International Conference on Multiagent Systems*, San Fransisco, CA, June 1995.

[Durfee *et al.*, 1993] Edmund H Durfee, J Lee, and Piotr J Gmytrasiewicz. Overeager reciprocal rationality and mixed strategy equilibria. In *AAAI-93*, pages 225–230, Washington DC, July 1993.

[Ephrati and Rosenschein, 1991] Eithan Ephrati and Jeffrey S Rosenschein. The clarke tax as a consensus mechanism among automated agents. In *AAAI*, pages 173–178, Anaheim, CA, 1991.

[Finin *et al.*, 1992] Tim Finin, Rich Fritzson, and Don McKay. A language and protocol to support intelligent agent interoperability. In *Proc. of the CE & CALS Washington '92 Conference*, June 1992.

[Garvey and Lesser, 1994] A Garvey and V Lesser. A survey of research in deliberative real-time artificial intelligence. *Real-Time Systems*, 6:317–347, 1994.

[General Magic, Inc., 1994] General Magic, Inc. Telescript technology: The foundation for the electronic marketplace, 1994. White paper.

[Good, 1971] Irving Good. Twenty-seven principles of rationality. In V Godambe and D Sprott, editors, *Foundations of Statistical Inference*. Toronto: Holt, Rinehart, Winston, 1971.

[Hardin, 1968] G Hardin. The tragedy of the commons. *Science*, 162:1243–1248, 1968.

[Kraus *et al.*, 1992] Sarit Kraus, Jonathan Wilkenfeld, and Gilad Zlotkin. Multiagent negotiation under time constraints. Univ. of Maryland, College Park, Computer Science TR-2975, 1992.

[Kreps, 1990] David M Kreps. *A course in microeconomic theory*. Princeton University Press, 1990.

[Kristol *et al.*, 1994] David M Kristol, Steven H Low, and Nicholas F Maxemchuk. Anonymous internet mercantile protocol. 1994. Submitted.

[Lin and Kernighan, 1971] S Lin and B W Kernighan. An effective heuristic procedure for the traveling salesman problem. *Operations Research*, 21:498–516, 1971.

[Moehlman *et al.*, 1992] T Moehlman, V Lesser, and B Buteau. Decentralized negotiation: An approach to the distributed planning problem. *Group Decision and Negotiation*, 2:161–191, 1992.

[Office of Technology Assesment (OTA), 1994] Office of Technology Assesment (OTA). Electronic enterprises: Looking to the future, 1994.

[Raiffa, 1982] H. Raiffa. *The Art and Science of Negotiation*. Harvard Univ. Press, Cambridge, Mass., 1982.

[Rosenschein and Zlotkin, 1994] Jeffrey S Rosenschein and G Zlotkin. *Rules of Encounter*. MIT Press, 1994.

[Sandholm and Lesser, 1994] Tuomas W Sandholm and Victor R Lesser. Utility-based termination of anytime algorithms. In *ECAI Workshop on Decision Theory for DAI Applications*, pages 88–99, Amsterdam, The Netherlands, 1994. Extended version: Univ. of Mass. at Amherst, Comp. Sci. Tech. Report 94-54.

[Sandholm and Lesser, 1995a] Tuomas W Sandholm and Victor R Lesser. Advantages of a leveled commitment contracting protocol. Univ. of Mass. at Amherst, Comp. Sci. Tech. Report, 1995. In preparation.

[Sandholm and Lesser, 1995b] Tuomas W Sandholm and Victor R Lesser. Automated contracting among self-interested bounded rational agents. Technical report, University of Massachusetts at Amherst Computer Science Department, 1995. In preparation.

[Sandholm and Lesser, 1995c] Tuomas W Sandholm and Victor R Lesser. Coalition formation among bounded rational agents. In *Proc. 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, 1995.

[Sandholm and Lesser, 1995d] Tuomas W Sandholm and Victor R Lesser. Equilibrium analysis of the possibilities of unenforced exchange in multiagent systems. In *Proc. 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, Montreal, 1995.

[Sandholm, 1993] Tuomas W Sandholm. An implementation of the contract net protocol based on marginal cost calculations. In *Proc. 11th National Conference on Artificial Intelligence (AAAI-93)*, July 1993.

[Sathi and Fox, 1989] A Sathi and M Fox. Constraint-directed negotiation of resource reallocations. In Michael N. Huhns and Les Gasser, eds., *Distributed Artificial Intelligence*, vol. 2 of *Research Notes in Artificial Intelligence*, ch. 8, pages 163–193. Pitman, 1989.

[Sen, 1993] Sandip Sen. *Tradeoffs in Contract-Based Distributed Scheduling*. PhD thesis, Univ. of Michigan, 1993.

[Simon, 1982] Herbert A Simon. *Models of bounded rationality*, volume 2. MIT Press, 1982.

[Smith, 1980] Reid G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29(12):1104–1113, December 1980.

[Turner, 1992] Roy M Turner. The tragedy of the commons and distributed ai systems. In *Proceedings of the 12th International Workshop on Distributed Artificial Intelligence*, pages 379–390, May 1992.

[Van Dyke Parunak, 1987] H Van Dyke Parunak. Manufacturing experience with the contract net. In Michael N. Huhns, editor, *Distributed Artificial Intelligence*, Research Notes in Artificial Intelligence, chapter 10, pages 285–310. Pitman, 1987.

[Varian, 1992] Hal R Varian. *Microeconomic analysis*. New York: W. W. Norton, 1992.

[Waters, 1987] C D Waters. A solution procedure for the vehicle-scheduling problem based on iterative route improvement. *Journal of the Operational Research Society*, 38(9):833–839, 1987.

[Zilberstein, 1993] Shlomo Zilberstein. *Operational rationality through compilation of anytime algorithms*. PhD thesis, University of California, Berkeley, 1993.