# Generalized Commitment Alignment

Amit K. Chopra
Lancaster University
Lancaster LA1 4WA, United Kingdom
a.chopra1@lancaster.ac.uk

Munindar P. Singh
North Carolina State University
Raleigh, NC 27695-8206, USA
singh@ncsu.edu

## ABSTRACT

The interoperability of interacting components means that their expectations of each other remain in agreement. A commitment captures what one agent (its creditor) may expect from another agent (its debtor). Chopra and Singh (C&S) motivate commitment alignment as a meaning-based form of interoperation and show how to ensure alignment among agents despite asynchrony.

Although C&S's approach demonstrates the key strengths of relying on commitment semantics, it suffers from key shortcomings, which limit its applicability in practice. One, C&S do not model commitments properly, causing unacceptable interference between commitments in different transactions. Two, they require that the communication infrastructure guarantee first-in first-out (FIFO) delivery of messages for every agent-agent channel. Three, C&S guarantee alignment only in quiescent states (where no messages are in transit); however, such states may never obtain in enactments of real systems.

Our approach retains and enhances C&S's key strengths and avoids their shortcomings by providing a declarative semantics-based generalized treatment of alignment. Specifically, we (1) motivate a declarative notion of alignment-relevant system states termed completeness; (2) prove that it coincides with alignment; and (3) provide the computations by which a system of agents provably progresses toward alignment assuming eventual delivery of messages.

## Categories and Subject Descriptors

H.1.0 [**Information Systems**]: Models and Principles—*General*; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*

## General Terms

Theory

## Keywords

Interoperability; Commitments; Protocols; Transactions

**Appears in:** *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May, 4–8, 2015, Istanbul, Turkey.*

## 1. INTRODUCTION

This paper concerns decentralized multiagent systems in which autonomous agents interact with each other. Normally, each agent in such a system would represent an autonomous social principal, such as a human or an organization. Specification of interaction protocols and infrastructures for the distributed enactment of such protocols are major themes of research in multiagent systems.

An important class of approaches for specifying interaction protocols is that of commitment protocols [12, 25, 26]. The formula $\mathsf{C}(x, y, r, u)$ represents a commitment, meaning that $x$ (the debtor) is committed to $y$ (the creditor) that if $r$ (the antecedent) holds, then $u$ (the consequent) will hold [22]. A commitment protocol specifies the meanings of the messages in terms of their effects on commitments. For example, in a protocol for e-business, one could specify that the message *Offer* from a merchant to a customer means that the merchant is committed to the customer for delivering the goods offered if the customer pays the price quoted. A growing body of work has addressed various aspects of commitments and commitment protocols [2, 4, 8, 9, 13].

We address the challenges of enacting commitment protocols in decentralized settings where the agents communicate asynchronously and no central party maintains the definitive global state. Instead each agent maintains its own state based on the messages it has observed, both sent and received. Generally, in a distributed system, agents' observations may differ: agents may observe different sets of messages, and even when they observe the same messages, they may do so in different orders. Coupled with the fact that messages affect commitments, this means that the agents may infer different active commitments at any given moment. Since commitments reflect the expectations one agent has of another, such a difference in inferred commitments is an indication of a failure of interoperability at the semantic level. Preventing such interoperability failures would be important in business and security settings.

Chopra and Singh [6] (C&S, for short) define commitment misalignment as the following situation: an agent $y$ infers it is the creditor of a commitment whose debtor is $x$, but $x$ does not infer that it is the debtor of that commitment to $y$. For example, upon observing the *Offer* message Bob (a customer) may expect that if he pays the quoted price, then Alice (the merchant) will deliver the offered goods. If Alice does not also infer the same commitment, the two would be misaligned. And that would indicate a potential breakdown in interoperability. Commitment *alignment* is a novel and useful concept: it characterizes interoperability in terms of the meanings of interactions not merely in terms of message transmission as traditionally understood, e.g., [1].

C&S formalize alignment and provide a method for ensuring alignment among agents. Although we share C&S's intuitions about commitment alignment, we find that their formalization and method embody limiting assumptions that preclude practical application. Accordingly, we contribute a generalized approach that tackles the following situations.

**Independent transactions.** Informally, a transaction logically encapsulates a set of correlated commitments. Reasoning about commitments in one transaction ideally should not interfere with reasoning about commitments in another. Suppose a customer is involved in two transactions with a merchant, each involving a commitment to deliver some book for some payment. Then canceling one should not affect the other. C&S's approach does not support multiple transactions and generates erroneous interference between commitments of the same debtor-creditor pair.

We contribute a language that supports modeling commitments as belonging to different transactions. We formalize alignment for commitments in this language and present a method that guarantees alignment.

**Message delivery order.** C&S require ordered communication, i.e., each channel is first-in first-out (FIFO). FIFO delivery is not acceptable in every setting because it requires buffering messages to ensure in-order delivery. Specifically, it hinders applications that require extremely low latency (e.g., in finance) and that are storage-constrained (e.g., in the Internet of Things).

We contribute an approach that dispenses with FIFO channels, which additionally leads to looser coupling between agents with potential gains in performance.

**System states under consideration.** C&S define alignment to apply only in states that are *quiescent*, meaning those states where no message is in transit. Doing so eliminates false negatives. For instance, we should not expect alignment when a debtor of a commitment has sent a message to the creditor canceling the commitment, but which the creditor has not yet received. However, C&S's approach additionally eliminates true positives. Specifically, these are cases where we should determine that the agents are aligned even though some messages are in transit.

We contribute a notion of complete states that captures better where we expect alignment to hold. We show that under our method, completeness with respect to commitment implies alignment with respect to the commitment, and vice versa.

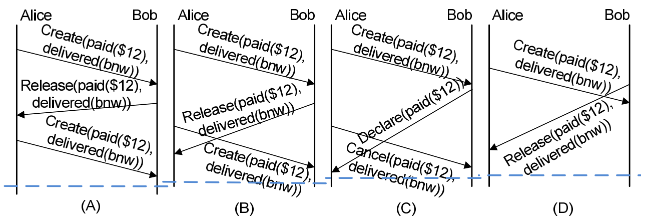
**Complexity.** C&S's approach mixes in procedural elements to describe the notifications required from agents and to model priority between debtors and creditors.

We contribute a significantly simplified, purely declarative approach that requires fewer notifications and avoids the notion of priority.

*Organization.* Section 2 motivates the challenge of alignment and provides a technical overview of C&S's approach. Section 3 describes the opportunities for improving over C&S. Section 4 describes the details of our approach. Section 5 discusses related work and directions for future work.

## 2. BACKGROUND ON ALIGNMENT

Fig. 1 reproduces C&S's scenarios to motivate commitment alignment. Each subfigure represents an enactment in the manner of a sequence diagram. Here, a vertical line represents an agent's history; an arrow represents a message exchange; time flows downward. Table 1 lists the messages that concern us in this paper.



Alice
Bob
Create(paid($12), delivered(bnw))
Release(paid($12), delivered(bnw))
Create(paid($12), delivered(bnw))
(A)
Alice
Bob
Create(paid($12), delivered(bnw))
Release(paid($12), delivered(bnw))
Create(paid($12), delivered(bnw))
(B)
Alice
Bob
Create(paid($12), delivered(bnw))
Declare(paid($12))
Cancel(paid($12), delivered(bnw))
(C)
Alice
Bob
Create(paid($12), delivered(bnw))
Release(paid($12), delivered(bnw))
(D)

Figure 1: Alice and Bob are aligned in A, not in B, C, D.

Table 1: Messages and their effects; $x$ and $y$ are agents; $r$ and $u$ are propositions.

| Message | Sender | Receiver | Naïve effect |
|---|---|---|---|
| $\mathsf{Create}(x, y, r, u)$ | $x$ | $y$ | $\mathsf{C}(x, y, r, u)$ |
| $\mathsf{Cancel}(x, y, r, u)$ | $x$ | $y$ | $\neg\mathsf{C}(x, y, r, u)$ |
| $\mathsf{Release}(x, y, r, u)$ | $y$ | $x$ | $\neg\mathsf{C}(x, y, r, u)$ |
| $\mathsf{Declare}(x, y, p)$ | $x$ | $y$ | $p$ |

Table 1 lists the messages corresponding to commitment operations [21] and their naïve effects. We see below that this naïve formulation is inapplicable in distributed settings. Table 1 introduces Declare, by which a suitably empowered agent [14] brings about relevant conditions that feature in the antecedents or consequents of commitments. For example, declaring a payment or delivery would cause the detach or discharge of some commitments in Table 2. A received Declare may be forwarded by an agent to others. Since the identity of the declarer is not relevant for this paper, we use Declare for both purposes.

Table 2 lists the example commitments we use as running examples, especially to illustrate C&S's approach.

Table 2: Example commitments. When the antecedent is $\top$, the commitment is unconditional. Here, bnw, gow, and soc are names of books.

| | |
|---|---|
| $c_A$ | $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \mathsf{paid}(\$12), \mathsf{delivered}(\mathsf{bnw}))$ |
| $c_{UA}$ | $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \top, \mathsf{delivered}(\mathsf{bnw}))$ |
| $c_B$ | $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \mathsf{paid}(\$12), \mathsf{delivered}(\mathsf{gow}))$ |
| $c_{UB}$ | $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \top, \mathsf{delivered}(\mathsf{gow}))$ |
| $c_{AB}$ | $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \mathsf{paid}(\$12), \mathsf{delivered}(\mathsf{bnw}) \wedge \mathsf{delivered}(\mathsf{gow}))$ |
| $c_{UAB}$ | $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \top, \mathsf{delivered}(\mathsf{bnw}) \wedge \mathsf{delivered}(\mathsf{gow}))$ |
| $c_C$ | $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \mathsf{paid}(\$15), \mathsf{delivered}(\mathsf{soc}))$ |
| $c_{UC}$ | $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \top, \mathsf{delivered}(\mathsf{soc}))$ |
| $c_{UAC}$ | $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \top, \mathsf{delivered}(\mathsf{bnw}) \wedge \mathsf{delivered}(\mathsf{soc}))$ |

Following the naïve effects in Table 1, let us see how the enactments from Fig. 1 fare from the point of view of alignment at the moment denoted by the dashed line.

*Example 1:* In Fig. 1(A), there is no misalignment. Both Alice and Bob infer that $c_A$.

In Fig. 1(B), Alice and Bob are misaligned. Bob infers $c_A$ but Alice does not. The reason is that Bob observes the second create after the release whereas Alice observes them in the opposite order.

In Fig. 1(C), Alice and Bob are misaligned. Bob infers $c_{UA}$ (because of the detach when he sends the declare) but Alice does not because she had already canceled $c_A$ by the time the declare arrives and so there is nothing to detach.

In Fig. 1(D), Alice and Bob are misaligned. Bob infers $c_A$ but Alice does not.

## 2.1 Technical Preliminaries

We introduce some technical details. $B_1$–$B_8$ are postulates for commitment reasoning [22].

$B_1$. DISCHARGE. $u \rightarrow \neg \mathsf{C}(r, u)$

$B_2$. DETACH. $\mathsf{C}(r \wedge s, u) \wedge r \rightarrow \mathsf{C}(s, u)$

$B_3$. AUGMENT. From $\mathsf{C}(r, u)$, $s \vdash r$, $s \nvdash u$ infer $\mathsf{C}(s, u)$

$B_4$. L-DISJOIN. $\mathsf{C}(r, u) \wedge \mathsf{C}(s, u) \rightarrow \mathsf{C}(r \vee s, u)$

$B_5$. R-CONJOIN. $\mathsf{C}(r, u) \wedge \mathsf{C}(r, v) \rightarrow \mathsf{C}(r, u \wedge v)$

$B_6$. CONSISTENCY. $\neg \mathsf{C}(r, \bot)$

$B_7$. NONVACUITY. From $r \vdash u$ infer $\neg \mathsf{C}(r, u)$

$B_8$. WEAKEN. $\mathsf{C}(r, u \wedge v) \wedge \neg u \rightarrow \mathsf{C}(r, u)$

*Definition* 1*:* A commitment $\mathsf{C}(r, u)$ is *stronger than* $\mathsf{C}(s, v)$ (with the same debtor and creditor), or $\mathsf{C}(r, u) \succeq \mathsf{C}(s, v)$, if and only if $u \vdash v$ and $s \vdash r$. If $\mathsf{C}(r, u) \succeq \mathsf{C}(s, v)$ but $\mathsf{C}(s, v) \not\succeq \mathsf{C}(r, u)$, $\mathsf{C}(r, u)$ is *strictly stronger than* $\mathsf{C}(s, v)$ or $\mathsf{C}(r, u) \succ \mathsf{C}(s, v)$.

For example, from Table 2, $c_{UAB} \succ c_{AB} \succ c_A$ and $c_{UAB} \succ c_{UA} \succ c_A$ but $c_B \not\succeq c_A$ and $c_A \not\succeq c_B$. Let's say an agent infers $c_A$ and $c_B$. Then, by $B_5$, the agent infers $c_{AB}$ as well, which is strictly stronger than both $c_A$ and $c_B$.

## 2.2 C&S's Solution for Ensuring Alignment

In C&S's solution, agents can send any message at any time. The effects of the messages, however, are not computed according to Table 1 but by rules C&S provide. For instance, Bob's observation of the second create in Fig. 1(B) is a *noop* because it is not strictly stronger than commitments that held previously (*Principle of Novel Creation*). This solves the misalignment in Fig. 1(B). C&S address Fig. 1(D) by treating a released or canceled commitment as having held previously (*Principle of Accommodation*), thus making Bob's observation of the create a noop.

C&S characterize the problem in Fig. 1(C) as arising from a race between the debtor's (Alice's) cancelation and the creditor's (Bob's) detach. To address this misalignment, they assign priorities to actions as part of the protocol specification, and prescribe actions that agents must take depending on who has priority (*Principle of Priority*). In this example, if Alice has priority (for $c_A$'s cancellation), then Bob must release Alice from $c_{UA}$ upon observing the cancel. Alternatively, if Bob has priority, then Alice must explicitly create $c_{UA}$. In either case, the misalignment is avoided.

C&S introduce two additional principles, which we mention for completeness. The *Principle of Complete Erasure* states that canceling or releasing a commitment removes all weaker commitments if no strictly stronger commitment holds; otherwise, it is a noop. The *Principle of Notification* states that debtors must be notified by creditors of detaches and creditors must be notified by debtors of discharges.

C&S show that a multiagent system is aligned in states where all notifications have been sent ("integral"), and no messages are in transit ("quiescent"): if a creditor infers a commitment, then the debtor also infers it.

## 3. GENERALIZATION OPPORTUNITIES

We now motivate opportunities for improvements by discussing the limitations of C&S's approach.

## 3.1 Independent Transactions

Conceptually, the idea behind a transaction is that it encapsulates a set of related events or operations. Specifically, events in independent transactions will not affect each other. Thus, events concerning a commitment in one transaction should not affect a commitment in another transaction. For example, as illustrated in Section 1, canceling a commitment for a book in one transaction should not affect a commitment for a book in another transaction. A key limitation of C&S's approach is that it does not support multiple transactions. In effect, it treats all commitments as being in the same *implicit* transaction. Example 2 illustrates this.

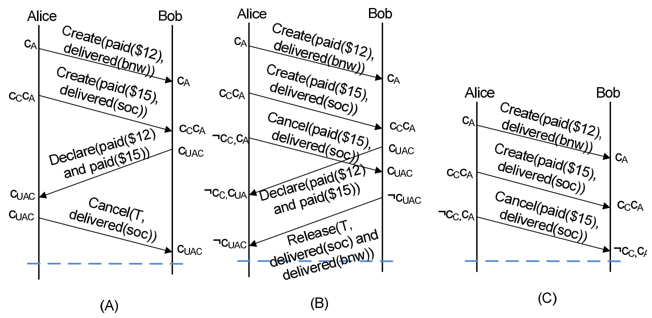

Alice
Bob
Create(paid($12), delivered(bnw))
Create(paid($15), delivered(soc))
Declare(paid($12) and paid($15))
Cancel(T, delivered(soc))
Cancel(paid($15), delivered(soc))
Release(T, delivered(soc) and delivered(bnw))
$c_A$
$c_Cc_A$
$c_{UAC}$
$\neg c_C, c_A$
$\neg c_C, c_{UA}$
$\neg c_{UAC}$
(A)
(B)
(C)

Figure 2: Aligned but unrealistic outcomes due to the lack of transactions in C&S.

*Example* 2*:* See Fig. 2(A). Alice creates $c_A$ and $c_C$. Let's assume that the two commitments are in separate transactions. Accordingly, we expect no interference between them. Bob sends payment for both books (via a Declare message). Hence, Bob infers the unconditional versions of both commitments, $c_{UA}$ and $c_{UC}$. Due to $B_5$, however, Bob infers $c_{UAC}$, which is strictly stronger than $c_{UA}$ and $c_{UC}$. Therefore, Alice's cancelation of $c_{UC}$ is ineffective (by C&S's *Principle of Complete Erasure*).

Interestingly, C&S advocate using domain identifiers in the content of the commitment to enable making the "same" commitment twice (Ex. 5 on p. 940). In the spirit of their example, Alice could have created $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \mathsf{paid}(\mathsf{id0}, \$12)$, $\mathsf{delivered}(\mathsf{id0}, \mathsf{bnw}))$ and $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \mathsf{paid}(\mathsf{id1}, \$15)$, $\mathsf{delivered}(\mathsf{id1}, \mathsf{soc}))$ instead of $c_A$ and $c_C$, respectively; here, id0 and id1 serve to distinguish the transactions. However, notice that from their unconditional versions, one would still infer $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \top, \mathsf{delivered}(\mathsf{id0}, \mathsf{bnw}) \wedge \mathsf{delivered}(\mathsf{id1}, \mathsf{soc}))$ (again, by $B_5$). This means that, just as in Example 2, the unconditional commitment for soc cannot be canceled by Alice. This means that the two commitments, even with domain identifiers to help distinguish them, are not in separate transactions. In Section 4, we adopt C&S's intuition about using domain identifiers but treat it more fully in order to avoid such anomalies.

Example 3 shows an enactment where the outcome following C&S's approach is especially undesirable.

*Example 3:* See Fig. 2(B). Assume $c_A$ and $c_C$ are in different transactions. Alice's cancel is treated as a noop by Bob because Bob already infers the stronger $c_{UAC}$. Alice, in contrast, infers $c_{UA}$ but not $c_{UC}$ upon seeing Bob's declare because she had already canceled $c_C$. Therefore, Alice does not infer $c_{UAC}$. A potential misalignment ensues: Bob infers $c_{UAC}$, which Alice does not. C&S's *Principle of Priority* kicks in to handle this. Let's say Alice has priority. Accordingly, Bob releases Alice from $c_{UAC}$, thus fixing the misalignment. Notice that a release of just $c_{UC}$ would not have worked—it would in fact be a noop because $c_{UAC}$ holds (*Principle of Complete Erasure*). But releasing $c_{UAC}$ gets rid of $c_{UA}$ as well (by *Principle of Complete Erasure*). In effect, fixing a misalignment for one commitment $c_C$ gets rid of an unrelated commitment in a different transaction, which is highly undesirable.

To bring out the magnitude of the problem highlighted in Example 3, imagine that instead of just $c_{UA}$ there were a million unconditional commitments from Alice to Bob, each for the delivery of some item, that held for Bob at the point where Bob receives Alice's cancelation of $c_C$. Then, Bob would need to release Alice from all those—practically speaking, wipe the slate clean—to produce alignment.

Let us consider how we can avoid priority. Consider Example 4, which C&S support.

*Example 4:* See Fig. 2(C). Alice creates $c_A$ and $c_C$. She then cancels $c_C$, which makes $c_C$ go away. Only $c_A$ remains.

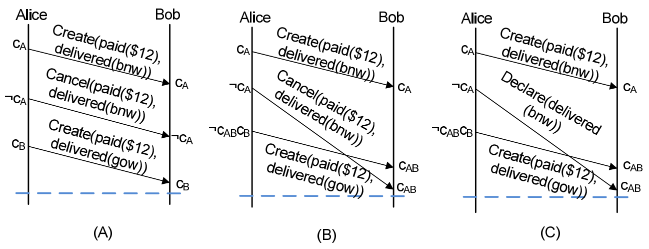
At first glance, it appears reasonable to remove commitments piecemeal, as in Example 4. But what if Bob had detached the commitments concurrently with Alice's cancelation, as in Example 3?

In general, a conditional commitment may be detached at any time and thus its cancelation would inherently lead to an undefined state. C&S's notion of priority seeks to repair this undefined state. Instead we should prevent it. In this spirit, a simpler alternative is to treat cancelations and releases not at the level of commitments but at the level of the transaction. In other words, a cancelation wipes out all the commitments in a transaction. Notice that unlike the wipeout in Example 3, the scope of the wipeout is limited by the transaction. Applying this idea to Example 4, if $c_A$ and $c_C$ were part of the same transaction, canceling the transaction would remove them both; if they belonged to different transactions, then canceling the transaction of $c_C$ would remove only $c_C$. Analogous reasoning applies to Example 3.

Once a transaction is removed by the observation of either a cancel or release, no commitment belonging to the transaction can hold again—the transaction is *used up.* This idea is especially important in non-FIFO settings, where messages can be delivered in any order (as we shall see below). In case a debtor changes it mind after sending the cancel and wants to "recreate" the commitments in a canceled transaction, it must do so in a new transaction.

## 3.2 Message Delivery Order

Fig. 3(A) shows a first-in first-out (FIFO) enactment, in which messages between any two agents are delivered in order. Fig. 3(B) shows a non-FIFO variant of the enactment. Bob sees the creates of $c_A$ and $c_B$ before the cancel of $c_A$. Therefore, Bob already infers $c_A$ (by $B_5$) when the cancel arrives. Since the cancel is for a weaker commitment than what already holds ($c_{AB} \succ c_A$), the cancel is a noop. Hence Alice and Bob are misaligned: Bob infers $c_{AB}$ at the dashed line but Alice does not. An analogous situation obtains when Alice discharges $c_A$, as in Fig. 3(C).



Alice
Bob
Create(paid($12), delivered(bnw))
Cancel(paid($12), delivered(bnw))
Create(paid($12), delivered(gow))
Declare(delivered (bnw))
$c_A$
$\neg c_A$
$c_B$
$\neg c_{AB}$
$c_{AB}$
(A)
(B)
(C)

Figure 3: In (A), messages are delivered on FIFO basis; Alice and Bob are aligned. In (B) and (C) messages are not delivered on FIFO basis, which causes misalignment.

When we deal with commitments in transactional scope, a general solution is to treat the observation of a cancel as removing all the commitments in that transaction (taking care of the problem in Fig. 3(B)), and to prohibit the sending of creates after some commitment in the transaction has been discharged (taking care of the problem in Fig. 3(C)).

C&S require FIFO message delivery. Although FIFO is supported in enterprise message queues, it limits applicability in settings where buffering is not appropriate, such as low-latency finance and sensing applications and settings where the infrastructure has reduced capabilities for storage. Our approach avoids assuming FIFO but without limiting the range of enactments that the agents can realize.

## 3.3 Systems States

C&S's notion of alignment exploits the inherent asymmetry of a commitment. Alignment requires only that a debtor meet a creditor's expectations, not that a creditor must have some expectation: introducing unnecessary expectations would only couple the parties unnecessarily.

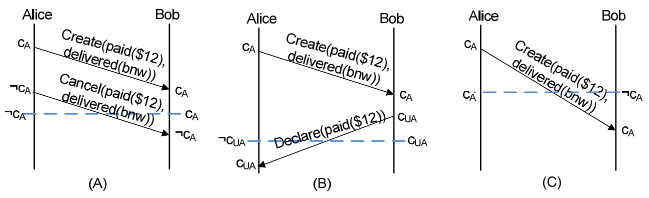


Figure 4: The dotted lines indicate nonquiescent system states. In (A) and (B), the agents are misaligned. However, in (C) the agents are aligned.

Asynchronous message transmission is causally asymmetric: the sender is aware of a message before the receiver is. Therefore, agents can be in disagreement in states where the relevant information has not propagated to the concerned agents. For example, when a debtor has sent a cancel message for a commitment but the creditor has not seen it yet, the creditor will infer the commitment but the debtor will not. Fig. 4(A) shows this state using a dashed line. In Fig. 4(B), the dashed line indicates a system state where the detach of a commitment has not propagated to the debtor. C&S formalize the idea that only *quiescent* states, where no message is in transit, are relevant for alignment. That is, the states denoted by the dashed lines in Fig. 4 are irrelevant.

However, the above formulation is too strong. In Fig. 4(C), even though the state indicated by the dashed line is not quiescent, the agents are aligned. Two commitments in different transactions would not affect each other. So even if a cancel for one is in transit, the agents may be aligned with respect to the other. In other words, we should able to claim alignment for each commitment independently.

Further, consider that in practice quiescent states may never obtain in a distributed system. Hence, to define alignment in quiescent states severely limits its practical value. Ideally, we would like to semantically characterize relevant states with respect to a commitment and show alignment in those states with respect to that commitment. Further, to show that our characterization is not too strong, we would like to show that if the agents are in agreement with respect to the commitment in some state, then the state is relevant in our characterization.

## 4. TECHNICAL FRAMEWORK

We now describe our framework in terms of a language and models of computation.

Let $\mathcal{P}$ be a finite set of predicates. Each predicate has a signature consisting of a list of domains. The first domain is the set of natural numbers ($\mathbb{N}$) and is treated as the ID of the predicate. Each domain is a set of values, written as constants in the syntax.

An *atom* is obtained from a predicate by applying the predicate on a list of constants, each constant representing a value drawn from the corresponding domain (the particular constants chosen would depend on an agent's internal reasoning, which is outside the scope of this paper). Let $\iota$ be the value of ID in an atom. Then we say that the atom belongs to the transaction whose identifier is $\iota$. More simply, we will say the atom belongs to transaction $\iota$. We require that any expression that involves two or more atoms, e.g., a DNF or a commitment expression, involves atoms with the same value for ID. Let $\iota$ be the value of ID in a commitment expression. Then we say that the commitment (expression) belongs to transaction $\iota$. Any commitment belongs to only one transaction but one transaction may include multiple commitments. Below, we use $\iota$ as the generic transaction identifier in our formal definitions. For each transaction $\iota$, we define a special atom $\top(\iota)$ that represents $\mathsf{true}$ in that transaction.

Table 3 introduces our syntax. Here $\mathcal{X}$ is a set of agents. As remarked above, we restrict our attention to expressions (Comm in Table 3) that involve atoms with the same ID. This is a reasonable restriction because otherwise there would be no way to correlate the events that discharge a commitment with those that detach it. Specifically, the commitment $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \mathsf{paid}(20, \$12), \mathsf{delivered}(15, \mathsf{bnw}))$ is malformed because it mixes transactions with IDs 20 and 15, respectively. But $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \mathsf{paid}(15, \$12), \mathsf{delivered}(15, \mathsf{bnw}))$ meets our restriction. $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \top(15), \mathsf{delivered}(15, \mathsf{bnw}))$ represents the unconditional variant of the commitment; it too meets our restriction. Unconditional commitments of the sort allowed in C&S, e.g., $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \top, \mathsf{delivered}(15, \mathsf{bnw}))$, are not included in our language.

The fact that the antecedent in an unconditional commitment is qualified with a transaction identifier is key to blocking reasoning across transactions. For example, from $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \top(15), \mathsf{delivered}(15, \mathsf{bnw}))$ and $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \top(16), \mathsf{delivered}(16, \mathsf{gow}))$, you cannot derive a stronger commitment by applying B$_5$, as you could with $c_{UA}$ and $c_{UB}$ to derive $c_{UAB}$. From $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \top(15), \mathsf{delivered}(15, \mathsf{bnw}))$ and $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \top(15), \mathsf{delivered}(15, \mathsf{gow}))$, however, you get $\mathsf{C}(\mathsf{Alice}, \mathsf{Bob}, \top(15), \mathsf{delivered}(15, \mathsf{bnw}) \wedge \mathsf{delivered}(15, \mathsf{gow}))$.

Table 3: Formal syntax.

| | | |
|---|---|---|
| Base | $\longrightarrow$ | Comm \| Atom \| $\mathsf{removed}(\mathcal{X}, \mathcal{X}, \mathbb{N})$ |
| Comm | $\longrightarrow$ | $\mathsf{C}(\mathcal{X}, \mathcal{X}, \text{DNF}, \text{CNF})$ |
| DNF | $\longrightarrow$ | And \| And $\vee$ DNF |
| CNF | $\longrightarrow$ | Or \| Or $\wedge$ CNF |
| And | $\longrightarrow$ | Atom \| Atom $\wedge$ And |
| Or | $\longrightarrow$ | Atom \| Atom $\vee$ Or |
| Message | $\longrightarrow$ | $\mathsf{Declare}(\mathcal{X}, \mathcal{X}, \text{And})$ \| $\mathsf{Cancel}(\mathcal{X}, \mathcal{X}, \mathbb{N})$ \| $\mathsf{Create}(\mathcal{X}, \mathcal{X}, \text{DNF}, \text{CNF})$ \| $\mathsf{Release}(\mathcal{X}, \mathcal{X}, \mathbb{N})$ |

Notice in Table 3, cancel and release take a transaction identifier as a parameter. And, $\mathsf{removed}$ is a stative that records that a cancel or release was observed.

Below, $x$, $y$, $z$ are agents; $p \ldots w$ are DNF or CNF expressions; $\vee$, $\wedge$, and $\neg$ are the usual connectives; $\vdash$ is the usual propositional inference symbol. To reduce clutter, we omit the debtor and creditor from expressions when they are understood, e.g., writing $\mathsf{C}(r, u)$ instead of $\mathsf{C}(x, y, r, u)$.

Let $\iota$ be the ID of all atoms that appear in $\mathsf{C}(r, u)$; that is, let $\mathsf{C}(r, u)$ belong to transaction $\iota$. Since all atoms have the same ID, if we need to refer to the ID, we need write it only once to make it explicit. Specifically, we will write $\mathsf{C}(\iota, r, u)$ when we wish to make the ID explicit.

### 4.1 Messages and Observations

Agents communicate by point-to-point messaging. Here $m(x, y)$ indicates a message $m$ from $x$ to $y$. An agent *observes* all and only those messages that it sends or receives. All facts relevant to an enactment arise from the observations an agent makes during its interactions with others. An agent's internal reasoning is irrelevant for our purposes.

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. For an observation sequence $o = \langle \ldots, m \rangle$ and message $m'$, we define the *concatenation* of $o$ with $m'$ as $\langle \ldots, m, m' \rangle$ and write it as $o; m'$.

Let $\mathcal{A}$ be a system of $k$ agents. Then, $O = [O_0 \ldots O_i \ldots O_{k-1}]$ is an *observation vector* over $\mathcal{A}$, where the $O_i$s are the observation sequences, one for each of the $k$ agents. An observation vector is thus a snapshot of the system. Below, $O$ is an observation vector and $O_x$ is agent $x$'s observation sequence within $O$. Now we define important concepts concerning observation sequences and vectors.

*Definition 2:* The length of an observation sequence is the number of observations it contains. The length of an observation vector is the sum of the lengths of its observation sequences. If $O = [O_0 \ldots O_i \ldots O_{k-1}]$ is of length $n$, then we write $O^n$ to denote its length and $O_i^n$ to denote the observation sequence of agent $i$.

*Definition 3:* Let $o'$ be an observation sequence. Then $o$ is a subsequence of $o'$, denoted by $o \sqsubseteq o'$ iff

- $o' = o$, or
- for some message $m$, $o' = o; m$, or
- for some observation sequence $o''$, $o \sqsubseteq o'' \sqsubseteq o'$.

If $o \sqsubseteq o'$ but $o \neq o'$, then we say that $o'$ strictly extends $o$ and write it as $o \sqsubset o'$.

*Definition 4:* Let $O = [O_0 \dots O_i \dots O_{k-1}]$ and $O' = [O'_0 \dots O'_i \dots O'_{k-1}]$. We say that that $O'$ extends $O$, denoted by $O \sqsubseteq O'$, iff each $O_i \sqsubseteq O'_i$. If $O \sqsubseteq O'$ but $O \neq O'$, then we say that $O'$ strictly extends $O$ and write it as $O \sqsubset O'$.

Finally, Definition 5 defines an operator that gives the remainder of an observation sequence after removing a prefix.

*Definition 5:* Let $o = \langle m_0, \dots, m_l \rangle$ and $o' = \langle m_0, \dots, m_l, m_{l+1}, \dots, m_i \rangle$. Then $o' \setminus o = \langle m_{l+1}, \dots, m_i \rangle$.

## 4.2 Agent Snapshots and Local Maintenance

We now formalize the treatment of an agent snapshot, its local state as arising from its observation sequence. In syntactic terms, a snapshot is a finite data structure. In semantic terms, a snapshot yields a set of propositions that an agent may logically infer from its observations. To enable how an agent's snapshot progresses based on its observations, we introduce two *update operators* called *addition* $\oplus$ and *subtraction* $\ominus$, which respectively "add" or "subtract" propositions from its snapshot. These operators ensure that the resulting representation is consistent according to the postulates of this paper, and that only the essential (i.e., minimal) changes are made due to each message. We now describe a way in which to implement the update operators.

Let $S$ be a set of base propositions, i.e., atoms, commitments, or $\mathsf{removed}$s. Specifically, $S$ may represent the snapshot of an agent arising from a sequence of observations. In logical terms, the state of the agent is $[\![S]\!]$, the deductive closure of its snapshot. We extend $\vdash$ to apply to snapshots: $S \vdash p$ means $p \in [\![S]\!]$. We assume that $\vdash$ respects $B_1$ to $B_8$.

Then we define the operators to modify snapshots. Below, $\beta$ is an atom or a $\mathsf{removed}(\iota)$. And, $\chi(S)$ is the set of commitments in $S$. That is, $\chi(S) = \{\mathsf{C}(r,u) : \mathsf{C}(r,u) \in S\}$. Definition 6 says that adding a $\beta$ gets rid of all commitments that are discharged by the addition, and adding a commitment is only effective if the transaction has not been removed and if the commitment has not been discharged.

*Definition 6:* We define $\oplus$ as follows.

- Given snapshot $S$ and $\beta$, $S \oplus \beta = (S \cup \{\beta\}) \setminus \{\mathsf{C}(r,u) : \mathsf{C}(r,u) \in \chi(S), S \cup \{\beta\} \not\vdash u\}$
- Given snapshot $S$ and $\beta_1, \dots, \beta_k$, $S \oplus (\beta_1 \wedge \dots \wedge \beta_k) = (S \oplus \beta_1) \dots \oplus \beta_k$
- Given snapshot $S$ and $\mathsf{C}(\iota, r, u)$, $S \oplus \mathsf{C}(\iota, r, u) = S$ if $S \vdash \mathsf{removed}(\iota) \vee u$; else $S \cup \{\mathsf{C}(\iota, r, u)\}$.

Definition 7 says that removing a transaction amounts to removing all commitments with that transaction's identifier.

*Definition 7:* Given snapshot $S$ and $\iota$, we define $\ominus$ as follows: $S \ominus \iota = S \oplus \mathsf{removed}(\iota) \setminus \{\mathsf{C}(\iota, s, v) : \mathsf{C}(\iota, s, v) \in \chi(S)\}$.

Let $S(o)$ be the snapshot corresponding to the sequence of observations $o$. Then we compute $S(o; m)$ from $S(o)$ via one or more applications of the above update operators. The specific applications of the operators depend upon the semantics of the message $m$. In $U_1$–$U_5$, $o$ is either $x$'s or $y$'s observation sequence. $S(\langle\ \rangle)$ is easily finitely stored.

$U_1$. $S(\langle\ \rangle) = \{\top[\iota]\}$, for every possible binding of $\iota$

$U_2$. $S(o; \mathsf{Create}(r,u)) = S(o) \oplus \mathsf{C}(r,u)$

$U_3$. $S(o; \mathsf{Release}(\iota)) = S(o) \ominus \iota$

$U_4$. $S(o; \mathsf{Cancel}(\iota)) = S(o) \ominus \iota$

$U_5$. $S(o; \mathsf{Declare}(\beta_0 \wedge \dots \wedge \beta_k) = S(o) \oplus \beta_0 \wedge \dots \wedge \beta_k$

Our only constraint is that once an agent infers an atom with identifier $\iota$, it no longer send any further creates for commitments with $\iota$. This reflects the intuition that once an agent begins discharging commitments in a transaction, it should not introduce new commitments to the transaction. The constraint helps address the problem in Fig. 3(C).

Lemma 1 means that if observation sequence derives a commitment of identifier $\iota$, then it cannot derive its consequent or $\mathsf{removed}(\iota)$.

*Lemma 1:* If $S(o) \vdash \mathsf{C}(\iota, r, u)$, then $S(o) \vdash \neg\mathsf{removed}(\iota) \wedge \neg u$.

## 4.3 Alignment in Relevant States

As discussed in Section 3, defining alignment as holding only in quiescent states is too strong. We need a notion that coincides with alignment. Definition 8 characterizes such a notion. It defines only those vectors to be complete with respect to a commitment where if the debtor does not infer the commitment, then the creditor does not infer it too or the creditor knows that it was either discharged or its transaction removed. Consider a vector where the creditor infers a commitment because of a detach but the debtor does not because it has not received information about the detach. Such a vector would be incomplete as well.

*Definition 8:* $O$ is *complete* with respect to $\mathsf{C}(\iota, r, u)$ if and only if $S(O_x) \vdash \neg\mathsf{C}(r,u) \Rightarrow S(O_y) \vdash \neg\mathsf{C}(r,u) \vee u \vee \mathsf{removed}(\iota)$.

Notice that if $O$ is complete with respect to $\mathsf{C}(s,v)$ and $\mathsf{C}(s,v) \succ \mathsf{C}(r,u)$, it does not imply that $O$ is complete with respect to $\mathsf{C}(r,u)$. To see this, let $O_x$ be $\langle \mathsf{Create}(x,y,r,u \wedge v), \mathsf{Declare}(z,x,u) \rangle$ and $O_y$ be $\langle \mathsf{Create}(x,y,r,u \wedge v) \rangle$; this $O$ is complete with respect to $\mathsf{C}(x,y,r,u \wedge v)$ but not complete with respect to $\mathsf{C}(x,y,r,u)$.

Definition 9 says that a vector is aligned for a commitment if and only if the following condition is met: if the creditor's observation sequence in the vector derives the commitment, then the debtor's observation sequence in the vector derives the commitment as well.

*Definition 9:* $O$ is aligned with respect to $\mathsf{C}(r,u)$ if and only if $S(O_y) \vdash \mathsf{C}(r,u)$ implies $S(O_x) \vdash \mathsf{C}(r,u)$.

Notice that Definition 9 does not rely upon any internal reasoning or state of the agent. Whether a commitment holds or not for an agent is solely a property of the agent's observations.

Theorem 1 and Theorem 2 show that alignment and completeness coincide.

*Theorem 1:* [Alignment] If $O$ is complete with respect to $\mathsf{C}(r,u)$, then $O$ is aligned with respect to $\mathsf{C}(r,u)$.

*Proof.* Let $\iota$ be $\mathsf{C}(r,u)$'s identifier. If $O$ is complete with respect to $\mathsf{C}(r,u)$, then by Definition 8, $S(O_x) \vdash \neg\mathsf{C}(r,u) \Rightarrow S(O_y) \vdash \neg\mathsf{C}(r,u) \vee u \vee \mathsf{removed}(\iota)$, that is either (i) $S(O_x) \vdash \mathsf{C}(r,u)$, or (ii) $S(O_y) \vdash \neg\mathsf{C}(r,u) \vee u \vee \mathsf{removed}(\iota)$. If (i), then $O$ is aligned with respect to $\mathsf{C}(r,u)$. If (ii), then (a) if $S(O_y) \vdash \neg\mathsf{C}(r,u)$, then $O$ is aligned with respect to $\mathsf{C}(r,u)$; (b) if $S(O_y) \vdash \vee u \vee \mathsf{removed}(\iota)$, then by the contraposition of Lemma 1, $S(O_y) \vdash \neg\mathsf{C}(r,u)$. Therefore, $O$ is aligned with respect to $\mathsf{C}(r,u)$. □

*Theorem 2:* [Completeness] If $O$ is aligned with respect to $\mathsf{C}(r,u)$, then $O$ is complete with respect to $\mathsf{C}(r,u)$.

*Proof.* We identify two cases. Case 1: $S(O_y) \vdash \mathsf{C}(r,u)$. Then, because $O$ is aligned with respect to $\mathsf{C}(r,u)$, $S(O_x) \vdash \mathsf{C}(r,u)$. Therefore, by Definition 8, $O$ is complete with respect to $\mathsf{C}(r,u)$. Case 2: $S(O_y) \vdash \neg\mathsf{C}(r,u)$. By Definition 8, $O$ is complete with respect to $\mathsf{C}(r,u)$. □

## 4.4 Progress

We proved that all vectors complete with respect to a commitment are aligned with the respect to a commitment. But what if a vector incomplete with respect to a commitment is realized by the agents' interactions? Can the agents take actions that will guarantee the realization of a vector that is complete with respect to the commitment? In other words, can we ensure that the system always progresses to completion (and, therefore, alignment) with respect to every commitment? To establish that a multiagent system will advance toward alignment, we formalize the elements of a decentralized enactment of a multiagent system.

An underlying assumption for messaging in decentralized settings is *causality*, namely, that each message is sent before being received [16]. Definition 10 captures causality in a recursive formulation of realizability, wherein an agent is always able to send a message (reflecting its autonomy in saying anything it chooses) and is able to receive a message only if the message has previously been sent.

Further, showing progress requires the assumption of *reliability*, namely, that each message that is sent arrives at its destination, i.e., every sent message is received.

*Definition* 10 (Realize)*:* A multiagent system $\mathcal{A}$ of $n$ agents $\{1 \ldots i \ldots n\}$ *realizes* an observation vector $O = [O_1 \ldots O_n]$ if and only if one of these conditions holds:

- $O = [\langle\rangle \ldots \langle\rangle]$, a vector of $n$ empty observation sequences;
- $O = [O_1 \ldots O_i; \text{send}(j,m) \ldots O_n]$ and $\mathcal{A}$ realizes $\langle O_1 \ldots O_i \ldots O_n\rangle$; or
- $O = [O_1 \ldots O_i; \text{receive}(j,m) \ldots O_n]$ and $\mathcal{A}$ realizes $[O_1 \ldots O_i \ldots O_n]$ and $O_j$ contains send$(i,m)$.

C&S give action rules that specify the messages an agent must send based upon its local observations to propagate the information necessary for alignment. Specifically, debtors must notify creditors of discharges and creditors must notify debtors of detaches. Their notifications are of four kinds. One, if a commitment holds and then is discharged (*Discharge Posterior*); two, if a commitment whose consequent already holds is created (*Discharge Prior*); three, if a commitment holds and then is detached (*Detach Posterior*); and four, if a commitment whose detach condition already holds is created (*Detach Prior*). We give declarative characterizations analogous to each notification kind, except *Discharge Prior*. *Discharge Prior* does not apply for us because we adopt the restriction to not send creates for commitments of identifier $\iota$ if some atom with identifier $\iota$ already holds.

Definition 11 characterizes the messages that a debtor should send to the creditors of commitments that are discharged upon the debtor making an observation, say $n+1$. We refer to this set of messages as the discharge notification set for observation $n+1$. The only caveat is that we must be careful to send notification only for every strongest commitment (because for any commitment that holds there are arbitrarily many weaker commitments that hold).

Below, to save space, we assume that the transaction identifier of the commitments involved is $\iota$.

*Definition* 11*:* $DI(z,n)$, the discharge notification set of $z$ at observation $n+1$ is $\{\mathsf{Declare}(z,y,u') : S(O_x^n) \vdash \mathsf{C}(z,y,r,u)$ and $S(O_z^n) \vdash \neg\mathsf{C}(z,y,s,v)$ such that $\mathsf{C}(z,y,s,v) \succ \mathsf{C}(z,y,r,u)$, and $S(O_z^{n+1}) \vdash u'$ such that $u' \vdash u\}$.

Definition 12 characterizes the messages a creditor should send to its debtors upon making an observation that detaches currently holding commitments (again, strongest).

*Definition* 12*:* $DEPO(z,n)$, the detach posterior notification set of $z$ at $n+1$ is $\{\mathsf{Declare}(z,x,s) : S(O_z^n) \vdash \mathsf{C}(x,z,(r \wedge s) \vee t,u) \wedge \neg\mathsf{C}(x,z,w,v) \wedge \neg s$ such that $\mathsf{C}(x,z,w,v) \succ \mathsf{C}(x,z,(r \wedge s) \vee t,u)$ and $S(O_z^{n+1}) \vdash s \wedge \neg u \wedge \neg\mathsf{removed}(\iota)\}$.

Definition 13 characterizes the messages that a creditor should send to the debtor of a commitment upon receiving the create of a commitment that would be at least partially detached because parts of the antecedent already hold.

*Definition* 13*:* $DEPR(z,n)$, the detach prior notification set of $z$ at $n+1$ is $\{\mathsf{Declare}(z,x,s) : S(O_z^n) \vdash s \wedge \neg\mathsf{C}(x,z,(r \wedge s) \vee t,u) \wedge \neg u \wedge \neg\mathsf{removed}(\iota)$ and $n+1$ is $z'$s observation $\mathit{receive}(x,\mathsf{Create}(x,z,(r \wedge s) \vee t,u))$ such that $s \not\equiv \top(\iota)\}$.

Definition 14 characterizes the set of all messages that an agent must send upon making an observation.

*Definition* 14*:* $NS(z,n)$, the notification set of agent $z$ at observation $n$ is $DI(z,n) \cup DEPO(z,n) \cup DEPR(z,n)$.

Definition 15 captures the state of an agent that has sent messages in the notification set for an observation.

*Definition* 15*:* Agent $z$ is notification-done for $n$ at $k$ iff $O_z^n \sqsubseteq O_z^k$ and for each message $m \in NS(z,n)$, $O_z^k \setminus O_z^n$ contains send$(z,m)$.

Messages sent from the notification set may themselves generate notifications. Definition 16 characterizes the idea of a closure of notifications.

*Definition* 16*:* Agent $z$ is notification-closed if and only if for any $n$ there exists a $k$ such that $z$ is notification-done for each $j$ $(n \leqslant j \leqslant k)$ at $k$.

Lemma 2 expresses an important property: If an agent merely sends the messages in the notification set for an observation $n$, without any other intervening observations, then the resulting sequence is guaranteed to be notification-closed. This enables us to treat the sending of notifications as atomic with the observation that made them necessary.

*Lemma* 2*:* Let $k$ be the smallest number such that $z$ is notification-done for $n$ at $k$. Then, $z$ is notification-closed.

*Definition* 17*:* A system $\mathcal{A}$ of $n$ agents $\{1 \ldots i \ldots n\}$ acceptably realizes an observation vector $O = [O_1 \ldots O_n]$ iff

- $\mathcal{A}$ realizes $O = [O_1 \ldots O_i \ldots O_n]$, and
- each $i$ is notification-closed.

Informally, Theorem 3 states that if the system acceptably realizes an observation vector that is incomplete with respect to a commitment, then the system will acceptably realize an observation vector that is complete with respect to the commitment. Theorem 3 guarantees completeness only with respect to strongest commitments. In other words, if the vector is complete with respect to, say, $\mathsf{C}(r,u \wedge v)$, then we do not care that it may be incomplete with respect to $\mathsf{C}(r,u)$ (which it may well be, as illustrated earlier).

*Theorem* 3*:* [Progress] If a system $\mathcal{A}$ acceptably realizes an observation vector $O$ that is incomplete with respect to $\mathsf{C}(r,u)$ and $\mathsf{C}(r,u)$ is a strongest incomplete commitment, then either $\mathcal{A}$ acceptably realizes an observation vector $O'$ such that $O \sqsubset O'$ and $O'$ is complete with respect to $\mathsf{C}(r,u)$ or $(\exists i \in \mathcal{A}, \mathsf{C}(s,v) : \mathsf{C}(s,v) \succ \mathsf{C}(r,u)$ and $\mathcal{S}(o_i) \vdash \mathsf{C}(s,v))$.

*Proof.* We sketch out a proof here. The proof is by induction on the length of the commitment expression, which is defined as the number of atoms in the expression. Let $\mathsf{C}(p, q)$ be a minimum length commitment (that is, of length two). By Definition 8, $O^0$ is acceptably realizable and complete with respect to $\mathsf{C}(p, q)$. Consider any complete vector $O^c$. Let $O^i$ be a smallest acceptable extension of $O^c$ that is incomplete with respect to $\mathsf{C}(p, q)$. This means that the following condition holds by Definition 8: $S(O^i_x) \vdash \neg\mathsf{C}(p, q)$ and $S(O^i_y) \vdash \mathsf{C}(p, q) \wedge \neg q \wedge \neg\mathsf{removed}(\iota)$. There are two primary cases to consider. (1) $S(O^{i-1}_x) \vdash \mathsf{C}(p, q)$, meaning that what made $O^i$ incomplete was that $x$ stopped inferring $\mathsf{C}(p, q)$. In this case, either a Cancel or a discharge notification is on the way to $y$, which when $y$ receives them will make the resulting vector complete. (2) $S(O^{i-1}_y) \vdash \neg\mathsf{C}(p, q)$, meaning that what made $O^i$ incomplete was that $y$ began inferring $\mathsf{C}(p, q)$. If it is because of a stronger create, we are done (by theorem statement). If it is by a detach, then acceptability means that a notification is on the way to $x$, which when it arrives will produce a complete vector.

Assume that the theorem holds for all commitments of length $k$. Then we must prove that it holds for commitments of length $k+1$. By inspection, we see that there two relevant cases. These correspond to $\mathrm{B}_4$ and $\mathrm{B}_5$, respectively.

Consider $\mathrm{B}_4$ first. Let $\mathsf{C}(r, u)$ and $\mathsf{C}(s, u)$ be each of length $k$ and $\mathsf{C}(r \vee s, u)$ of length $k + 1$. We know that $O^0$ is complete with respect to $\mathsf{C}(r \vee s, u)$. Let $O^c$ be any vector complete with respect to $\mathsf{C}(r \vee s, u)$. As before, let $O^i$ be the smallest incomplete acceptably realized extension with respect to $\mathsf{C}(r, u \wedge v)$. Then, we want to prove that the system acceptably realizes a complete extension of $O^i$. From the inductive hypothesis, we know there there will be an extension $O^j$ of $O^i$ that will be complete with respect to both $\mathsf{C}(r, u)$ and $\mathsf{C}(s, v)$. This means that $S(O^j_x) \vdash \neg\mathsf{C}(r, u)$ and $S(O^j_y) \vdash \mathsf{C}(r, u) \wedge \neg u \wedge \neg\mathsf{removed}(\iota)$ and $S(O^j_x) \vdash \neg\mathsf{C}(s, u)$ and $S(O^j_y) \vdash \mathsf{C}(s, u) \wedge \neg u \wedge \neg\mathsf{removed}(\iota)$. From these we infer that, $S(O^j_x) \vdash \neg\mathsf{C}(r \vee s, u)$ and $S(O^j_y) \vdash \mathsf{C}(r \vee s, u) \wedge \neg u \wedge \neg\mathsf{removed}(\iota)$. In other words, $O^j$ is complete with respect to $\mathsf{C}(r \vee s, u)$.

Reasoning about the other case with $\mathrm{B}_5$ is analogous. □

## 4.5 Revisiting the Motivating Scenarios

It is worth revisiting the enactments in Fig. 1 to see what their outcomes would be under our approach, specifically, when the commitments and operations on commitments are expressed in our syntax and include identifiers. Assume they involve the same transaction identifier.

Consider Fig. 1(A). Then, the second create is ineffective because of the earlier release; Bob and Alice are aligned (with respect to all commitments). Consider Fig. 1(B). Bob's observation of the second create is a noop because of the earlier release; Bob and Alice are aligned.

Consider Fig. 1(C). Bob's observation of the cancel removes $c_{UA}$; therefore, no commitment holds for either Bob or Alice; so they are aligned. Reasoning for Fig. 1(D) is analogous to (B).

Instead, consider the case that follows Fig. 1(A), but where the first create and release involve the same identifier and the second create involves a distinct identifier. Then, the second create makes a commitment with its identifier hold. Hence, Bob and Alice are still aligned.

## 5. DISCUSSION

We develop a novel treatment of alignment that addresses important shortcomings in C&S via a more general and simpler treatment than C&S. Our approach supports multiple transactions and does not require FIFO delivery. Compared to C&S, we require fewer kinds of notifications, fewer kinds of statives, and fewer reasoning postulates. We avoid C&S's *Principle of Priority* and the associated action rules, which make C&S's approach unnecessarily complex. Although we adopt the broad intuitions of C&S, our formal development is almost entirely novel, including the formal language (Table 3) and the definition of alignment (Definition 9).

Several works express protocols and commitments in first-order languages, e.g., the event calculus [4, 26]. However, a systematic treatment of information in protocols and commitments has a more recent history [7, 18, 19, 23]. Important questions here concern the identification of instances—of events, commitment, and protocol enactments. This paper is in the same vein and contributes to the proper treatment of commitments despite decentralization.

Some works [9, 10, 11, 17, 20] propose a purely syntactic notion of commitment identifiers. These approaches place an identifier in an explicit slot in a commitment expression but do not associate the identifier with the content (antecedent and consequent) of the commitment. Further, because these are syntactic approaches, they give unique identifiers to commitments. As C&S explain, this is problematic for reasoning over commitments. For example, from a commitment for $p \wedge q$, one would not be able to infer a commitment for $p$ (what would its identifier be?); nor would one be able to infer a commitment for $p \wedge q$ from commitments for $p$ and $q$. C&S proposed the semantic solution that commitments be identified based upon identifiers in the content. We adopt their core intuition though with enhancements to be able to accommodate distinct transactions.

Winikoff [25] notes problems with interoperability among agents enacting commitment protocols but does not provide a general solution. Kafalı et al. [15] consider it an overhead to send notifications for detaches and discharges. In a distributed system, where parties have incomplete information about relevant events, we consider notifications as essential for correct reasoning and a pragmatic way to ensure alignment. However, there may be opportunities for reducing the cost of notifications, for example, via piggybacking.

We foresee four important directions. One, implement our approach in an agent-oriented programming platform, such as JaCaMo [3]. Two, develop a general theory of commitment-based transactions. A fuller treatment of transactions that covers nested and multiparty cases would require a richer modeling of information in commitments. Approaches such as Cupid [7] could potentially be applied for this purpose. Three, develop a methodology and tool support for specifying commitment-based transactions. Protos [5] is a methodology for deriving commitment-based specifications from stakeholder requirements; it could be extended to commitment-based transactions. Four, develop a approach for alignment that applies to all kinds of social expectations, including norms that capture security requirements [24].

## Acknowledgments

Thanks to the anonymous reviewers for comments.

## 6. REFERENCES

[1] M. Baldoni, C. Baroglio, A. K. Chopra, N. Desai, V. Patti, and M. P. Singh. Choice, interoperability, and conformance in interaction protocols and service choreographies. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, pages 843–850, Budapest, 2009. IFAAMAS.

[2] M. Baldoni, C. Baroglio, E. Marengo, and V. Patti. Constitutive and regulative specifications of commitment protocols: A decoupled approach. *ACM Transactions on Intelligent Systems and Technologies*, 4(2):22:1–22:25, 2013.

[3] O. Boissier, R. H. Bordini, J. F. Hübner, A. Ricci, and A. Santi. Multi-agent oriented programming with JaCaMo. *Science of Computer Programming*, 78(6):747–761, June 2013.

[4] F. Chesani, P. Mello, M. Montali, and P. Torroni. Representing and monitoring social commitments using the event calculus. *Autonomous Agents and Multi-Agent Systems*, 27(1):85–130, 2013.

[5] A. K. Chopra, F. Dalpiaz, F. B. Aydemir, P. Giorgini, J. Mylopoulos, and M. P. Singh. Protos: Foundations for engineering innovative sociotechnical systems. In *Proceedings of the 18th IEEE International Requirements Engineering Conference (RE)*, pages 53–62, 2014. IEEE Computer Society.

[6] A. K. Chopra and M. P. Singh. Multiagent commitment alignment. In *Proceedings of the Eighth International Conference on Autonomous Agents and Multiagent Systems*, pages 937–944. IFAAMAS, 2009.

[7] A. K. Chopra and M. P. Singh. Cupid: Commitments in relational algebra. In *Proceedings of the 23rd Conference on Artificial Intelligence (AAAI)*, pages 1–8, Austin, Texas, Jan. 2015. AAAI Press.

[8] S. Cranefield and M. Winikoff. Verifying social expectations by model checking truncated paths. In *Coordination, Organizations, Institutions and Norms in Agent Systems IV*, volume 5428 of *LNCS*, pages 204–219. Springer, 2009.

[9] M. El-Menshawy, J. Bentahar, and R. Dssouli. A new semantics of social commitments using branching space-time logic. In *International Joint Conferences on Web Intelligence and Intelligent Agent Technologies*, pages 492–496, 2009.

[10] R. A. Flores, P. Pasquier, and B. Chaib-draa. Conversational semantics sustained by commitments. *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS)*, 14(2):165–186, Apr. 2007.

[11] N. Fornara and M. Colombetti. Operational specification of a commitment-based agent communication language. In *Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 535–542. ACM Press, July 2002.

[12] N. Fornara and M. Colombetti. A commitment-based approach to agent communication. *Applied Artificial Intelligence*, 18(9-10):853–866, 2004.

[13] A. Günay, M. Winikoff, and P. Yolum. Generating and ranking commitment protocols. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems*, pages 1323–1324. 2013.

[14] A. J. I. Jones and M. J. Sergot. A formal characterisation of institutionalised power. *Logic Journal of the IGPL*, 4(3):427–443, June 1996.

[15] Ö. Kafalı, F. Chesani, and P. Torroni. What happened to my commitment? Exception diagnosis among misalignment and misbehavior. In *Proceedings of the 11th International Workshop on Computational Logic in Multi-Agent Systems*, volume 6245 of *LNCS*, pages 82–98. Springer, 2010.

[16] L. Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, 21(7):558–565, July 1978.

[17] A. U. Mallya, P. Yolum, and M. P. Singh. Resolving commitments among autonomous agents. In F. Dignum, editor, *Advances in Agent Communication, International Workshop on Agent Communication Languages, ACL 2003*, volume 2922 of *LNCS*, pages 166–182. Springer, 2004.

[18] F. Meneguzzi, P. R. Telang, and M. P. Singh. A first-order formalization of commitments and goals for planning. In *Proceedings of the 23rd Conference on Artificial Intelligence (AAAI)*, pages 697–703, Bellevue, Washington, July 2013. AAAI Press.

[19] M. Montali, D. Calvanese, and G. D. Giacomo. Verification of data-aware commitment-based multiagent system. In *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems*, pages 157–164, Paris, May 2014.

[20] M. Rovatsos. Dynamic semantics for agent communication languages. In *Proceedings of the 6th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 100–107, Honolulu, May 2007. IFAAMAS.

[21] M. P. Singh. An ontology for commitments in multiagent systems: Toward a unification of normative concepts. *Artificial Intelligence and Law*, 7(1):97–113, Mar. 1999.

[22] M. P. Singh. Semantical considerations on dialectical and practical commitments. In *Proceedings of the 23rd Conference on Artificial Intelligence*, pages 176–181, 2008.

[23] M. P. Singh. Information-driven interaction-oriented programming: BSPL, the blindingly simple protocol language. In *Proceedings of the 10th International Conference on Autonomous Agents and MultiAgent Systems*, pages 491–498, 2011.

[24] M. P. Singh. Cybersecurity as an application domain for multiagent systems. In *Proceedings of the 14th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 1–4, Istanbul, May 2015. IFAAMAS. Blue Sky Ideas Track.

[25] M. Winikoff, W. Liu, and J. Harland. Enhancing commitment machines. In *Proceedings of the 2nd International Workshop on Declarative Agent Languages and Technologies (DALT)*, volume 3476 of *LNAI*, pages 198–220, Berlin, 2005. Springer-Verlag.

[26] P. Yolum and M. P. Singh. Flexible protocol specification and execution: Applying event calculus planning using commitments. In *Proceedings of the 1st International Joint Conference on Autonomous Agents and MultiAgent Systems*, pages 527–534. 2002.