An Exchange Protocol without Enforcement

Tuomas W. Sandholm and Victor R. Lesser
Computer Science Department
University of Massachusetts at Amherst
CMPSCI Technical Report 94-44
July 1994

An Exchange Protocol without Enforcement*

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

Abstract

In multiagent systems, interaction protocols are usually enforced by law. Enforcement is problematic among computational agents, because they may operate under incomplete or different laws, the laws may not be uniformly enforced, and the agents can vanish easily. This paper presents an enforcement free method for carrying out exchanges so that both agents are motivated to abide to their contract. This is achieved by splitting the exchanged goods into partial exchanges so that at each step, both agents benefit more from the future of the exchange than from vanishing with the goods or payment. The conditions for such exchange are presented in general, and the maximum deliveries and payments (for any point in the exchange) are explicitly solved for. Similar analysis is carried out for the case, where the agents' current actions affect their future contracts. Strategic delaying is also discussed. The paper presents a fast algorithm that will find a sequence of independent partial deliveries in a way that enables unenforced exchange if such a sequence exists. This problem cannot be solved in polynomial time if the partial deliveries are interdependent. Finally, the paper shows that the unenforced exchange scheme hinders unfair renegotiation.

1 Introduction

In cooperative distributed problem solving [3], the system designer imposes an interaction *protocol* and a *strategy* (a mapping from state history to actions; a way to use the protocol) for each agent. In multiagent systems [9, 12, 2, 5, 11], the agents are provided with an interaction protocol, but each agent may choose its own strategy. This allows the agents to be constructed by separate designers and/or represent different real world parties. Agents in such systems often act based on self-interest, and the protocols have to be constructed accordingly. An example interaction protocol is the *auction*, where some agents bid to take responsibility for a task, which is awarded to the lowest price bidder. The bids are binding: if an agent makes a bid and the task is awarded to it, it must take responsibility for the task at that price. Among real world agents, this protocol is enforced by law.

Such enforced protocols are problematic when used among computational agents. First, there may be a lack of laws for interactions of computational agents, or the agents may be governed by different laws (eg. sited in different countries). It may also be the case that the laws are not strictly enforced or that enforcing them (eg. by litigation) is

*This paper is an extended version of [10].

†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.

impractically expensive. We would like the agents' interactions to work properly independent of fluctuations in enforcement. Secondly, a computational agent may vanish at any point in time, eg. by killing its own process. Thus, the laws cannot be enforced unless the terminated agent represented some real world party and the connection between the agent and the real world party can be traced. For example, the Telescript technology [4] follows the approach of strictly trying to tie each agent to its real world party. On the contrary, we analyze exchanges among more autonomous agents and study possibilities of exchange without enforcement (eg. with unknown real world parties or no litigation possibility). In cases where this type of exchange is possible, it is clearly preferable to the strictly enforced mode of exchange due to savings in enforcement costs and lack of enforcement uncertainty.

The fulfillment of a mutual contract can be viewed as one agent delivering and the other agent paying. We propose a method for carrying out such an exchange without enforcement. The exchange is managed so that for both agents (supplier and demander), at any point in the exchange, the future gains from carrying out the rest of the exchange (*cooperating* according to the contract) are larger than the gains from *defecting*. Defection is equivalent to prematurely terminating the exchange by vanishing. For example, defection may be beneficial to a demander agent if the supplier agent has delivered much more than what the demander has yet paid for. By intelligently splitting the exchange into smaller portions, the agents can avoid situations where at least one of them is motivated to defect. We will call a sequence of deliveries and payments *safe*, if neither agent is motivated to defect at any point in the exchange. The basic idea of enhancing cooperation by making the present less important compared to the future has been suggested for example in [1].

We propose an *exchange strategy manager* module to be added to each agent's architecture. This module is potentially different for each agent. Its role is to schedule the agent's deliveries (or payments) in such a way that the opponent is not motivated to defect at any point in the exchange (this is in the agent's self-interest). The exchange strategy manager also provides the agent's negotiator module with information on whether a certain proposed contract can be carried out safely. Unless protocol enforcement is guaranteed, the negotiator should only agree to contracts that can be executed so that the opponent is not motivated to defect at any point of the exchange.

Section 2 handles one exchange in isolation. Conditions for safe exchange are derived and an inherent problem concerning the completion of the exchange is identified. Section 3 takes the agents' future transactions into account in describing safe exchange in order to solve the completion problem. The role of time in an exchange is discussed in section 4. Section 5 analyses the case, where the delivering order of independent goods can be varied. A quadratic sequencing algorithm is presented that finds a safe sequence if one exists. Section 6 studies sequencing of interdependent goods. Section 7 describes the problem of forced renegotiation, and section 8 concludes.

2 Isolated exchange of goods for compensation

This section describes conditions under which a rational agent cooperates in an exchange. The criteria are based on the idea that an agent will cooperate if its future gain from

cooperation is greater than its gain from instant defection. The exchange proceeds on two axis: the portion of goods of the contract delivered so far $x \in [0,1]$, and the cumulative payment so far $P(x)$. The payment may be monetary or other goods. The agents value the goods x according to increasing value functions that are in equivalent units of payment $P(x)$. The supplier's value function $V_s(x)$ describes how much cost the supplier incurs by generating and delivering x. The demander's value function $V_d(x)$ describes what the goods x are worth to the demander. Trivially, $V_s(0) = 0$ and $V_d(0) = 0$.

At any point in an exchange, an agent has the options of defecting or cooperating. Defecting gives no added gains (that have not already been received) and no added costs, so its net benefit is 0. Therefore, a rational (net benefit maximizing) supplier agent cooperates at x in an isolated exchange if (and only if) its future compensation is at least as great as its future cost[1]. So for a rational supplier agent to cooperate throughout an exchange in the case of continuous goods and value functions, it has to hold that $\forall x \in [0,1]$,

$$\int_x^1 P'(\alpha) - V_s'(\alpha) d\alpha \geq 0, \tag{1}$$

and in the discrete case, $\forall x \in [0,1]$,

$$\sum_{\alpha \in [x,1]} \Delta P(\alpha) - \Delta V_s(\alpha) \geq 0. \tag{2}$$

The supplier controls x (based on how much payment it has received so far), but the demander controls $P(x)$. Next we analyze how the demander should control its payments to motivate the supplier to cooperate. Solving equation 1 (or equation 2) gives the following condition that is necessary and sufficient to avoid defection by the supplier: $P(x) - V_s(x) \leq P(1) - V_s(1)$. It gives an upper limit on the cumulative payment at any point x. Let us call this maximum payment $P^{max}(x)$. See fig. 1 left.

$$P^{max}(x) \stackrel{\text{def}}{=} V_s(x) + P(1) - V_s(1). \tag{3}$$

The supplier will assume that $P(1) = P^{contr}$ (the contract price), because it can control the demander's cooperation as follows. A rational demander agent cooperates in an isolated exchange if (and only if) the future compensation it has to pay is smaller than or equal to its added value. So for a rational demander agent to cooperate throughout an exchange in the case of continuous goods and value functions, it has to hold that $\forall x \in [0,1]$,

$$\int_x^1 V_d'(\alpha) - P'(\alpha) d\alpha \geq 0. \tag{4}$$

and in the discrete case, $\forall x \in [0,1]$,

$$\sum_{\alpha \in [x,1]} \Delta V_d(\alpha) - \Delta P(\alpha) \geq 0. \tag{5}$$

As above, we can calculate how the supplier should control the delivering in order to motivate the demander to cooperate (equation 4 or 5): $P(x) - V_d(x) \geq P(1) - V_d(1)$.

[1] If equality holds, the agent is indifferent between cooperating and defecting, in which case we will assume it will cooperate.

This equation gives a lower limit for the cumulative payment at point x. Let us call this minimum payment $P^{min}(x)$. See fig. 1 left.

$$P^{min}(x) \stackrel{\text{def}}{=} V_d(x) + P(1) - V_d(1). \tag{6}$$

These equations reflect the situation where the demander assumes that the delivery will be totally made, i.e. finally $x = 1$. This corresponds to the supplier cooperating throughout the exchange, which the demander can control as shown above.

In order for any point $(x, P(x))$ to be safe, it clearly has to hold that

$$P^{min}(x) \leq P(x) \leq P^{max}(x) \tag{7}$$

or stated in terms of the agents' value functions:

$$V_s(1) - V_s(x) \leq V_d(1) - V_d(x). \tag{8}$$

At $x = 0$ this gives $V_s(1) \leq V_d(1)$, which is an intuitive condition for the contract to have been made in the first place. Specifically, $V_s(1) \leq P^{contr} \leq V_d(1)$, fig. 1 left. Equation 8 can be written as $V_d(x) - V_s(x) \leq V_d(1) - V_s(1)$. This means that the agents' combined profit must be higher at $x = 1$ than at any other $x \in [0, 1[$. If the agents would have been better off by making the contract for a smaller amount, an isolated safe exchange is impossible.

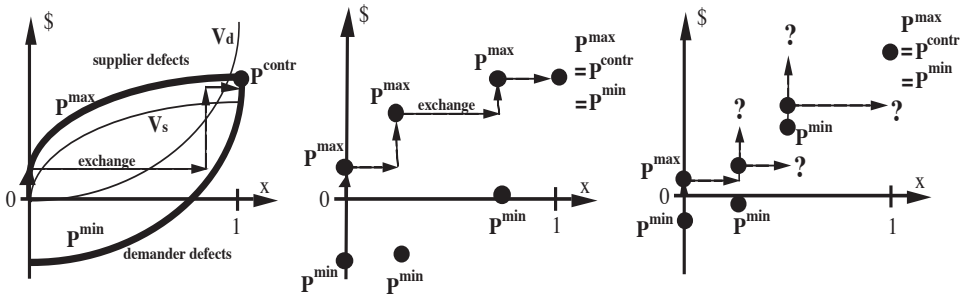

Figure 1: *Left: example of safe exchange. Middle: safe exchange of discrete goods possible. Right: safe exchange of discrete goods not possible.*

If the agents do not know each others value functions, they can use bounds they know. The supplier is safe using an upper bound for $P^{min}(x)$ (i.e. a lower bound for $V_d(1)$ and an upper bound for $V_d(x)$). The demander is safe using a lower bound for $P^{max}(x)$. Although the agents are safe using these bounds, even possible exchanges are disabled if the bounds are too far off.

Condition 7 is not sufficient for safe exchange. Under that condition, each cumulative delivery amount $x \in [0, 1]$ has some cumulative payment $P(x)$ so that at the point $(x, P(x))$ neither the supplier nor the demander wants to defect. This does not guarantee that such safe points can be reached. Thus it may happen that the exchange cannot proceed. In the next sections we will state the necessary and sufficient conditions for safe exchange.

2.1 Discrete goods

Discrete goods are goods that are inherently split into atomic chunks. Such chunks cannot be further split, and we assume in this section that the delivery order of the chunks is strictly predefined. In the TRACONET (TRAnsportation COoperation NET) multiagent system [9], agents representing dispatch centers negotiated over who's vehicles should transport which parcels. Taking care of one parcel is an atomic chunk, because the task cannot be split. Sometimes a contract between two agents involved multiple tasks (in order to avoid local optima in the distributed optimization [9]), so the total exchange could have been split into smaller parts. The following theorem describes the conditions for safe exchange of discrete goods:

Theorem 2.1 *With discrete goods, for increasing $P^{max}(x)$ and $P^{min}(x)$, a safe exchange is possible iff for every two consecutive steps x and x', $P^{max}(x) \geq P^{min}(x')$.*

Proof. **If:** Assume that the condition holds. We present an example path (fig. 1 middle) that is safe (because it never violates $P^{min}(x) \leq P(x) \leq P^{max}(x)$) and that contains no gaps. First, the demander pays $P(0) = P^{max}(0)$ thus moving from $(0,0)$ to $(0, P^{max}(0))$. Then, at any point $(x, P^{max}(x))$, the supplier delivers the next portion of the goods (increases cumulative delivery from x to x'). This move does not violate the safety constraint $P^{max}(x) \geq P^{min}(x')$ by our assumption. Next, the demander increases cumulative payment to $P(x') = P^{max}(x')$ thus moving to $(x', P^{max}(x'))$. This is obviously possible. The process continues by alternating moves between the supplier and the demander.

Only if: Assume that $P^{max}(x) < P^{min}(x')$. At x, the demander cannot move beyond $P^{max}(x)$, because otherwise the supplier would defect. The supplier cannot move to x', because the demander would defect at any point (x', P), where $P \leq P^{max}(x) < P^{min}(x')$. See fig. 1 right. □

Considering isolated exchanges of discrete goods, theorem 2.1 is a negative result. Substituting $x = 1$ in the definitions (equations 3 and 6) of $P^{max}(x)$ and $P^{min}(x)$, we see that $P^{max}(1) = P^{min}(1) = P(1)$. Theorem 2.1 requires that $P^{max}(x) \geq P^{min}(x')$ for any two consecutive x and x'. Let us call the size of the last delivery Δx. So for safe exchange the following has to hold: $P^{max}(1 - \Delta x) \geq P^{min}(1) = P^{max}(1)$. Thus the increasing function $P^{max}(x)$ has to be constant during the last step (fig. 1 middle). This means that the supplier's value function $V_s(x)$ is constant. So an isolated safe exchange is possible only if the supplier does not incur any cost from generating and delivering the last chunk. An example of this is when the supplier has had to acquire a number of the last deliverables as an atomic item. Then its cost of acquiring the deliverables can be entirely attributed to the first one, while it can deliver these items separately with only the first one increasing $V_s(x)$ (assuming negligible costs of physically delivering). This may not occur very often in practise (fig. 1 right). Intuitively, when there is no future benefit to be gained from exchanging, agents are better off defecting on the current move.

If this problem occurs, it spoils the entire exchange. On the last turn, the supplier does not want to increase delivery to $x = 1$, because the demander would defect. Similarly, the demander does not want to increase cumulative payment above $P^{max}(1 - \Delta x)$, because the supplier would defect. Both agents know that the last part of the exchange will not take place due to this. So they can analyze the exchange as if it did not have the last part.

Now the second to last part has the same problem (unless the supplier can deliver that part without cost): neither agent wants to initiate that part. Again, both agents know this and so on. This backward induction can be carried out up to the first exchange. So, neither agent will make any move, and the exchange will not take place.

In the case of continuous goods, this problem appears to be less significant. Continuous goods are discussed in the next section. In both the discrete and the continuous case, the problem can be overcome by considering possible related future interactions of the agent. This issue will be discussed in section 3.

2.2 Continuous goods

In this section we analyze the exchange of continuous goods, i.e. goods that can be split arbitrarily. Again we assume that the order of goods to deliver is fixed beforehand. First, we state the conditions for safe exchange:

Theorem 2.2 *With continuous goods, for increasing $P^{max}(x)$ and $P^{min}(x)$, a safe exchange can enter a point $x_0 \in]0,1]$ iff one of the following holds:*

1. *$P^{max}(x_0) = P^{min}(x_0)$, and $P^{max}(x)$ is constant in some left neighborhood of x_0.*
2. *$P^{max}(x_0) > P^{min}(x_0)$, and $\lim_{x \to x_0^-} P^{max}(x) > P^{min}(x_0)$.*
3. *$P^{max}(x_0) > P^{min}(x_0)$, $\lim_{x \to x_0^-} P^{max}(x) = P^{min}(x_0)$, and $P^{max}(x)$ is constant in some left neighborhood of x_0.*

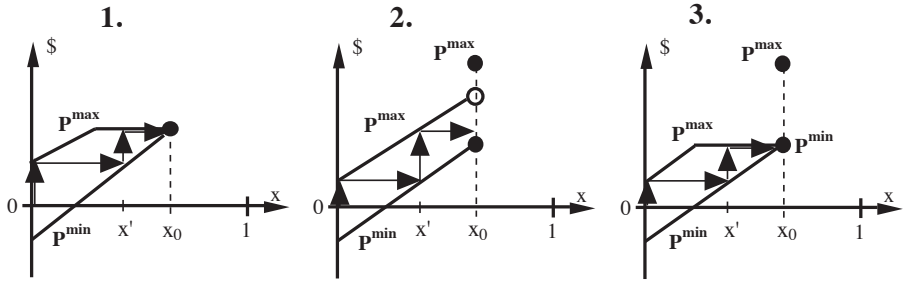

Figure 2: *Exchanging continuous goods: reaching a point (cases of the theorem).*

Proof. We need only consider moves by the supplier, because only they move the exchange along the x-axis. Obviously, if $P^{max}(x_0) < P^{min}(x_0)$, the exchange cannot proceed to x_0, because safety requires that $P^{min}(x_0) \leq P(x_0) \leq P^{max}(x_0)$. Next we prove the three cases of the theorem.

Case 1. If: Assume the conditions hold (fig. 2:1). Let us choose a point x' ($x' < x_0$) in the mentioned left neighborhood. The point $(x', P^{max}(x_0))$ is safe, because $P^{max}(x') = P^{max}(x_0)$. Now the supplier can make the move from $(x', P^{max}(x_0))$ to $(x_0, P^{max}(x_0))$.

Only if: Assume $P^{max}(x_0) = P^{min}(x_0)$, but that the second part does not hold, i.e. $P^{max}(x)$ is strictly increasing in some left neighborhood of x_0. Every safe point $(x', P(x')), x' < x_0$ has to satisfy $P(x') < P^{max}(x_0)$, because $P^{max}(x)$ is strictly increasing. Any move to x_0 by the supplier would lead to some point $(x_0, P(x')$, which is not safe, because $P(x') < P^{max}(x_0) = P^{min}(x_0)$.

Case 2. **If:** Assume the conditions hold (fig. 2:2). Because $\lim_{x \to x_0^-} P^{max}(x) > P^{min}(x_0)$, $\exists x'$, s.t. $x' < x_0$, and $P^{max}(x') > P^{min}(x_0)$. Therefore, the supplier's move from $(x', P^{max}(x'))$ to $(x_0, P^{max}(x'))$ is safe.

Only if: Assume $P^{max}(x_0) > P^{min}(x_0)$, but that $\lim_{x \to x_0^-} P^{max}(x) < P^{min}(x_0)$ (equality is handled as case 3). So, $\forall x' < x_0$, $P^{max}(x') < P^{min}(x_0)$ implying $P(x') < P^{min}(x_0)$. Therefore, the supplier's move from any safe point $(x', P(x'))$ to $(x_0, P(x'))$ is not safe.

Case 3. **If:** Let us choose a point x' ($x' < x_0$) in the mentioned left neighborhood. The point $(x', P^{max}(x_0))$ is safe, because $P^{max}(x') = P^{max}(x_0)$. Now the supplier can make the move from $(x', P^{max}(x_0))$ to $(x_0, P^{max}(x_0))$, fig. 2:3.

Only if: Assume $P^{max}(x_0) > P^{min}(x_0)$, and $\lim_{x \to x_0^-} P^{max}(x) = P^{min}(x_0)$, but $P^{max}(x)$ is strictly increasing in some left neighborhood of x_0. So, $\forall x' < x_0$, $P^{max}(x') < P^{min}(x_0)$ implying $P(x') < P^{min}(x_0)$. Therefore, the supplier's move from any safe point $(x', P(x'))$ to $(x_0, P(x'))$ is not safe. □

We showed the conditions under which a safe exchange can proceed to some amount of cumulative delivery x. It is obvious that if the exchange cannot proceed to x, it cannot proceed beyond x either (because $P^{min}(x)$ is increasing). Having reached x the exchange may or may not be able to be carried out further. The next theorem states the exact conditions for the exchange to proceed.

Theorem 2.3 *With continuous goods, for increasing $P^{max}(x)$ and $P^{min}(x)$, a safe exchange can proceed from a point $x_0 \in [0, 1[$ iff one of the following holds:*

1. *$P^{max}(x_0) = P^{min}(x_0)$, and $P^{min}(x)$ is constant in some right neighborhood of x_0.*
2. *$P^{max}(x_0) > P^{min}(x_0)$, and $\lim_{x \to x_0^+} P^{min}(x) < P^{max}(x_0)$.*
3. *$P^{max}(x_0) > P^{min}(x_0)$, $\lim_{x \to x_0^+} P^{min}(x) = P^{max}(x_0)$, and $P^{min}(x)$ is constant in some right neighborhood of x_0.*

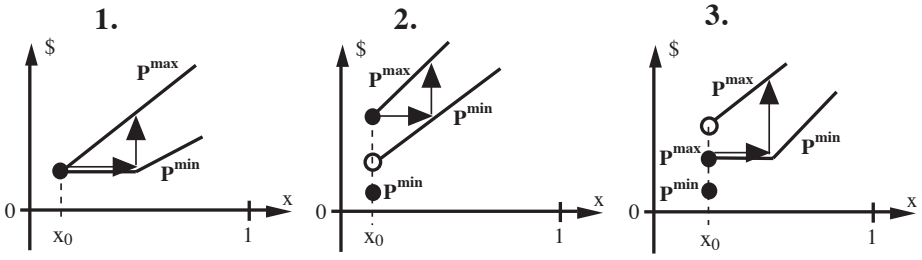

Figure 3: *Exchanging continuous goods: leaving a point (cases of the theorem).*

Proof. Similar to the proof of theorem 2.2. □

Safe exchange can be carried out if (and only if) some initial delivery can be made, every intermediate amount of delivery can be reached and departed, and the final amount of delivery can be reached. So if the conditions of theorem 2.3 hold for $x = 0$, conditions of theorems 2.2 and 2.3 hold for every $x \in]0, 1[$, and conditions of theorem 2.2 hold for

$x = 1$, then safe exchange can be carried out. Theorems 2.2 and 2.3 do not assume continuity of $P^{max}(x)$ (equivalently $V_s(x)$) or $P^{min}(x)$ (equivalently $V_d(x)$). Neither do they assume that $P^{max}(x)$ or $P^{min}(x)$ is strictly increasing. If $P^{max}(x)$ and $P^{min}(x)$ are continuous, safe exchange can be carried out if and only if $\forall x \in [0,1]$, either $P^{max}(x) > P^{min}(x)$ or $P^{max}(x) = P^{min}(x)$ and $P^{max}(x)$ is constant in a left neighborhood of x and $P^{min}(x)$ is constant in a right neighborhood of x. If in addition to continuity, $P^{max}(x)$ and $P^{min}(x)$ are strictly increasing, safe exchange is doable if and only if $\forall x \in [0,1], P^{max}(x) > P^{min}(x)$.

Isolated safe exchange is problematic also in the case of continuous goods. Substituting $x = 1$ into the definitions of $P^{max}(x)$ and $P^{min}(x)$, we see that $P^{max}(1) = P(1) = P^{min}(1)$. From case 1 of theorem 2.2 we see that full delivery ($x = 1$) can be reached only if $P^{max}(x)$ is constant in some left neighborhood of $x = 1$. So if the value function of the supplier $V_s(x)$ is not constant in the end of the exchange, the exchange cannot be completed. In other words, an isolated safe exchange is possible only if the supplier does not incur any cost from generating and delivering the last portion of the goods. This problem is less severe than in the case of discrete goods, because the size of the last portion to be delivered can be made arbitrarily small. In theory, the agents can carry out the exchanges infinitely long, thus getting arbitrarily close to $x = 1$. Furthermore, the backward induction argument that disabled the entire exchange in the case of discrete goods does not hold. There is no exchange step at which the agents could reason that neither will make a move.

We presented three theorems stated in terms of $P^{max}(x)$ and $P^{min}(x)$ that analyzed when safe exchange was possible, i.e. when each agent is motivated to cooperate at any point in the exchange *given that the opponent will cooperate* throughout the exchange. So in game theoretic terms, the cooperation of both agents throughout the exchange is a *subgame perfect Nash equilibrium* [6].

3 Non-isolated exchange

So far we have considered one exchange in isolation. Often, an agent interacts with other agents more than just once in its lifetime. One interaction may affect the agent's future interactions. For example, if an agent defects in the current exchange, its counterpart may not want to take on future contracts with that agent. Moreover, the counterpart can notify other agents that the agent defected. Thus, the agent's interactions with third parties may also be hindered by defecting in the current exchange. The hindering future impact of a defection can be thought of as an extra cost. Fearing this future cost, rational self-motivated agents can cooperate in the current exchange, even if it would be rational to defect in it when considered in isolation. The methods for calculating defection impacts on the future are beyond the scope of this paper. We assume that the agent knows the defection cost of its opponent. We denote the defection cost of the supplier by c_s^{def} and the one of the demander by c_d^{def}.

Now we can incorporate the defection costs into the cooperation conditions of a rational supplier agent (equations 1 and 2) so that the demander would know how to control payments in order to motivate the supplier to cooperate. In the case of continuous goods

and value functions, $\forall x \in [0,1]$,

$$\int_x^1 P'(\alpha) - V_s'(\alpha) d\alpha \geq -c_s^{def}, \tag{9}$$

and in the discrete case, $\forall x \in [0,1]$,

$$\sum_{\alpha \in [x,1]} \Delta P(\alpha) - \Delta V_s(\alpha) \geq -c_s^{def}. \tag{10}$$

Similarly, the cooperation conditions of a rational demander can be rewritten. The supplier should make the conditions hold by controlling deliveries - thus motivating the demander to cooperate. In the case of continuous goods and value functions, $\forall x \in [0,1]$,

$$\int_x^1 V_d'(\alpha) - P'(\alpha) d\alpha \geq -c_d^{def}. \tag{11}$$

and in the discrete case, $\forall x \in [0,1]$,

$$\sum_{\alpha \in [x,1]} \Delta V_d(\alpha) - \Delta P(\alpha) \geq -c_d^{def}. \tag{12}$$

Solving equation 9 (or 10) gives the following condition that is necessary and sufficient to avoid supplier's defection: $P(x) - V_s(x) \leq P(1) - V_s(1) + c_s^{def}$. This allows us to redefine $P^{max}(x)$ (figure 4):

$$P^{max}(x) \stackrel{\text{def}}{=} V_s(x) + P(1) - V_s(1) + c_s^{def}. \tag{13}$$

Similarly, we solve how the supplier should control delivery to motivate the demander to cooperate (equation 11 or 12): $P(x) - V_d(x) \geq P(1) - V_d(1) - c_d^{def}$, and we redefine $P^{min}(x)$ (figure 4):

$$P^{min}(x) \stackrel{\text{def}}{=} V_d(x) + P(1) - V_d(1) - c_d^{def}. \tag{14}$$

In terms of the agents' value functions, condition $P^{min}(x) \leq P^{max}(x)$ becomes

$$V_d(x) - V_d(1) - c_d^{def} \leq V_s(x) - V_s(1) + c_s^{def}. \tag{15}$$

The necessary and sufficient conditions for (subgame perfect Nash equilibrium) safe exchange (theorems 2.1, 2.2, and 2.3) apply directly to the case of non-isolated exchange with the new definitions of $P^{max}(x)$ and $P^{min}(x)$.

In the isolated exchange case, substituting $x = 1$ in the definitions of $P^{max}(x)$ and $P^{min}(x)$ gave $P^{min}(1) = P(1) = P^{max}(1)$. This led to the problem that the exchange could not be carried out to completion (unless $P^{max}(x)$ was constant in the end of the exchange). Now, in the non-isolated exchange case, let us substitute $x = 1$ in the definitions of $P^{max}(x)$ and $P^{min}(x)$. The condition $P^{min}(1) \leq P(1) \leq P^{max}(1)$ gives $P(1) - c_d^{def} \leq P(1) \leq P(1) + c_s^{def}$. The defection penalties give leeway to the exchange, thus possibly enabling safe exchange to be completed (even if $P^{max}(x)$ is not constant in the end of the exchange). The contract price P^{contr} can be overshot due to this leeway. To avoid this, the demander just limits the total payment at each x to $min(P^{contr}, P^{max}(x))$. This will not hinder exchange, because the condition takes effect after the full contract price has

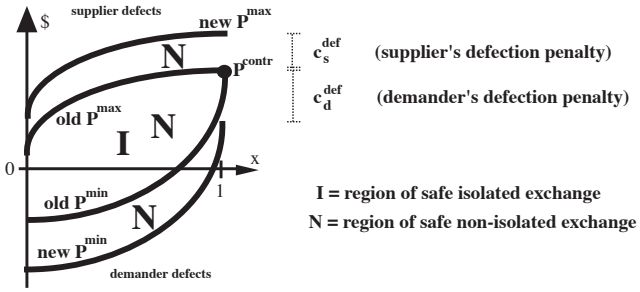

Figure 4: *Defection penalties of non-isolated exchange give leeway to safe moves.*

been paid. So, non-isolated safe exchange is more fruitful than isolated safe exchange, because it facilitates completing the exchange.

If the agent does not know the defection cost of the opponent, it can use a lower bound for that cost. This way the agent is safe, but if the bound is too far off, even possible exchanges are disabled.

4 Role of time

What will motivate the agents to take as large safe steps as possible? If there is a partitioning cost, for example packaging each piece of information sent over a net, then taking largest possible safe steps minimizes costs by minimizing the number of required parts. But if there are packaging costs for each part, the last part cannot be delivered without cost and as we have shown earlier, isolated safe exchange becomes impossible. Anyhow, non-isolated safe exchange may be possible also in that case. If there are no partitioning costs, the agents are indifferent between large and small steps.

Another question has to do with time. What will motivate the agents to carry out the deliveries as fast as possible? Will the agents move as soon as possible even if they discount payments and the value of goods (or cost of producing goods)? We use general nonincreasing discount functions $f(t)$, $(0 \leq f(t) \leq 1)$ with subscripts s and d to distinguish between the supplier and the demander, and superscripts p and v to characterize whether the discount applies to payment or the value of goods. In practise, $f(0) = 1$. For example, using constant interest rate (r) compounded interest, the discount function is $f(t) = e^{-rt}$.

A rational supplier tries to maximize $\int_t^\infty f_s^p(\tau)p'(\tau) - f_s^v(\tau)v_s'(\tau)d\tau$ by controlling the amount of delivery $x(t)$ and thus controlling $v_s(t)$ and $v_d(t)$. A rational demander tries to maximize $\int_t^\infty f_d^V(\tau)v_d'(\tau) - f_d^p(\tau)p'(\tau)d\tau$ by controlling payment $p(t)$. When controlling $x(t)$, the supplier can take into account the payments so far, and when controlling $p(t)$, the demander can take into account the deliveries so far, but we will not explicitly write out this dependency in the function definitions $x(t)$ and $p(t)$.

We assume that each agent knows the *current value* of its opponent's defection cost, and that the current value does not change, which seems realistic. In the case where the opponent discounts payments, this means that its *absolute value* of the defection cost increases with time. We denote the current value of the defection cost of the supplier by c_s^{def} and the one of the demander by c_d^{def}.

In the following analysis we will show that under certain conditions, each agent's optimal strategy is to move immediately (and cooperate throughout the exchange) given that the opponent moves immediately (and cooperates throughout the exchange). So again our solution concept is the subgame perfect Nash equilibrium. The analysis is valid for situations where non-time-dependent safe exchange is possible, i.e. theorems 2.2 and 2.3, or 2.1[2] hold. Two conditions have to be satisfied for an agent to move immediately:

- Cooperating immediately must be better than cooperating after any delay, and
- Cooperating immediately must be better than defecting after any delay.

The first condition for the supplier can be stated as follows (if the agents move immediately, time discounts are 1): $\forall t \geq 0$,

$$\int_t^\infty p'(\tau) - v_s'(\tau)d\tau \geq \int_t^\infty f_s^p(\tau)p'(\tau) - f_s^v(\tau)v_s'(\tau)d\tau, \tag{16}$$

and the second one (analog of 9) as: $\forall t \geq 0$,

$$\int_t^\infty p'(\tau) - v_s'(\tau)d\tau \geq -c_s^{def}. \tag{17}$$

The first condition for a rational demander agent can be written as: $\forall t \geq 0$,

$$\int_t^\infty v_d'(\tau) - p'(\tau)d\tau \geq \int_t^\infty f_d^V(\tau)v_d'(\tau) - f_d^p(\tau)p'(\tau)d\tau, \tag{18}$$

and the second one (analog of 11) as: $\forall t \geq 0$,

$$\int_t^\infty v_d'(\tau) - p'(\tau)d\tau \geq -c_d^{def}. \tag{19}$$

It turns out that in the region of safe isolated exchange ($c_d^{def} = c_s^{def} = 0$ in inequalities 17 and 19; region I in figure 4) immediate cooperation of both agents is a subgame perfect Nash equilibrium if the supplier discounts payments more sharply than production costs and the demander discounts value of goods more sharply than payment. This is because the safety inequalities (17 and 19) *imply* the immediacy inequalities (16 and 18):

Theorem 4.1 *Let $f_s^p(t)$, $f_s^v(t)$, $f_d^p(t)$ and $f_d^v(t)$ be nonincreasing functions that range between 1 and 0. Let $c_d^{def} = c_s^{def} = 0$. If $\forall t \geq 0, f_s^p(t) \leq f_s^v(t)$, then inequality 17 implies inequality 16. Similarly, if $\forall t \geq 0, f_d^p(t) \geq f_d^v(t)$, then inequality 19 implies inequality 18.*

Proof. We first prove that inequality 17 implies inequality 16. Let $g(t) = p'(t) - v_s'(t)$. Let $S(t) = \int_t^\infty g(\tau)d\tau$. Thus $S(t) = -\int_\infty^t g(\tau)d\tau$ and $g(t) = -S'(t)$. Now, $\int_t^\infty f_s^p(\tau)p'(\tau) - f_s^v(\tau)v_s'(\tau)d\tau \leq \int_t^\infty f_s^p(\tau)p'(\tau) - f_s^p(\tau)v_s'(\tau)d\tau = \int_t^\infty f_s^p(\tau)g(\tau)d\tau = \int_t^\infty -f_s^p(\tau)S'(\tau)d\tau =^3 -f_s^p(\tau)S(\tau)|_t^\infty + \int_t^\infty f'^p_s(\tau)S(\tau)d\tau \leq^4 -f_s^p(\tau)S(\tau)|_t^\infty \leq S(t) = \int_t^\infty p'(\tau) - v_s'(\tau)d\tau$. The fact that inequality 19 implies inequality 18 can be proven similarly. □

[2]The discrete case will not be discussed for brevity.

[3]Integration by parts.

[4]Because $f'^p_s(\tau) \leq 0$ and by inequality 17, $S(\tau) \geq 0$.

The condition on the supplier's discount functions is rather natural, because for example in a stable environment the supplier's current value of producing an item should remain constant, but obviously payment is discounted. The condition on the demander's discount functions is more stringent. It is realistic in the case where the demander needs the goods urgently. An agent need not know the opponent's exact discount functions. It is sufficient to know whether they fulfill the conditions.

We derived the desired immediacy equilibrium solution for the often impossible isolated exchange. Immediacy may be an equilibrium also in non-isolated exchange (region N in figure 4), but it requires stricter conditions on the discount functions. If $0 > \int_t^\infty p'(\tau) - v_s'(\tau)d\tau \geq -c_s^{def}$ (corresponding to being in the region where the supplier will not defect for non-isolated exchange, but will for isolated exchange), the supplier is motivated to move as fast as it can if (and only if) its discount factor for producing the goods ($f_s^v(t)$) stays sufficiently much greater than its discount factor for payment ($f_s^p(t)$) for all t. Similarly, if $0 > \int_t^\infty v_d'(\tau) - p'(\tau)d\tau \geq -c_d^{def}$ (corresponding to being in the region where the demander will not defect for non-isolated exchange, but will for isolated exchange), the demander is motivated to move as fast as it can if (and only if) its discount factor for payment ($f_d^p(t)$) stays sufficiently much greater than its discount factor for the value of goods ($f_d^v(t)$) for all t. Inequalities 16 and 18 specify exactly what is meant by "sufficiently much". Intuitively, an agent wants to postpone a negative net benefit into the future where it is heavily discounted.

Even though immediate exchange is an equilibrium under the mentioned conditions, it may be more practical to force it. This can be done by specifying *deadlines* or *lateness penalty schedules* for both agents in the contract. The role of the contract is to state such exchange conditions, and if the contract is not abided to, the defecting agent will suffer the defection penalty (c_s^{def} or c_s^{def}) due to how its defection will affect its future contracts. So, strictly speaking a contract matters only in non-isolated exchange, and therefore forcing timely exchange is possible only in such cases. This highlights the value of our theorem that immediate exchange is an equilibrium in isolated exchange under the mentioned conditions. Even in non-isolated exchange, deadlines and lateness penalties are meaningful only as long as abiding to the deadline or paying the lateness penalty is less expensive than suffering the defection penalty. Lateness penalty schedules are preferable to strict deadlines, because they are less risky for an agent who is potentially subject to the lateness penalty, but the other agent can still tailor the lateness penalty schedule to motivate the former to move immediately.

In this section we have viewed immediate exchange as a desideratum. The discount functions also play a dual role by sometimes enabling non-immediate safe exchange that would not be safe if carried out immediately. If $f_s^p(t) < f_s^v(t)$ (i.e. the supplier discounts payment more sharply than the value of goods), time delays in an exchange may make it safe. The same may occur if $f_d^p(t) > f_d^v(t)$, i.e. the demander discounts payments less sharply than the value of goods. Although this method may be safe, it is not efficient, because it takes advantage of the fact that agents lose gains in time.

5 Ordering of discrete independent goods

So far we have looked at exchanges in which the delivering order of the goods is fixed beforehand (eg. at the time of the contract). In this section we analyze an exchange, where discrete partial deliveries (individual goods or atomic chunks) can be delivered in any order, as long as all of them get delivered. Here we assume that the demander's added value from one chunk does not depend on the other chunks delivered so far. Similarly, we assume that the supplier's cost from delivering a chunk does not depend on other chunks delivered earlier. This enables us to associate each chunk c with two values, ΔP_c^{max} and ΔP_c^{min}, that fully characterize how much the maximum and the minimum cumulative payments change as a chunk c is delivered.

As an application example, a computationally powerful agent could make a contract to carry out some number of independent matrix multiplications. Multiplying two matrices neither facilitates nor hinders multiplying some other two, so the chunks are independent with respect to the supplier. The chunks are truly independent if they are independent with respect to the demander also (based on the uses of the multiplication results).

Let us call an ordering of the deliveries safe if safe exchange is possible under that ordering. We provide a fast greedy algorithm that will find a safe ordering if one exists. The algorithm takes five inputs: a set of chunks C, a vector of ΔP_c^{max} values, a vector of ΔP_c^{min} values, the maximum payment before anything is delivered P_{init}^{max}, and the minimum payment before anything is delivered P_{init}^{min}.

Algorithm 5.1 SEQUENCE-CHUNKS$(C, \boldsymbol{\Delta P}^{max}, \boldsymbol{\Delta P}^{min}, P_{init}^{max}, P_{init}^{min})$

1. *Divide C into two sets POS and NEG s.t. $POS = \{c \in C | \Delta P_c^{max} - \Delta P_c^{min} \geq 0\}$ and $NEG = \{c \in C | \Delta P_c^{max} - \Delta P_c^{min} < 0\}$.*
2. *$P^{max} = P_{init}^{max}$, $P^{min} = P_{init}^{min}$, $p = |POS|$, $n = |NEG|$.*
3. *For $i = 1$ to p {*
   $FEASIBLES = \{c \in POS | P^{min} + \Delta P_c^{min} \leq P^{max}\}$.
   If $FEASIBLES = \emptyset$ return NO SOLUTION POSSIBLE.
   $c^* = \arg\max_{c \in FEASIBLES} \Delta P_c^{max} - \Delta P_c^{min}$.
   $chunk[i] = c^*$.
   $P^{max} = P^{max} + \Delta P_{c^*}^{max}$, $P^{min} = P^{min} + \Delta P_{c^*}^{min}$.
   $POS = POS - \{c^*\}$.
   }
4. *For every c in NEG: $P^{max} = P^{max} + \Delta P_c^{max}$, $P^{min} = P^{min} + \Delta P_c^{min}$.*
5. *For $i = n + p$ down to $p + 1$ {*
   $FEASIBLES = \{c \in NEG | P^{min} \leq P^{max} - \Delta P_c^{max}\}$.
   If $FEASIBLES = \emptyset$ return NO SOLUTION POSSIBLE.
   $c^* = \arg\max_{c \in FEASIBLES} \Delta P_c^{min} - \Delta P_c^{max}$.
   $chunk[i] = c^*$.
   $P^{max} = P^{max} - \Delta P_{c^*}^{max}$, $P^{min} = P^{min} - \Delta P_{c^*}^{min}$.
   $NEG = NEG - \{c^*\}$.
   }
6. *Return the vector "chunk". First chunk to be delivered is in "chunk[1]".*

Step 3 of the algorithm sequences the chunks with positive $\Delta P_c^{max} - \Delta P_c^{min}$ in a forward passing greedy manner to try to increase P^{max} as much as possible while increasing P^{min} as little as possible. Intuitively, the algorithm tries to maximize the range of possible safe prices at each x. Step 4 just computes P^{max} and P^{min} at the end of the whole sequence of chunks. Step 5 makes a greedy backward pass. It tries to allocate the chunks with negative $\Delta P_c^{max} - \Delta P_c^{min}$ so as to use as little as possible of the beneficial difference $\Delta P_c^{max} - \Delta P_c^{min}$ in the end of the sequence. Intuitively, this difference is saved for the middle of the sequence, from where it has time to affect more chunks (lying later in the sequence). The entire algorithm runs in time $O(|C|^2)$.

To attack our sequencing problem, we tried several greedy algorithms starting with the most intuitive ones. Most of them do not guarantee that the algorithm will find a safe sequence even if one exists. For example, the algorithms that greedily pass only forward and maximize $\Delta P_c^{max} - \Delta P_c^{min}$ or minimize ΔP_c^{min} can be defeated by counterexamples with just two chunks. Our algorithm cannot be defeated:

Theorem 5.1 *Algorithm 5.1 finds a safe ordering if one exists.*

Proof. Let us call the sequence of goods that algorithm 5.1 suggests s. In case the algorithm terminated saying "NO SOLUTION POSSIBLE", s is only a partial sequence. Assume (for contradiction) that $\exists s'$ s.t. s' is a safe sequence (i.e. acceptable), but s is not. So s' fulfills the requirement (theorem 2.1) that for all subsequent x and x', $P^{max}(x) \geq P^{min}(x')$, but s does not.

First, we prove that s' can be reordered (while maintaining acceptability) s.t. the chunks with $\Delta P_c^{max} - \Delta P_c^{min} \geq 0$ lie before those with $\Delta P_c^{max} - \Delta P_c^{min} < 0$. Take any pair of chunks c_i, c_j s.t. $\Delta P_{c_i}^{max} - \Delta P_{c_i}^{min} < 0$, $\Delta P_{c_j}^{max} - \Delta P_{c_j}^{min} \geq 0$, and c_i is assigned right before c_j in s'. Now we can swap c_i and c_j. This will not affect $P^{max}(x)$ or $P^{min}(x)$ before or after the pair. c_j can be moved before c_i, because c_i only made $P^{max}(x) - P^{min}(x)$ smaller. c_i can be moved after c_j, because c_j can only make $P^{max}(x) - P^{min}(x)$ larger (or same). We apply these swaps until the desired property holds. Let us call this new sequence s''.

Let us denote the sequence of chunks with $\Delta P_c^{max} - \Delta P_c^{min} \geq 0$ in s by s_{pos}, and the corresponding part of sequence s'' by s''_{pos}. We show that s''_{pos} can be converted into s_{pos} without loosing acceptability. Let i be the position at which s_{pos} and s''_{pos} first differ. Let s_i be the item in s_{pos} at position i, and let s''_i be the item in s''_{pos} at position i. Let $k > i$ be the position, where s_i was allocated in s''_{pos}. Now we can swap s''_i and s''_k in s''_{pos}. s''_k can be moved up to position i, because our algorithm allocated it there after having checked feasibility. s''_i can be moved down to position k, because the chunks in s''_{pos} between i and k could only have increased $P^{max}(x) - P^{min}(x)$. The presented swap can only increase $P^{max}(x) - P^{min}(x)$ for position between i and k, so no chunk that was feasible earlier can become infeasible. We can apply these swaps until $s''_{pos} = s_{pos}$.

Let us denote the sequence of chunks with $\Delta P_c^{max} - \Delta P_c^{min} < 0$ in s by s_{neg}, and the corresponding part of sequence s'' by s''_{neg}. Now we show that s''_{neg} can be converted into s_{neg} without loosing acceptability. Let i be the last position at which s_{neg} and s''_{neg} differ. Let s_i be the item in s_{neg} at position i, and let s''_i be the item in s''_{neg} at position i. Let $k < i$ be the position, where s_i was allocated in s''_{neg}. Now we can swap s''_i and s''_k in s''_{neg}. s''_k can be moved down to position i, because our algorithm allocated it there

after having checked feasibility. s''_i can be moved up to position k, because the chunks in s''_{neg} between k and i could only have decreased $P^{max}(x) - P^{min}(x)$. The presented swap can only increase $P^{max}(x) - P^{min}(x)$ for position between k and i, so no chunk that was feasible earlier can become infeasible. We can apply these swaps until $s''_{neg} = s_{neg}$.

So, s is a safe sequence iff s'' is. Earlier we showed that s'' is a safe sequence iff s' is. In other words, s is a safe sequence iff s' is. This contradicts our assumption and completes the proof. □

Sometimes the division of the entire exchange into chunks is not fixed a priori, but can be decided by the agents - eg. at contract time. A top down method for doing this is to generate a chunking and then test its safety by running the presented algorithm. If it is not safe, the chunking can be refined by splitting chunks further. Splitting is nicely monotonic in the sense that no split can make a safe exchange unsafe. The top down method can be used for continuous goods also. The minus side of the top down approach is the need to guess the splits. If they are guessed badly, possibly many more chunks are generated than is necessary to enable safe exchange. A bottom up approach for chunking is to take the smallest possible atomic chunks and sequence them using our algorithm. Next, each agent can see how many atomic chunks they can deliver at once at each step without changing the order and while still keeping the exchange safe. Bottom up chunking requires no guessing of splits, but it can be computationally complex if the number of smallest possible chunks is large. Also, it cannot be applied to continuous goods, because there chunks can be made arbitrarily small.

6 Ordering of discrete interdependent goods

In the previous chapter we looked at sequencing independent chunks. Sometimes partial deliveries are interdependent, though. The value of a chunk may depend on which chunks have been delivered before it. For example, in a production plant, the first products can be thought of as more costly than subsequent similar products, because later on the fixed costs (eg. rent, acquired equipment) can be spread among more products than in the beginning. Similarly, a data retrieval agent may incur large costs in searching for certain information. Once the information is found, subsequent searches of related information may be considerably less expensive.

The demander agent may also value a chunk differently depending on the other chunks delivered so far. In TRACONET (see section 2.1), the chunks (transport tasks) were interdependent for both the supplier and the demander. Transporting a parcel often affects the marginal cost of transporting others. For example, one vehicle may be able to carry two parcels to adjacent locations, thus reducing the marginal cost of each single task. On the other hand, one parcel may fill up the vehicle so that another task must be handled by a more costly vehicle. Some contracts involved multiple tasks. So if the safe exchange mechanisms of this paper had been used, sequencing of interdependent chunks would have been required. This was not crucial, because the agents represented real world dispatch centers, whose contracts were enforced by law.

In general, interdependent goods cannot be sequenced in polynomial time in the number of chunks if it is required that a safe solution is found if one exists. Just representing

the problem requires $O(2^{|C|})$ space, because for each set of chunks in the power set of all chunks, P^{max} and P^{min} have to be represented. Anyhow, if the number of chunks per contract is low (as in TRACONET), exponential search among sequences of chunks is viable. In such cases, the advantages of a robust exchange protocol without enforcement outweigh the extra computational load.

7 Renegotiation risk

So far we have presented mechanisms for motivating both agents to cooperate in an exchange given a contract. After an irrevocable delivery (or payment), the agent that gained from it may want to renegotiate the contract. For example, after the first partial delivery the demander may want to renegotiate the contract for a lower price. The demander knows that the original contract price was safe for the supplier, so now that the supplier has already "lost" the first delivery, the supplier should be willing to carry out the rest of the exchange at a lower price. On the other hand, the supplier knows that any point in the exchange is safe for the demander. Therefore, if the supplier refuses to renegotiate, the demander is motivated to follow the original contract and not vanish.

Renegotiation is more likely in unsafe exchange. For example, when an international company initiates a mining venture in a developing country, it has to invest most of the capital up-front. This unsafe move motivates the developing country to renegotiate the conditions of the mining venture (profit division etc.). Due to expropriation risk the company cannot avoid renegotiation [7, 8].

8 Conclusions and future research

This paper presented a method for carrying out mutual exchanges among self-motivated agents without third party enforcement. Larger exchanges were split into smaller parts so that at no point was either agent motivated to defect. The maximum size delivery that the supplier can safely make at any point in the exchange was calculated as well as the maximum amount that the demander can safely pay at any point. Safe exchange is possible if the demander's value function does not cumulate too fast compared to the supplier's. So, the possibility that a supplier agent incurs most of its cost from the early portion of the exchange enhances safe exchange, while the possibility that the demander acquires most of its value already from the early parts hinders safe exchange.

Considered in isolation, safe exchange can be carried out entirely only if the supplier can deliver the last part without cost. With continuous goods it can be carried out arbitrarily close to completion even if this is not the case. Considering defection's adverse effect on future negotiations often enables completing the current exchange.

Under the presented conditions on their discount functions, agents are motivated to carry out the exchange as fast as they can. These conditions are less strict for isolated exchange than for non-isolated exchange. In the non-isolated case, immediate moves can be forced by deadlines or lateness penalties.

Some domains allow goods to be delivered in different orders. The presented quadratic algorithm finds a safe ordering for independent goods if one exists. The problem cannot be

solved in polynomial time for interdependent goods. Finally, we showed that safe exchange helps prevent unfair renegotiation.

In this paper we looked at totally safe exchanges, where each agent knew its opponent's value function, discount functions, and defection penalty (i.e. cost of making reputation worse). We explained how agents could use bounds for these if they are not exactly known. If the bounds were too far off, even possible exchanges were disabled. Often it is the case that agents can estimate a distribution for each of these, although strict bounds are not available or they are too far off. Using these distributions the agents can take a calculated risk of making moves that are unsafe with a certain probability. This approach of using distributions is also useful to the agent in trying to model the possibility of changes in the opponent's value function, discount functions or defection penalty that happen during the exchange due to the opponent interacting in its environment (getting other offers, contracts etc.).

Another approach is to try to bound ones losses by making the partial exchanges small enough so that even if the opponent defects, the loss will be within a bound. In both the probabilistic risk method and the loss bounding method there is a tradeoff between making the exchange safer by using small partial exchanges and minimizing partitioning costs by using large ones. Finally, either a probabilistic approach or a loss bounding approach can be used to address the risk of the opponent accidentally defecting - eg. loosing contact due to a technical fault.

Acknowledgements

We thank Neil Immerman for the backward pass idea for algorithm 5.1, John Oliensis for helping prove theorem 4.1, and Herbert Gintis for game theoretic comments.

References

[1] Axelrod, R. 1984. The Evolution of Cooperation. BasicBooks.

[2] Durfee, E., Lee, J. and Gmytrasiewicz, P. 1993. Overeager Reciprocal Rationality and Mixed Strategy Equilibria. In *Proc. Eleventh National Conference on Artificial Intelligence*, pp. 225-230, Washington D.C.

[3] Durfee, E., Lesser, V. and Corkill, D. 1989. Cooperative Distributed Problem Solving. In: Barr, A., Cohen, P., and Feigenbaum, E. (eds.). *The Handbook of Artificial Intelligence Vol. IV*. Addison Wesley.

[4] General Magic, Inc. 1994. Telescript Technology: The Foundation for the Electronic Marketplace. General Magic White Paper.

[5] Kraus, S. and Wilkenfeld, J. 1992. Multiagent Negotiation Under Time Constraints. Comp. Sci. Tech. Report Series CS-TR-2975, University of Maryland, College Park, Maryland.

[6] Kreps, D. 1990. A course in microeconomic theory. Princeton University Press.

[7] Lax, D. A. and Sebenius, J. K. 1981. Insecure Contracts and Resource Development. *Public Policy* 29:4, pp. 417-436.

[8] Raiffa, H. The Art and Science of Negotiation. Harvard University Press, 1982.

[9] Sandholm, T. 1993. An Implementation of the Contract Net Protocol Based on Marginal Cost Calculations. In *Proc. Eleventh National Conference on Artificial Intelligence*, pp. 256-262, Washington D.C.

[10] Sandholm, T. and Lesser, V. 1994. An Exchange Protocol without Enforcement. In *Proc. 13th International Workshop on Distributed Artificial Intelligence*, Washington.

[11] Wellman, M.P. 1992. A General Equilibrium Approach to Distributed Transportation Planning. In *Proc. Tenth National Conference on Artificial Intelligence*, pp. 282-289, San Jose, California.

[12] Zlotkin, G. and Rosenschein, J.S. 1993. A Domain Theory for Task Oriented Negotiation. In *Proc. Thirteenth International Joint Conference on Artificial Intelligence*, pp. 417-442, Chamberry, France.