Multiagent Commitment Alignment

Amit K. Chopra
University of Trento, Italy
akchopra.mail@gmail.com

Munindar P. Singh
North Carolina State University, USA
singh@ncsu.edu

ABSTRACT

Commitments provide a basis for understanding interactions in multiagent systems. Successful interoperation relies upon the interacting parties being aligned with respect to their commitments. However, alignment is nontrivial in a distributed system where agents communicate asynchronously and make different observations. We propose a formalization for commitments that ensures alignment despite asynchrony. This formalization consists of three elements: (1) a semantics of commitment operations; (2) messaging patterns that implement the commitment operations; and (3) weak constraints on agents' behaviors to ensure the propagation of vital information. We prove that our formalization ensures alignment. We illustrate the generality of our formalization with several real-life scenarios.

Categories and Subject Descriptors

D.2.12 [**Software Engineering**]: Interoperability; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent Systems*

General Terms

Theory

Keywords

Commitments, Alignment, Autonomy, Asynchrony

1. INTRODUCTION

$\mathsf{C}(debtor, creditor, antecedent, consequent)$ means the debtor commits to the creditor that if antecedent holds, then the consequent will hold. An important insight in agent communication is that the interactions among agents may be understood in terms of their effects on the agents' commitments. For example, an offer for a copy of the book *Beating the Odds* from Bookie to Alice may be interpreted as $\mathsf{C}(Bookie, Alice, \$12, BeatingtheOdds)$. In other words, Bookie commits to Alice that if Alice pays \$12, then Bookie will deliver the book.

Imagine if Alice presumes that Bookie is committed to sending her the book she paid for, but Bookie is not committed to sending her the book. Their interaction would break down. In general, a key requirement for successful interaction is that the interacting agents remain aligned with respect to their commitments. *Crucially, it turns out that even well-designed, well-behaved agents may become misaligned simply because of the distributed nature of the given system.* Previous approaches have largely ignored this problem or addressed it through restrictive, ad hoc assumptions. However, as commitment protocols expand into real-life distributed settings, a rigorous treatment becomes essential.

We consider realistic, distributed settings where agents communicate via asynchronous messaging. Asynchrony means that an agent is never blocked from sending a message. In such a system, the messages that the agents send each other may cross on the wire. Thus, in general, the agents may observe different messages in different orders. Since messages are understood in terms of their effects on commitments, the agents involved would become misaligned, i.e., come to conflicting conclusions about which commitments hold and which do not.

It is crucial to develop a formalization of commitments that ensures alignment despite asynchrony. First, distributed computing infrastructure is necessarily asynchronous. Large-scale systems exhibit high latency making synchronous interactions simply intractable in practice. Second, any formalization that works despite asynchrony also works in "more synchronous" settings, that is, those imposing additional constraints on agent behavior—for example, one where agents take turns sending messages. Third, asynchrony is inherently compatible with agent autonomy simply because an agent is never blocked from sending a message and, more pertinently, from acting upon its commitments.

In the absence of a formalization that supports reasoning about commitments in distributed settings, all research in applications of commitments is bound to report results that are either not general enough or are unduly complex. Such a formalization is currently missing; this paper seeks to fill this gap.

Motivation. Informally, we say that agents are aligned, if whenever an agent infers a commitment in which it is the creditor, the debtor of the commitment also infers that commitment. There are two possible causes of misalignment. One, the agents may assign incompatible meanings to the messages they are exchanging. Two, even when the agents assign identical meanings to the relevant messages, they may make incompatible observations. Chopra and Singh [2] solve the former for a language similar to ours. This paper addresses the second problem. Let's consider some examples to highlight the problem.

EXAMPLE 1. *(Figure 1(A)). Bookie sends Alice (a message that expresses) an offer that if she pays \$12, then Bookie will deliver to her a copy of the book* Beating the Odds. *Alice sends Bookie a rejection of the offer. Upon receipt, Bookie resends the offer.* ∎

As is typical in commitment protocols, Bookie's offer creates a commitment from Bookie to Alice for the book *Beating the Odds* in return for \$12. In Example 1, both Alice and Bookie observe the messages in the same order, and therefore remain aligned.

Cite as: Multiagent Commitment Alignment, Amit K. Chopra and Munindar P. Singh, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. XXX-XXX.

EXAMPLE 2. *(Figure 1(B)). Bookie makes Alice an offer. Not seeing a response from Alice, Bookie resends the offer. Suppose that, in the meantime, Alice sends Bookie a rejection of the offer. Then the rejection crosses Bookie's repetition of the offer.* ▮

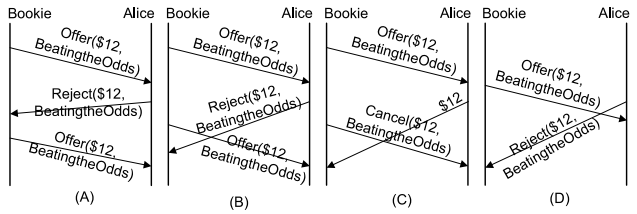

Figure 1: Scenarios (B),(C), and (D) end in misalignment

What ought Bookie and Alice to infer about the offer at the end of the exchange shown in Figure 1(B)? After seeing Alice's rejection of the offer, Bookie may infer that there no longer exists an offer to Alice. However, having seen an offer message last, Alice may infer that the offer holds. That is, Alice infers a commitment from Bookie for a copy of *Beating the Odds* for $12, whereas Bookie does not infer that commitment. This misalignment occurs because Alice's rejection and Bookie's offer messages crossed in transit. Note that Figures 1(A) and 1(B) imply a race condition between offer and rejection: their order (as viewed by Bookie) matters and yet Alice cannot distinguish between the two orders.

EXAMPLE 3. *(Figure 1(C)). Bookie makes an offer that Alice accepts and sends the payment for. In the meantime, Bookie cancels the offer. Bookie's cancellation and Alice's payment cross.* ▮

In Example 3, upon sending the payment, Alice infers that Bookie is committed to sending her a copy of the book. Later, when Alice sees Bookie's cancel message, she regards it as spurious. However, Bookie sees the payment only after he has canceled its offer. So Bookie considers Alice's payment late. The result is that Alice infers an unconditional commitment for the book from Bookie, but one that Bookie does not infer. A race between cancellation and payment causes misalignment.

EXAMPLE 4. *(Figure 1(D)) Here, Bookie sends an offer, but in the meantime Alice sends a rejection.* ▮

In the scenario in Example 4, Bookie infers the offer was rejected because that is the message it last sees, whereas Alice infers the offer exists because that is the message she last sees. Admittedly, the scenario is pathological: it makes no sense for Alice to reject an offer that Bookie never made. However, scenarios where messages arrive unexpectedly can occur when multiple parties are involved, and messages happen to be delayed differently on different paths. This is analogous to when one receives a group reply to an email before receiving the original email.

As the above examples demonstrate, asynchrony throws a major challenge in the face of alignment. Even agents who are perfectly designed and who assign identical meanings to messages may end up misaligned. Another way to cast this problem is in terms of the commitment operations, which show how to manipulate commitments [12]. Existing formalizations of the operations, e.g., [3], do not support reasoning in distributed settings.

Current approaches for alignment fall into two main categories. Some use acknowledgments [8] as a way of serializing the operations in distributed settings. The idea is that the agents involved would observe the relevant messages in the same order, and hence make the same inferences. Such approaches are incompatible with autonomy. *Autonomy compatibility* means that no agent should have to wait for approval from other agents to effect a change in its commitments. In an acknowledgment-based approach, for example, to effect a cancellation or discharge of a commitment, the debtor would have to seek the creditor's approval, which completely begs intuition.

Others suggest commitments of the form $\mathsf{C}(id, x, y, r, u)$, where id is a unique identifier termed as the *commitment identifier* [4, 11]. Commitment operations would then reference these identifiers. Commitment identifiers fail to meet *semanticity*. Semanticity means that the proposal should accommodate general reasoning about commitments. For example, with identifiers, if $\mathsf{C}(id_0, x, y, r, u)$ and $\mathsf{C}(id_1, x, y, r, v)$ hold, semantically it still ought to be the case that $\mathsf{C}(\_, x, y, r, u \wedge v)$ holds ('_' is some identifier). To reason with identifiers, one would need to track dependencies for commitments a la distributed truth maintenance [6]. Any such approach would be more complex than the approach presented here, without being more general.

Contributions. Our primary contribution is a formalization consisting of three elements: (1) messaging patterns that communicate the commitment operations; (2) a semantics of the operations that determines each participating agent's inferences regarding commitments; and (3) constraints on agent behavior described as messages the agents must send under specific circumstances. We prove that our formalization eliminates misalignments, and illustrate its intuitiveness and generality with the help of various examples. A noteworthy feature of our formalization is that it does not involve computing global system states [7] and then detecting misalignments; the formalization guarantees alignment without any coordination whatsoever between agents.

Our formalization is both autonomy compatible and semantic. In particular, our formalization does not rely upon using commitment identifiers as introduced above. Later in the paper, we show how *domain identifiers* may be used, if necessary.

Organization. The structure of this paper is as follows. Section 2 discusses commitments. Section 3 introduces the principles of our approach. Section 4 presents a formal model of communication and defines alignment. Section 5 formalizes the principles and proves that alignment is guaranteed for all possible multiagent executions. Section 6 discusses related work and summarizes our contributions.

2. COMMITMENTS

Below, x,y, etc are variables over agents; p, q, r, etc. are propositional variables; $\vee$, $\wedge$, $\neg$, $\rightarrow$, $\leftrightarrow$ are the usual propositional connectives; $\top$ and $\bot$ are the constants for truth and falsity, respectively; $\vdash$ is the usual propositional inference symbol. Read $\Rightarrow$ as *implies*.

A commitment is of the form $\mathsf{C}(x, y, r, u)$. If r holds, then $\mathsf{C}(x, y, r, u)$ is *detached*, and the commitment $\mathsf{C}(x, y, \top, u)$ holds. If u holds, then the commitment is *discharged* and doesn't hold any longer. All commitments are *conditional*; an unconditional commitment is merely a special case where the antecedent equals $\top$. Reasoning postulates for commitments are reproduced below [13]. For brevity, we omit the agents when they can be understood from the context. Further, when the postulates uniformly use the debtor x and creditor y, we write $\mathsf{C}(r, u)$ instead of $\mathsf{C}(x, y, r, u)$.

B1. DISCHARGE. $u \rightarrow \neg\mathsf{C}(r, u)$

B2. DETACH. $\mathsf{C}(r \wedge s, u) \wedge r \rightarrow \mathsf{C}(s, u)$.

B3. AUGMENT. From $\mathsf{C}(r, u) \wedge s \vdash r$, infer $\mathsf{C}(s, u)$

B4. L-DISJOIN. $\mathsf{C}(r, u) \wedge \mathsf{C}(s, u) \rightarrow \mathsf{C}(r \vee s, u)$

B5. R-CONJOIN. $\mathsf{C}(r, u) \wedge \mathsf{C}(r, v) \rightarrow \mathsf{C}(r, u \wedge v)$

B6. CONSISTENCY. $\neg\mathsf{C}(r, \bot)$

B7. NONVACUITY. From $r \vdash u$, infer $\neg\mathsf{C}(r, u)$

B8. WEAKEN. $\mathsf{C}(r, u \wedge v) \wedge \neg u \rightarrow \mathsf{C}(r, u)$

Notice that B1 covers the discharge of commitments. B2 generalizes their detach. *Semanticity* means that alignment must not fail in the face of reasoning postulates B1–B8. That is, we must make sure that the effects of the various messages on commitments are consistent with respect to the above postulates.

The commitment operations are reproduced below (from [12]). CREATE, CANCEL, and RELEASE are two-party operations, whereas DELEGATE and ASSIGN are three-party operations.

CREATE(x, y, r, u) is performed by x, and it causes $\mathsf{C}(x, y, r, u)$ to hold. CANCEL(x, y, r, u) is performed by x, and it causes $\mathsf{C}(x, y, r, u)$ to not hold. RELEASE(x, y, r, u) is performed by y, and it causes $\mathsf{C}(x, y, r, u)$ to not hold. DELEGATE(x, y, z, r, u) is performed by x, and it causes $\mathsf{C}(z, y, r, u)$ to hold. ASSIGN(x, y, z, r, u) is performed by y, and it causes $\mathsf{C}(x, z, r, u)$ to hold.

Let us define the set of messages that agents can exchange. Let Φ be a finite set of atomic propositions $\phi_0, \ldots, \phi_i$ (commitments are not atomic propositions). $Inform(x, y, p)$ is a message from x to y, where p is a conjunction over Φ. In the commitment operations, r is a DNF formula over Φ (for example, $(\phi_0 \wedge \phi_1) \vee (\phi_3 \wedge \phi_4)$), and u is a CNF formula over Φ (for example, $(\phi_0 \vee \phi_1) \wedge (\phi_3 \vee \phi_4)$). $Create(x, y, r, u)$ and $Cancel(x, y, r, u)$ are messages from x to y; $Release(x, y, r, u)$ from y to x; $Delegate(x, y, z, r, u)$ from x to z; and $Assign(x, y, z, r, u)$ from y to x. Suppose $c = \mathsf{C}(x, y, r, u)$. Then $Create(c)$ stands for $Create(x, y, r, u)$. We similarly define $Delegate(c, z)$, $Assign(c, z)$, $Release(c)$, and $Cancel(c)$.

All atomic propositions are stable, that is, if an atomic proposition holds, it holds forever. In English, stability corresponds to the perfective aspect, for example, *book has been delivered, payment has been made*, and so on [13]. Propositions with explicit time, such as *the book is delivered by 3PM* are also stable. Thus each atomic proposition corresponds to the occurrence of an *event*: when the proposition holds, the corresponding event is said to have occurred. A commitment, however, is not a stable proposition. A commitment may come to not hold because it was discharged, cancelled, or released, leaving the agents sensitive to race conditions over commitments.

Below, let $c_B = \mathsf{C}(Bookie, Alice, \$12, BeatingtheOdds)$;
$c_G = \mathsf{C}(Bookie, Alice, \$12, GamblingTips)$;
$c_0 = \mathsf{C}(Bookie, Alice, \$12, BeatingtheOdds \wedge GamblingTips)$;
$c_1 = \mathsf{C}(Bookie, Alice, \$12 \vee coupon, BeatingtheOdds)$;
$c_2 = \mathsf{C}(Bookie, Alice, \$12 \wedge coupon, BeatingtheOdds)$. Intuitively, c_0 is a stronger commitment than c_B (an additional book for the same price); c_1 is stronger than c_B (two ways to obtain a book instead of one); c_B is stronger than c_2 (fewer conditions need to be satisfied to obtain a book). Definition 1 captures this intuition.

DEFINITION 1. *$\mathsf{C}(x, y, r, u)$ is stronger than $\mathsf{C}(x, y, s, v)$, denoted by $\mathsf{C}(x, y, r, u) \succeq \mathsf{C}(x, y, s, v)$, iff $s \vdash r$ and $u \vdash v$.*

Thus, for example, $c_0 \succeq c_B$. If $\mathsf{C}(x, y, r, u) \succeq \mathsf{C}(x, y, s, v)$ but $\mathsf{C}(x, y, s, v) \not\succeq \mathsf{C}(x, y, r, u)$, we say $\mathsf{C}(x, y, r, u) \succ \mathsf{C}(x, y, s, v)$. B3 and B8 capture the notion of strength deductively. For example, if c_1 holds, then by B3, c_B holds as well. Similarly, if c_0 holds, then by B8, c_B holds as well—unless $BeatingtheOdds$ holds already in which case according to B1, c_B cannot hold.

3. PRINCIPLES OF ALIGNMENT

The misalignments in Figure 1 are due to the naïve semantics that upon observing $Create(r, u)$, an agent infers $\mathsf{C}(r, u)$; upon observing $Release(r, u)$ or $Cancel(r, u)$ an agent infers $\neg\mathsf{C}(r, u)$.

We propose five principles that guarantee alignment. These principles are informed by the nature both of commitments and of distributed systems. Let us first consider three principles that address the misalignments in Figure 1.

NOVEL CREATION. Observing $Create(r, u)$ should have no effect if a stronger commitment $\mathsf{C}(s, v)$ has held before.

COMPLETE ERASURE. Observing $Release(r, u)$ should have no effect if a *strictly* stronger commitment $\mathsf{C}(s, v)$ holds. If no such $\mathsf{C}(s, v)$ holds, then each weaker commitment $\mathsf{C}(r', u')$ is released. $Cancel(r, u)$ is analogous.

ACCOMMODATION. Observing $Release(r, u)$ has the effect that each weaker commitment $\mathsf{C}(s, v)$ is treated as if it has held before. $Cancel(r, u)$ is analogous.

Figure 3(B) exemplifies our graphical notation. We represent an execution as a sequence diagram. Each point where a message is sent or received is annotated with the commitments that hold immediately after the observation; commitments that do not hold are not shown. If $\mathsf{C}(r, u)$ holds and $\mathsf{C}(r, u) \succeq \mathsf{C}(s, v)$, we only show $\mathsf{C}(r, u)$. Each agent's vertical line may be annotated at the top to indicate initial conditions of the interaction.

Figure 2 shows how these principles restore alignment to the misaligned scenarios of Figure 1. Figure 2 shows *offer* as $Create(c_B)$, and *reject* as $Release(c_B)$.

Contrast Figures 1(A) and 2(A). In both figures, Bookie and Alice remain aligned at the end. However, in Figure 1(A), Bookie and Alice both infer c_B, whereas in Figure 2(A), neither of them infers c_B. NOVEL CREATION supports Figure 2(A): the first offer causes c_B to hold and resending the offer after receiving a reject has no effect.

Contrast Figures 1(B) and 2(B). In Figure 1(B), in the end, Alice infers c_B, whereas Bookie does not. In Figure 2(B), however, neither Alice nor Bookie infers c_B. Upon receiving the reject, because of COMPLETE ERASURE, Bookie considers himself released from the offer; receiving the same offer again has no effect on Alice because of NOVEL CREATION.

Contrast Figures 1(D) and 2(C). In Figure 1(D), in the end, Alice infers c_B, whereas Bookie does not. In Figure 2(C), however, neither Alice nor Bookie infers c_B. Upon receiving the reject, because of COMPLETE ERASURE, Bookie considers himself released from the offer; receiving an offer which Alice has already rejected has no effect on Alice because of ACCOMMODATION and NOVEL CREATION acting in concert. ACCOMMODATION ensures that Alice's release of the offer makes it appear as if the offer had been made before, and hence when Bookie's actual offer arrives, NOVEL CREATION ensures the offer has no effect.

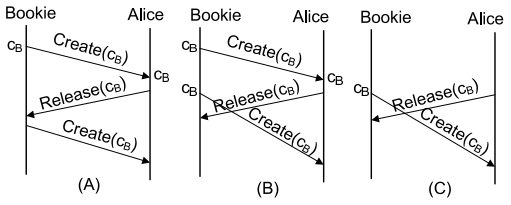

Figure 2: Proposed approach

NOVEL CREATION means that resending a $Create$ of a previous commitment has no effect. In that case, how can Bookie again offer Alice essentially the same deal that she has rejected before? Circumstances might have changed, and Bookie might want to see if Alice will accept the offer this time around.

A possible domain modeling approach is to include identifiers on the conditions involved so as to distinguish the offers. In practice, we would place such identifiers anyway, so as to distinguish com-

mitments made to different parties, e.g., to ensure that a different copy of the book would be delivered to each customer and each customer will pay for her purchase. Such identifiers are distinct from commitment identifiers: they do not apply on commitments and do not interfere with reasoning about commitments. In Example 5, at the end, both Alice and Bookie infer that the id_1 commitment holds and the id_0 commitment doesn't.

EXAMPLE 5. *Bookie sends* $Create(Bookie, Alice, \$12(id_0), BeatingtheOdds(id_0))$. *Alice sends* $Release(Bookie, Alice, \$12(id_0), BeatingtheOdds(id_0))$. *To offer the "same" deal again, Bookie sends* $Create(Bookie, Alice, \$12(id_1), Beatingthe Odds(id_1))$. ∎

Notice that NOVEL CREATION does not say that if a commitment has held before, then it can never hold again; it only says that a $Create$ message for such a commitment has no effect. A commitment may come to hold again because a $Create$ message for a stronger commitment is observed. In real life, it is common practice for a seller to improve its offers, effectively making stronger commitments, as in Example 6.

EXAMPLE 6. *Bookie makes Alice the offer* c_B. *Alice rejects the offer thus releasing Bookie from* c_B. *However, Bookie is persistent, and he makes Alice the stronger offer* c_0 *(two books for the same price). This automatically resurrects* c_B *to ensure consistency.* ∎

EXAMPLE 7. *Alice rejects Bookie's improved offer.* ∎

When Alice sends $Release(c_0)$, COMPLETE ERASURE means that this not only removes c_0, but also c_B and c_G. Notice that partial releases are unsuccessful. Because c_0 is stronger than c_B, $Release(c_B)$ has no effect—c_0 continues to hold.

NOTIFICATION. This principle ensures that two agent's states are compared only when both or neither has received vital information. This leads to two requirements. One, the creditor of a commitment must notify the debtor of a detach, and the debtor must notify the creditor of discharge. Two, until an agent sends its pending notifications, it doesn't have a well-defined visible state. Reducing the visible states proves crucial because we can define alignment as agreement between the concerned agents at such states.

Consider Figure 3(A). Initially, Alice is committed to Bob that if the sky is clear, then she will meet him at the lake, meaning $c_L = \mathsf{C}(Alice, Bob, clear, lake)$. We model Bob's observation of the sky as a message that Bob receives from the environment Env. Now, Bob infers the unconditional commitment $c_{UL} = \mathsf{C}(Alice, Bob, \top, lake)$ whereas Alice does not yet infer c_{UL} (maybe because she is in a basement and cannot look at the sky). Thus, Bob and Alice would be misaligned. The main problem is that Bob has received some vital information that Alice does not have.

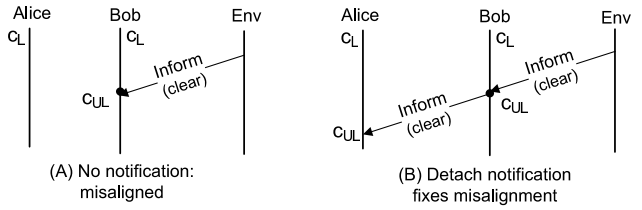

Figure 3: Notifying about detaches

Figure 3(B) shows how alignment is preserved. The bold dot along Bob's lifeline indicates that Bob must send the $clear$ notification to Alice. The middle state where Bob has detached the commitment but not notified Alice is excluded from consideration—it is not visible for the purposes of alignment. In this manner, we avoid a false negative claim about alignment. This case is of a creditor notifying the debtor of a detach. The case where a debtor notifies a creditor of a discharge is similar.

PRIORITY. It is possible that a debtor cancels a commitment concurrently with the creditor detaching it. Recall Example 3 where Alice's payment crosses Bookie's cancellation. Figure 4(A) annotates the same example with commitments. If Bookie's cancellation and Alice's payment cross, Alice and Bookie become misaligned—Alice infers $c_U = \mathsf{C}(Bookie, Alice, \top, BeatingtheOdds)$ whereas Bookie does not. The reason is that receiving $Cancel(c_B)$ has no effect on Alice because she already infers c_U, which is a stronger commitment than c_B. Receiving Alice's \$12 payment has no effect on Bookie because there is no commitment to detach anymore.

There is no fundamental reason to prefer the creditor's or the debtor's viewpoint. For each commitment, the parties involved simply have to agree on what takes priority: cancel over detach, or detach over cancel. Detach priority means that the debtor considers its cancellation of a commitment to be overridden by the detach of the commitment. Cancel priority means that the creditor considers its detach of a commitment to be overridden by the debtor's cancellation of the commitment. Our theory handles both alternatives, and shows what the agents must do in each case. Consider a commitment $\mathsf{C}(r \wedge s, u)$. Suppose detach has priority over cancel. If the debtor observes a message that brings about s (a detach) after it has cancelled $\mathsf{C}(r \wedge s, u)$, then it must send $Create(r, u)$. Alternatively, suppose that cancel has priority over detach. If the creditor has already detached $\mathsf{C}(r \wedge s, u)$ by sending a message that brings about s, and it then observes a cancellation for $\mathsf{C}(r \wedge s, u)$, then the creditor must send $Release(r, u)$.

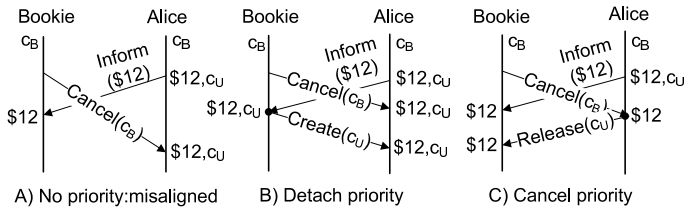

Figure 4: Race between cancel and detach

The protocol that Alice and Bookie are enacting would specify whether cancel or detach has priority for c_B. If detach has priority, then, as Figure 4(B) shows, Bookie considers its cancellation to be overridden by the detach, and creates c_U. If cancel has priority, then, as Figure 4(C) shows, Alice considers the detach to be overridden by the cancellation, and releases Bookie from c_U.

4. FORMALIZING ALIGNMENT

Alignment means that whenever an agent infers from its observations, a commitment in which it is the creditor, then the debtor must also infer the commitment from its own observations. An execution of a multiagent system is a progression of the system from one (system) state to another. Every time an agent sends or receives a message, the system progresses to a new state. We would like to consider all possible executions of the system; however, we need to ensure that alignment is considered only at well-defined milestones in any execution; otherwise, we would falsely claim misalignment. The appropriate milestones are expressed via *quiescence* and *integrity*.

A system state is quiescent if no messages are in transit. In considering only quiescent states, we ensure the agents are "synced" up when we verify their alignment. Without quiescence, alignment

is generally impossible because some agents may not yet have observed messages destined for them. Consider Figure 1(C) where Alice has sent $12 to Bookie but Bookie hasn't received the payment. Alice infers that Bookie is now committed to sending her the book, but Bookie has no clue of an incoming payment, and so isn't committed. At quiescence, Bookie would have received the payment. If even at quiescence, Alice and Bookie disagree, we have a problem on our hands. Quiescence may only be temporary, because the agents could be silently computing: it would end when an agent sends a message based on its internal computations.

We wish to exclude system states where an agent has received vital information that it hasn't yet propagated to relevant parties. In Figure 3(B), it would be premature to consider alignment before Bob notifies Alice of $clear$. In this sense, Bob's notifying Alice of $clear$ is integral with receiving $Inform(clear)$ from Env. Similarly, in Figure 4(B), Bookie sending the $Create$ is integral with receiving $Inform(\$12)$. We recognize no intervening states from the point of view of alignment until all integral observations have been made; in other words, the intervening states are not visible. We now turn to the formalization.

Communication. Agents communicate by messaging. Below m, m', m_0, ... are variables over messages. Assumptions A1–A4 model communication.

A1. Communication is *point-to-point*. Below $m(x, y)$ indicates a message m from x to y.

A2. An agent observes all and only those messages that it sends or receives. Observations are ordered *serially*. All observations pertain to messages. Observations of the environment are treated as messages from Env.

A3. Messaging is *reliable*. Messages are neither created nor destroyed by the infrastructure.

A4. Messaging is *ordered*. Any two messages sent by an agent to the same recipient are received in order.

An agent x's *observation sequence* $\langle m_0, \ldots, m_n \rangle_x$ describes the sequence of messages x observes in a particular execution. Let $\mathcal{A}$ be a system of k agents. Then, $O = [O_0, \ldots, O_{k-1}]$ is an *observation vector* over $\mathcal{A}$, where the O_is are the observation sequences, one for each of the k agents. Below, o is a variable over observation vectors; o_x, etc. are variables over a particular agent's observation sequence. A3 and A4 impose validity requirements on vectors.

DEFINITION 2. *An observation vector O over $\mathcal{A}$ is* valid *iff $\forall x, y \in \mathcal{A}$: (1) if $m(x, y)$ occurs in O_y, then $m(x, y)$ occurs in O_x; and (2) if $m_1(x, y)$ occurs in O_y, and $m_0(x, y)$ precedes $m_1(x, y)$ in O_x, then $m_0(x, y)$ precedes $m_1(x, y)$ in O_y.*

Conditions (1) and (2) in Definition 2 capture A3 and A4, respectively. This paper considers only valid observation vectors.

Think of an agent's observation sequence as representing the agent's state at the granularity of the interaction (i.e., ignoring aspects of the agent's state not reflected in its observations). Then an observation vector represents the state of the system. $\mathcal{O}_\mathcal{A}$, the set of all possible observation vectors for system $\mathcal{A}$, is the set of all possible executions of $\mathcal{A}$.

Quiescence. This means that there are no messages in transit in the system. Definition 3 states that an observation vector is *quiescent* if and only if every sent message has been received.

DEFINITION 3. *An observation vector $O \in \mathcal{O}_\mathcal{A}$ is quiescent iff $\forall x, y \in \mathcal{A}$, if $m(x, y)$ occurs in O_x, then $m(x, y)$ occurs in O_y.*

Integrity. We now show how to specify integrity constraints on observations. Section 5 specifies the integrity constraints relevant to alignment. First though, some preliminaries. Let O_x be an observation sequence of the form $\langle \ldots, m \rangle_x$. Then, for any message m', $O_x; m'$ is the concatenation of O_x with m', and is of the form $\langle \ldots, m, m' \rangle_x$. Let $\mathcal{S}(O_x)$ be the set of propositions that can be inferred from the observation sequence O_x. Section 5 formalizes $\mathcal{S}(O_x)$. The empty condition ε is trivially in $\mathcal{S}(O_x)$. $\mathcal{S}(O_x)$ may be thought of as the *state* of x after observing the messages in O_x.

$\lfloor m[B : A]m' \rfloor_x$ is an *integrity constraint* on the observations of agent x. Here, B and A are the *before* and *after* conditions for the *trigger* m, and m' is the *effect* of m if the *before* and *after* conditions are met.

DEFINITION 4. *Consider a constraint $\lfloor m[B : A]m' \rfloor_x$. m' is an enabled effect of m with respect to an observation sequence o and the constraint iff $B \in \mathcal{S}(o)$ and $A \in \mathcal{S}(o; m)$.*

An observation sequence O_x is integral *with respect to a set of constraints iff for any prefix $o; m$ of O_x, $o; m; M$ is a prefix of O_x, where M* contains *an interleaving of the enabled effects of m with respect to o and the set of constraints.*

An observation vector is integral *with respect to a set of constraints iff each observation sequence in it is integral with respect to the set of constraints.*

Definition 4 defines enabled messages as those that must be necessarily sent, as deduced from the integrity constraints. An observation sequence is not integral unless all enabled messages have been observed. Notice that to be integral, O_x must only *contain* the enabled effects (for every prefix); there is no restriction that the enabled effects must occur immediately after the trigger. This means that x may make extraneous observations between the trigger and its enabled effects; however, the system states corresponding to those observations are not visible for the purposes of alignment.

Alignment. Definition 5 formalizes the notion of alignment by considering all potential observations of all agents.

DEFINITION 5. *A multiagent system $\mathcal{A}$ is* aligned *(written $[\![\mathcal{A}]\!]$) iff $\forall O \in \mathcal{O}_\mathcal{A}$ such that O is quiescent and integral with respect to the integrity constraints, $\forall x, y \in \mathcal{A} : \mathsf{C}(x, y, r, u) \in \mathcal{S}(O_y) \Rightarrow \mathsf{C}(x, y, r, u) \in \mathcal{S}(O_x)$.*

Definition 5 considers the observations of creditors and debtors from the same integral and quiescent observation vectors. It says that if a creditor infers a commitment from its observations, then the debtor must infer that commitment from its own observations. When a debtor infers a commitment, but the creditor does not, no harm is done, and alignment is unaffected.

5. FORMALIZING THE PRINCIPLES

We introduce (nonatomic) propositions $created(x, y, r, u)$, $released(x, y, r, u)$, and $cancelled(x, y, r, u)$, each corresponding to the eponymous commitment operation having occurred. Our formalization does not require propositions corresponding to the occurrence of DELEGATE and ASSIGN. We adopt the postulates B9–B13 in addition to B1–B8.

B9. $released(r, u) \rightarrow created(r, u)$

B10. $cancelled(r, u) \rightarrow created(r, u)$

B11. $created(r, u)$ and $\mathsf{C}(r, u) \succeq \mathsf{C}(s, v) \Rightarrow created(s, v)$

B12. $released(r, u)$ and $\mathsf{C}(r, u) \succeq \mathsf{C}(s, v) \Rightarrow released(s, v)$

B13. $cancelled(r, u)$ and $\mathsf{C}(r, u) \succeq \mathsf{C}(s, v) \Rightarrow cancelled(s, v)$

Let's consider some examples to see how B9–B13 work. Suppose $created(c_0)$ holds; by B11, $created(c_B)$ and $created(c_G)$ hold. Suppose $released(c_0)$ holds; by B9, $created(c_0)$ holds too; by B12, $released(c_B)$ and $released(c_G)$ hold; by B9, $created(c_B)$ and $created(c_G)$ hold.

Let's see how B9–B13 relate to the principles introduced earlier. B12 and B13 relate to COMPLETE ERASURE. If a commitment

is released or if it is cancelled, all weaker commitments are released or cancelled, as may be the case. B9 and B10 (together with B12 and B13) portray ACCOMMODATION: if a commitment has been cancelled or released, treat all weaker commitments as if they had held. B11 relates to NOVEL CREATION. It ensures that once $created(r, u)$ holds, all commitments weaker than $\mathsf{C}(r, u)$ are also considered created.

Now we define the semantics of the operations themselves in terms of $\mathcal{S}(o)$, the set of propositions that can be inferred from the observation sequence o. For any set of propositions $\mathcal{Q}$, $\mathcal{Q}^*$ is the deductive closure of $\mathcal{Q}$. $\mathcal{Q}^\Pi$ is the *atomic projection* of $\mathcal{Q}$ such that a proposition q belongs to $\mathcal{Q}^\Pi$ if and only if two conditions are satisfied: (1) q belongs to $\mathcal{Q}$, and (2) q is either an atomic proposition, or of the form $\mathsf{C}(r, u)$, $created(r, u)$, $released(r, u)$, or $cancelled(r, u)$.

Let $\mathcal{S}(o_x)$ be the current state of x. The general pattern for computing the state $\mathcal{S}(o_x; m)$ is the following. First modify $\mathcal{S}(o_x)$ by adding or removing propositions relevant to m. Let $\mathcal{S}'(o_x; m)$ be the resulting set. $\mathcal{S}(o_x; m)$ is $(\mathcal{S}'(o_x; m)^*)^\Pi$, in other words, the atomic projection of the deductive closure of $\mathcal{S}'(o_x; m)$. Let us facilitate this pattern by introducing the notation $\mathcal{Q}^\odot$, the *atomic closure* of $\mathcal{Q}$, to mean $(\mathcal{Q}^*)^\Pi$.

B14 is the semantics of $Inform(r)$: r holds upon observing it.

B14. $\mathcal{S}(o; Inform(r)) = (\mathcal{S}(o) \cup \{r\})^\odot$

Two-Party Operations. The messages $Create(r, u)$, $Release(r, u)$, and $Cancel(r, u)$ realize the corresponding operations.

B15 and B16 give the semantics of $Create(r, u)$. B15 states that if $created(r, u)$ or the consequent u already hold, then upon observing $Create(r, u)$, we insert $created(r, u)$, and compute its atomic closure to obtain the resulting state. In particular, $\mathsf{C}(r, u)$ does not hold in the resulting state. The condition related to consequent u is present because the consequent of the commitment and the commitment both holding together is inconsistent according to B1. Hence, if u holds, $Create(r, u)$ has no effect. Conversely, B16 states that if neither $created(r, u)$ nor u holds in the current state, then upon observing $Create(r, u)$, we insert $\mathsf{C}(r, u)$ and $created(r, u)$, and compute the atomic closure to obtain the resulting state.

B15. $created(r, u) \in \mathcal{S}(o)$ or $u \in \mathcal{S}(o) \Rightarrow$
$\mathcal{S}(o; Create(r, u)) = (\mathcal{S}(o) \cup \{created(r, u)\})^\odot$

B16. $created(r, u) \notin \mathcal{S}(o)$ and $u \notin \mathcal{S}(o) \Rightarrow$
$\mathcal{S}(o; Create(r, u)) = (\mathcal{S}(o) \cup \{\mathsf{C}(r, u), created(r, u)\})^\odot$

Let $\lceil\lceil \mathsf{C}(r, u) \rceil\rceil$ denote the set $\{\mathsf{C}(s, v) | \mathsf{C}(r, u) \succeq \mathsf{C}(s, v)\}$, that is, the set of commitments weaker than $\mathsf{C}(r, u)$. According to B17, upon observing $Release(r, u)$, we remove all commitments weaker than $\mathsf{C}(r, u)$, insert $released(r, u)$, and then compute the atomic closure to obtain the resulting state. B18 analogously gives the semantics of $Cancel(r, u)$.

B17. $\mathcal{S}(o; Release(r, u)) =$
$((\mathcal{S}(o) \setminus \lceil\lceil \mathsf{C}(r, u) \rceil\rceil) \cup \{released(r, u)\})^\odot$

B18. $\mathcal{S}(o; Cancel(r, u)) =$
$((\mathcal{S}(o) \setminus \lceil\lceil \mathsf{C}(r, u) \rceil\rceil) \cup \{cancelled(r, u)\})^\odot$

B9–B18 accurately capture NOVEL CREATION, COMPLETE ERASURE, and ACCOMMODATION.

Three-Party Operations. Clearly, any implementation of DELEGATE and ASSIGN must involve at least two messages. Figure 5(A) exemplifies the message pattern for delegation. Bookie (the delegator) delegates c_B to Charlie (the delegatee). Bookie sends $Delegate(c_B, Charlie)$ to Charlie. Let $d\_c_B = \mathsf{C}(Charlie, Alice, \$12, BeatingtheOdds)$. Upon its receipt, Charlie sends $Create(d\_c_B)$ to Alice, thus fully realizing the delegation. Figure 5(B) exemplifies the message pattern for assignment. Here, Alice (the assignor) wants to assign c_B from Bookie to Bob (the assignee). Alice sends $Assign(c_B, Bob)$ to Bookie. Let $a\_c_B = \mathsf{C}(Bookie, Bob, \$12, BeatingtheOdds)$. Upon its receipt, Bookie sends $Create(a\_c_B)$ to Bob, thus fully realizing the assignment.

B19 and B20 state the semantics of $Delegate$ and $Assign$ messages, respectively: observing either of these messages has no direct effect on the agent.

B19. $\mathcal{S}(o; Delegate(x, y, z, r, u)) = \mathcal{S}(o)$

B20. $\mathcal{S}(o; Assign(x, y, z, r, u)) = \mathcal{S}(o)$

The computation of $\mathcal{S}(o)$ is closed under B14–B20.

In the delegate and assign patterns, the initiating messages—$Delegate$ and $Assign$, respectively—are *instructions* to an agent to create a new commitment. R1 and R2 in Table 1 capture the instructional nature of the delegate and assign messages, respectively, as integrity constraints. Each row in Table 1 is in fact, an integrity constraint on agent behavior, and is of the form $\lfloor Trigger[Before : After]Effect \rfloor_{Agent}$. For example, R1 is $\lfloor Delegate(x, y, z, r, u)[\varepsilon : \varepsilon] Create(z, y, r, u) \rfloor_z$. R3–R8 are explained below.

There are a few points of note about delegation and assignment as presented here. One, R1 and R2 have nothing to do with restoring alignment. That the $Create$ must follow the instruction simply alludes to the atomicity of delegation and assignment as operations.

Two, delegation does not involve a notification from the delegator to the creditor that the commitment is being delegated. No doubt, such notifications could be practically valuable; however, our aim here is to delineate the core patterns on top of which additional patterns, such as those involving a notification to the creditor may be built. For the same reason, assignment does not involve a notification from the assignor to the assignee.

Three, the new commitment must be explicitly created by the debtor—the delegatee in the case of delegation and the debtor in the case of assignment. This reflects upon a principled approach for manipulating commitments, by reusing the semantics of $Create$.

Four, if Bookie delegates c_B twice to Charlie, then the second time Charlie need not send a $Create$: such a message would be useless under NOVEL CREATION. This paper sacrifices optimization in favor of simplicity.

Considerations of when a commitment operation may successfully occur are beyond our scope (for delegation, [9] offers an interesting discussion). This papers assumes that all operations are successful. Hence, even though Figure 5(A) shows c_B to hold before delegation is initiated, that should not be interpreted as a success precondition for delegation. Even if Bookie did not infer c_B initially, Bookie's delegate message to Charlie would still cause Charlie to send the create message to Alice.

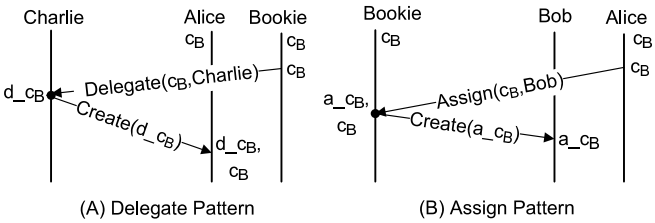

Figure 5: The delegate and assign patterns

5.1 Notifications

Recall that NOTIFICATION states that creditors must notify debtors of detaches, and debtors must notify creditors of discharges. Two cases arise for each kind.

#	Name	Agent	Trigger	Before	After	Effect
R1	Delegate	z	$Delegate(x, y, z, r, u)$	ε	ε	$Create(z, y, r, u)$
R2	Assign	x	$Assign(x, y, z, r, u)$	ε	ε	$Create(x, z, r, u)$
R3	Detach1	y	$Inform(z, y, s)$	$\mathsf{C}(x, y, r \wedge s', u) \wedge \neg\mathsf{C}(x, y, r, u) \wedge \neg s'$ where $s \vdash s'$	ε	$Inform(y, x, s')$
R4	Detach2	y	$Create(x, y, s, u)$	$\neg\mathsf{C}(x, y, r \wedge s', u) \wedge s'$	$\mathsf{C}(x, y, r \wedge s', u)$	$Inform(y, x, s')$
R5	Discharge1	x	$Inform(z, x, u)$	$\mathsf{C}(x, y, r, u') \wedge \neg u'$ where $u \vdash u'$	ε	$Inform(x, y, u')$
R6	Discharge2	x	$Create(x, y, r, u)$	$\neg\mathsf{C}(x, y, r, u) \wedge u'$ where $u \vdash u'$	ε	$Inform(x, y, u')$
R7	D-Priority	x	$Inform(z, x, s)$	$cancelled(x, y, r \wedge s', u) \wedge \neg\mathsf{C}(x, y, r \wedge s', u) \wedge \neg s'$ where $s \vdash s'$	ε	$Create(x, y, r, u)$
R8	C-Priority	y	$Cancel(x, y, r \wedge s, u)$	$s \wedge \mathsf{C}(x, y, r \wedge s, u) \wedge \neg\mathsf{C}(x, y, r', u')$ such that $\mathsf{C}(x, y, r', u') \succ \mathsf{C}(x, y, r, u)$	ε	$Release(x, y, r, u)$

Table 1: Integrity constraints on agent behavior

Detach1 (R3). y infers $\mathsf{C}(x, y, r \wedge s', u)$ and $\neg\mathsf{C}(x, y, r, u) \wedge \neg s'$, meaning that the commitment is not detached yet. y then observes $Inform(s)$ from some z such that $s \vdash s'$. As a result, s' holds and $\mathsf{C}(x, y, r \wedge s', u)$ is detached, and y infers $\mathsf{C}(x, y, r, u)$. y must now inform x about the detach by sending $Inform(y, x, s')$.

Detach2 (R4). y infers s' and $\neg\mathsf{C}(x, y, r \wedge s', u)$, and then observes $Create(x, y, s, u)$ such that $\mathsf{C}(x, y, r \wedge s', u)$ holds. $\mathsf{C}(x, y, r \wedge s', u)$ is detached upon s'; hence, y infers $\mathsf{C}(x, y, r, u)$. y must now inform x about the detach by sending $Inform(y, x, s')$.

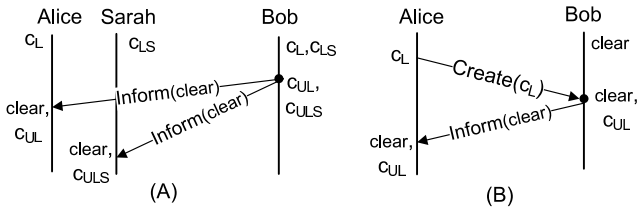

Figure 6: Detach notifications

Figure 3(B) illustrates R3. When Bob receives $Inform(clear)$, R3 kicks in and ensures Alice is notified, thus preserving alignment. Figure 6(A) is another example of R3 at work. Here Alice and Sarah are committed to meeting Bob at the lake if the sky is clear (c_L and c_{LS}, respectively). At some point, Bob figures the sky is clear and therefore infers that both Alice and Sarah are now unconditionally committed to meet him (c_{UL} and c_{ULS}, respectively). R3 ensures that both Alice and Sarah are notified that the clear condition has been met, thus preserving alignment. Figures 6(B) illustrates R4. Here, Bob already infers $clear$. So when Bob receives $Create(c_L)$, Bob infers that Alice is unconditionally committed (c_{UL}). R4 kicks in and ensures Alice is notified.

Discharge1 (R5). x infers $\mathsf{C}(x, y, r, u')$ and $\neg u'$. x then observes $Inform(u)$ from some z such that $u \vdash u'$. As a result, u' holds and $\mathsf{C}(x, y, r, u')$ is discharged. x must now inform the creditor y of the discharge by sending $Inform(x, y, u')$.

Discharge2 (R6). x infers u'. x then sends $Create(x, y, r, u)$ such that $u \vdash u'$. x will not infer $\mathsf{C}(x, y, r, u')$ because u' holds. However, y may not yet infer u'. Therefore, y may infer $\mathsf{C}(x, y, r, u')$. Hence, x must now send $Inform(x, y, u')$.

Figure 7 illustrates the usage of R5. Alice is committed to both Bob and Sarah to be at the lake (c_L and s_L, respectively). When

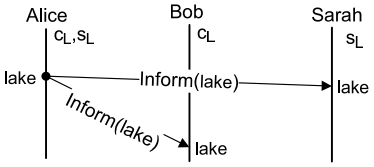

Figure 7: Discharge notification

Alice gets to the lake, she discharges those commitments. R5 kicks in and ensures that both Bob and Sarah are informed accordingly so that they also consider their respective commitments discharged.

In Figure 5(A), after Alice observes the create message from Charlie, suppose Alice sends Bookie $Inform(\$12)$ (if she already inferred \$12, then upon observing the create, R4 would apply). This detaches c_B. Then R3 kicks in and ensures that Alice also sends Charlie $Inform(\$12)$. This should not be taken to mean that Alice sends \$12 each to Bookie and Charlie—the proposition \$12 is semantically no different than the proposition $clear$. An analogous argument can be made for the scenario in Figure 5(B). Suppose that after Bookie sends the create message, it sends $Inform(BeatingtheOdds)$ to Alice. Now R5 would ensure that Bookie also sent $Inform(BeatingtheOdds)$ to Bob.

5.2 Priority

Below, we formalize the implications of detach priority and cancel priority for a commitment $\mathsf{C}(x, y, r \wedge s, u)$.

Detach Priority (R7). x infers $cancelled(x, y, r \wedge s', u)$ and $\neg\mathsf{C}(x, y, r \wedge s', u) \wedge \neg s'$. (Note that $cancelled(x, y, r \wedge s', u) \not\Rightarrow \neg\mathsf{C}(x, y, r \wedge s', u)$. A cancelled commitment may come to hold again because a stronger commitment was created.) x then observes $Inform(s)$ from some z such that $s \vdash s'$. If $\mathsf{C}(x, y, r \wedge s', u)$ had not been cancelled, it would have been detached. But y may not know about the cancellation yet. Therefore, the debtor must act as if the commitment has been detached. Hence, it must now send $Create(x, y, r, u)$.

Cancel Priority (R8). y infers s and $\mathsf{C}(x, y, r \wedge s, u)$. Therefore, it also infers $\mathsf{C}(x, y, r, u)$. y then observes $Cancel(x, y, r \wedge s, u)$. It could be that x sent $Cancel(x, y, r \wedge s, u)$ without knowing that s holds, and therefore x may not infer $\mathsf{C}(x, y, r, u)$. To fix this possible misalignment, y must now send $Release(x, y, r, u)$. y though does not have to send the release if a commitment strictly stronger than $\mathsf{C}(x, y, r, u)$ holds. Sending the release then will be ineffective because of COMPLETE ERASURE.

Figure 4(B) illustrates the case of detach priority to fix the misalignment of Figure 4(A), whereas Figure 4(C) illustrates the case

of cancel priority.

It could be that in the case of detach priority, Alice cheats by sending the payment even after receiving the cancel. Analogously, in the case of cancel priority, Bookie could cheat and get away with it. This paper does not address the issue of trust; it is orthogonal to the problem of alignment.

R1–R8 are weak and locally executable constraints on an agent's behavior because they only call for an agent to *send* messages. They involve neither receiving a message nor synchronizing with another agent.

5.3 Putting It All Together

Now it remains to show that under the assumptions we have made, the formalization of commitment operations we have proposed guarantees that any multiagent system is aligned. Notice that a commitment is strengthened only through a $Create$ or an $Inform$ (as detach). A commitment is removed or weakened only through a $Release$ or $Cancel$, or an $Inform$ (as discharge).

THEOREM 1. *For any $\mathcal{A}$, A1–A4, B1–B20 and R1–R8 guarantee alignment, that is, $[\![\mathcal{A}]\!]$.*

PROOF. (Sketch) $\mathcal{A}$ is aligned at the outset, i.e., in the observation vector of empty sequences, when no agent has made any observations. Inductively, assume that $\mathcal{A}$ is aligned up to a quiescent, integral observation vector O. Consider two agents, x and y in $\mathcal{A}$.

Now expand O to a quiescent, integral observation vector $O' = O; O^{\Delta}$. There are two possible threats to alignment: (1) if y infers a new commitment as creditor that its debtor doesn't; and (2) if y continues to infer a commitment as creditor that it previously inferred, but its debtor no longer does.

For (1), consider a commitment added by y, i.e., $\mathsf{C}(x, y, r, u) \in \mathcal{S}(O'_y) \setminus \mathcal{S}(O_y)$. Without loss of generality, assume $\mathsf{C}(x, y, r, u)$ is maximally strong, i.e., no other commitment added by y is strictly stronger than $\mathsf{C}(x, y, r, u)$. This means O^{Δ}_y includes receiving a detach ($Inform$) or a create message. For a detach, by integrity, y would have sent a message to x, which would have landed within O^{Δ}_x to ensure quiescence. A create would have originated from x. In either case, the quiescence of O' ensures that $O'_x \vdash \mathsf{C}(x, y, r, u)$.

For (2), consider a commitment not added by y but removed by x, i.e., $\mathsf{C}(x, y, r, u) \in \mathcal{S}(O_y)$ and $\mathsf{C}(x, y, r, u) \in \mathcal{S}(O_x) \setminus \mathcal{S}(O'_x)$. Without loss of generality, assume $\mathsf{C}(x, y, r, u)$ is maximally strong, i.e., no other commitment removed by x is strictly stronger than $\mathsf{C}(x, y, r, u)$.

Because $\mathsf{C}(x, y, r, u) \in \mathcal{S}(O_y)$, by our inductive hypothesis, $\mathsf{C}(x, y, r, u) \in \mathcal{S}(O_x)$. Hence, if $\mathsf{C}(x, y, r, u) \notin \mathcal{S}(O'_x)$, this means O^{Δ}_x includes receiving a discharge or release, or sending a cancel message. The release would be sent by y, thus $\mathsf{C}(x, y, r, u) \notin \mathcal{S}(O'_y)$. The cancel would be sent to y and the discharge would be propagated to y to ensure integrity. Therefore, by quiescence, $\mathsf{C}(x, y, r, u) \notin \mathcal{S}(O'_y)$. □

6. DISCUSSION

Our formalization of the commitment operations meets both autonomy compatibility and semanticity. It identifies the fundamental multiparty messaging patterns. Other business patterns may be built on top. For example, our delegation pattern may be thought of as *delegation while retaining responsibility* since the delegator remains committed too. A *delegation without responsibility* pattern would additionally involve a cancellation message from the delegator to the creditor. Singh *et al.* describe several such patterns from an architectural point of view [14].

Our approach can benefit areas where commitments are used as the central basis for semantics. The connection with communication languages [4, 5] and protocols [3] is the most obvious. Winikoff [15] studies how commitments may be implemented in a distributed setting. However, his solution only allows for a monotonically increasing set of commitments, and does not support discharge, release, and cancel.

Argumentation (for example, see [1]) is another major application. Players in a dialogue game are envisaged as having private commitment stores. In most current work, a dialogue protocol, which limits how and when the players may make moves, also helps to keep the agents aligned. However, it may be unduly restrictive; for example, it may only allow turn taking. Our results could lead to more flexible and robust dialogue protocols.

The problem of state alignment in distributed systems in a general one. Our approach exploits the semantics of commitments to enable a flexible and principled approach. The work on belief alignment is relevant [6, 10], although it doesn't involve the richness of commitment operations and multiparty interactions as studied here.

7. REFERENCES

[1] L. Amgoud, N. Maudet, and S. Parsons. An argumentation-based semantics for agent communication languages. In *Proc. ECAI*, 2002.

[2] A. K. Chopra and M. P. Singh. Constitutive interoperability. In *Proc. AAMAS*, 2008.

[3] N. Desai, A. K. Chopra, and M. P. Singh. Representing and reasoning about commitments in business processes. In *Proceedings of the 22nd Conference on Artificial Intelligence*, 2007.

[4] R. A. Flores, P. Pasquier, and B. Chaib-draa. Conversational semantics with social commitments. In *Agent Communication*, volume 3396 of *LNCS*. Springer, 2004.

[5] N. Fornara and M. Colombetti. A commitment-based approach to agent communication. *Applied Artificial Intelligence*, 18(9-10), 2004.

[6] M. Huhns and D. M. Bridgeland. Multiagent truth maintenance. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(6), 1991.

[7] A. D. Kshemkalyani, M. Raynal, and M. Singhal. An introduction to snapshot algorithms in distributed computing. *Distributed Systems Engineering*, 2(4), 1995.

[8] P. McBurney and S. Parsons. Posit spaces: a performative model of e-commerce. In *Proc. AAMAS*, 2003.

[9] T. J. Norman and C. Reed. Delegation and responsibility. In *Proc. Workshop on Agent Theories Architectures and Languages*, 2001.

[10] S. Paurobally, J. Cunningham, and N. R. Jennings. Ensuring consistency in the joint beliefs of interacting agents. In *Proc. AAMAS*, 2003.

[11] M. Rovatsos. Dynamic semantics for agent communication languages. In *Proc. AAMAS*, 2007.

[12] M. P. Singh. An ontology for commitments in multiagent systems: Toward a unification of normative concepts. *Artificial Intelligence and Law*, 7, 1999.

[13] M. P. Singh. Semantical considerations on dialectical and practical commitments. In *Proc. AAAI*, 2008.

[14] M. P. Singh, A. K. Chopra, and N. Desai. Commitment-based SOA. *IEEE Computer*, 42, 2009. Accepted.

[15] M. Winikoff. Implementing commitment-based interactions. In *Proc. AAMAS*, 2007.