Knowledge and the ordering of events in distributed systems Extended Abstract

Paul J. Krasucki
Dept. of Math. Sciences
Rutgers University
Camden College
Camden, NJ 08102
krasucki@crab.rutgers.edu

R. Ramanujam
The Institute of Mathematical Sciences
C.I.T. Campus
Madras - 600 113
India
jam@imsc.ernet.in

ABSTRACT

In asynchronous distributed systems logical time is usually interpreted as "possible causality", a partial order on event occurrences. We investigate the relationship between passage of time and changes in the knowledge of agents. We show that there is a certain duality between knowledge transition systems (defined here to model changes in the states of knowledge of agents) and partially ordered sets of event occurrences (the model of n-Asynchronously Communicating Sequential Agents).

1 Introduction

Consider a distributed system of n *agents* acting autonomously. Agents communicate by passing messages from one to another. Assume that every message sent is eventually delivered to the intended recipient and that messages are delivered in the order in which they were sent. Such a model is standard in the theory of distributed computing.

Lamport [Lam] discussed the ordering of event occurrences in such systems and argued that each agent 'locally' sees a linear order of event occurrences whereas 'globally', only a partial order consistent with the local linear ones is available. In this discussion, the ordering refers to causal dependency between event occurrences and thus incomparability under the ordering denotes causal independence, and therefore (in a sense) concurrency.

If we see concurrency as causal independence, one way of phrasing the assertion "event occurrences e_1 and e_2 can be concurrent" is: "no agent in the system *knows* that e_1 must precede e_2 or that e_2 must precede e_1". In a sense, this identifies the states of the system with the states of knowledge of agents in the system.

After Halpern and Moses [HM] there has been extensive work done in the study of knowledge states of agents in distributed systems. In particular, looking at how the occurrence

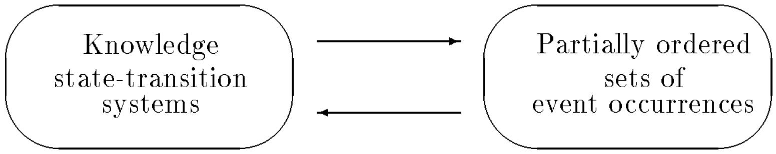

Figure 1:

of an event can cause a change in agents' states of knowledge leads to viewing distributed protocols as goal-oriented activity. Such a protocol is then a transformation from the given initial state of knowledge in the system to a desired state where agents know some specific facts.

We wish to study to what extent these notions of agents' knowledge (specified as equivalence relations on states, one relation for each agent) and partial orders on event occurrences are *dual*. In Figure 1, can we go back and forth without losing information about agents' behavior?

In the process, we would also like to understand more precisely assertions like the following ones: "an agent cannot lose knowledge by receiving a message", "an agent cannot gain knowledge by sending a message" and so on. Such statements are commonly used in the analysis of distributed protocols, and do make intuitive sense.

Similar questions have been addressed in the literature, but in different contexts: Chandy and Misra [CM] have related chains of messages to change in knowledge of agents, and Parikh and Krasucki [PK] have precisely characterized levels of knowledge of agents (for a formula) and specified what sequences of messages are required to attain a given level. In the area of knowledge- based protocols there has been extensive work relating states of knowledge of agents and message histories (see [DM], [HZ], [Ma] for some expositions). However, the question we study here is the formal relationship between knowledge structures specified as transition systems and temporal structures specified as partially ordered sets of event occurrences. In a sense, it is closer to the spirit of [Pra], [NPW].

In the following sections we show that there is a simple class of transition systems (Knowledge Transition Systems, KTSs for short) enriched with equivalence relations on states, which corresponds to a natural partial order model of event occurrences in distributed systems (Asynchronously Communicating Sequential Agents, abbreviated ACSAs). This correspondence is precise in the following sense: we associate a KTS with an ACSA and conversely an ACSA with a KTS in such a way that ACSA $\rightarrow$ KTS $\rightarrow$ ACSA is an isomorphism, and KTS $\rightarrow$ ACSA $\rightarrow$ KTS is a simulation.

2 ACSAs

In [LRT], a model of distributed systems has been defined in the following manner: assume a collection of n agents, each of which is sequential, interacting by message passing. Each agent is 'tree-like', in the sense that its behaviour is given as a 'backwards-linear' poset of event occurrences. The formal definition is as follows:

Def 1 : A system of n-Asynchronously Communicating Sequential Agents (n-ACSA) is a triple $\mathrm{E} = (E, \leq, \eta)$ where $n \in \mathcal{N}, n > 0$, and

(i) E is a set of *event occurrences*,

(ii) $\leq \subseteq E \times E$ is a partial order called the *causality relation*, and

(iii) $\eta : E \rightarrow \{1, ..., n\}$ is a *naming function* such that
$\forall e \forall i \in \{1, ..., n\}\ \{e' | e' \leq e\} \cap \eta^{-1}(i)$ is totally ordered by $\leq$.

□

We will use the notation $\downarrow x$ for the initial segment of the poset $(X, \leq)$ up to the element x, i.e. for $x \in X$, $\downarrow x = \{y | y \leq x\}$; similarly for $X' \subseteq X$, $\downarrow X' = \{y | \exists x \in X',\ y \leq x\}$. We will also speak of ACSAs, leaving n implicit.

An agent is simply the set of event occurrences having the same name. We will often speak of the set $E_i \stackrel{\text{def}}{=} \eta^{-1}(i)$ as agent i. Note that condition (iii) imposes backward-linearity, making each agent tree-like. It is in this sense that the agents are sequential. Since η is a function, each event occurrence is uniquely 'owned' by an agent, forcing asynchrony in communication; any communication is necessarily split into 'send' and 'receive'.

$\leq$ is a causality relation in the sense that when $e_1 \leq e_2$, every observation of event occurrence e_2 necessarily implies an earlier observation of e_1. For example when $e_1 \leq e_2$, $\eta(e_1) = i$, $\eta(e_2) = j \neq i$ and there is no e_3 'between' e_1 and e_2, we can interpret e_2 as the receipt by j of a message from i, where e_1 constitutes the sending of this message. Clearly, causality here is in the sense that the sending of a message causally precedes its receipt.

Given such a notion of causality, a computation in an ACSA is simply a downward-closed set of event occurrences. This leads us to notions of conflict and concurrency in ACSAs. When we consider event occurrences which are incomparable under the causality ordering, conflicting ones are those which cannot both occur in the same computation, and concurrent ones are those which can. These are formally given below.

We say that event occurrences e_1 and e_2 are in *local conflict* when neither $e_1 \leq e_2$, nor $e_2 \leq e_1$, and $\eta(e_1) = \eta(e_2)$. Since agents are sequential, we do not interpret causal independence within agents as potential concurrency but as denoting choice made in computation. e_1 and e_2 are in *conflict*, if and only if there exist $e_1' \leq e_1, e_2' \leq e_2$ such that e_1' and e_2' are in local conflict.

When are two event occurrences concurrent ? We can say that e_1 and e_2 are *concurrent* if $\eta(e_1) \neq \eta(e_2)$, e_1 and e_2 are incomparable under the causal ordering, and $\downarrow e_1 \cup \downarrow e_2$

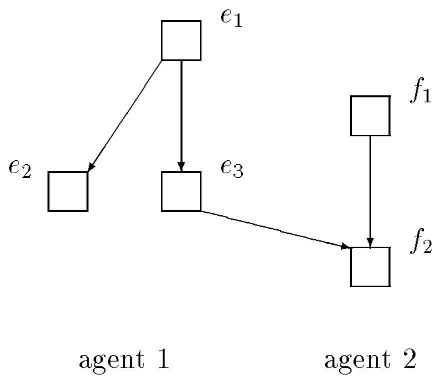

Figure 2: e_2 and e_3 are in local conflict,e_2 and f_2 in conflict (non-local)

is conflict-free. These three notions, namely causality, conflict and concurrency form the backbone of the behaviour theory of distributed systems. Note that for any $e_1, e_2 \in E$, we have $e_1 \leq e_2$ or $e_2 \leq e_1$ or e_1 is in conflict with e_2 or e_1 and e_2 are concurrent.

We can now define a notion of a *global state* in an ACSA: $c \subseteq E$ is a global state iff $\downarrow c \subseteq c$ and c is conflict-free. Thus a state being downward-closed and conflict-free, can be thought of as a partial run. We will often refer to global states as *configurations*. Note that the empty set is always a configuration. More importantly, for any $e \in E$, the set $\downarrow e$ is a configuration, a fact which we will use crucially later on.

Consider the 2-ACSA in Figure 2. Events e_2 and e_3 are in local conflict, whereas e_2 and f_2 are in conflict. e_2 and f_1 are concurrent. For this example, $\{e_1, e_3, f_1\}$ is a configuration, whereas $\{e_3, f_1\}$ and $\{e_1, e_2, e_3, f_1, f_2\}$ are not.

We say that an ACSA is *finitary* if and only if $\downarrow e$ is finite, for every $e \in E$. In the context of computation, it is natural to restrict attention to finitary ACSAs, and we will do precisely that. In fact, we will also be interested only in the *finite* configurations of finitary ACSAs. Note that finitariness implies discreteness of the partial order and hence we will freely make use of $<$, the covering relation of $\leq$.

A configuration in an ACSA is a state in the sense that event occurrences cause state transitions. This is seen precisely in the transition system $T\mathcal{E}$ associated with the finitary ACSA $\mathcal{E}$. Let C denote the set of finite configurations of $\mathcal{E}$. $T\mathcal{E} \stackrel{\text{def}}{=} (C, \rightarrow)$, where $c \rightarrow c'$ iff $\exists e \in E, c' = c \oplus \{e\}$. (Here, and in what follows, we use the notation $c \oplus \{e\}$ to mean the set $c \cup \{e\}$ alongwith the assertion that $e \notin c$). The transition system describes a connected acyclic graph, with an initial state, namely the empty configuration (it has no in-coming edges, and every other node is reachable from it).

This transition system has the following interesting properties:

Proposition 2 : Suppose $c, c_1, c_2 \in C$ and $e_1, e_2 \in E$ such that $e_1 \neq e_2, c_1 = c \oplus \{e_1\}$ and $c_2 = c \oplus \{e_2\}$. Then

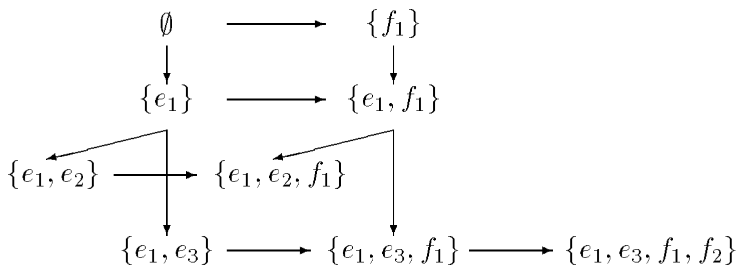

Figure 3: Transition system associated with the 2-ACSA of Figure 2.

(i) if $\eta(e_1) = \eta(e_2)$, then for every $c' \in C$ such that $c_1 \subseteq c', e_2 \notin c'$, and for every $c'' \in C$ such that $c_2 \subseteq c'', e_1 \notin c''$.

(ii) if $\eta(e_1) \neq \eta(e_2)$ then there exists a configuration $c' \in C$ such that $c' = c_1 \oplus \{e_2\}$ and $c' = c_2 \oplus \{e_1\}$.

$\square$

We can speak of the *view* of an agent at a state: the view of agent i at configuration c is simply, $c \cap E_i$. Given this, the knowledge of agents can be defined as equivalence relations: $c \approx_i c'$ iff $c \cap E_i = c' \cap E_i$ for $i \in \{1, ..., n\}$. Note that every event occurrence changes the knowledge of exactly one agent, by this definition.

In the next section, we define a class of Knowledge Transition Systems and show that $T\mathcal{E}$ belongs there. We can then come to the central question of the paper : which class of knowledge transition systems are those associated (in the above manner) with ACSAs ?

3 Knowledge Transition Systems

As in the case of n-ACSAs, we speak of systems of n agents, where $n \in \mathcal{N}, n > 0$, and in discussions often implicitly assume a fixed n without specifying it.

Def 3 : A Knowledge Transition System of n agents is a quadruple $\mathcal{K} = (S, \rightarrow, s_0, Eq_n)$ where

(a) S is a set of states (at most countable),

(b) $\rightarrow \subseteq S \times S$ is a transition relation on S,

(c) $s_0 \in S$ is an initial state such that every state in S is reachable from s_0 by paths through $\rightarrow$, and

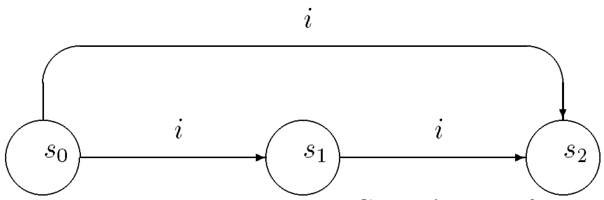

Figure 4: Granularity of transitions in KTS.

(d) $Eq_n = (\approx_1, ..., \approx_n)$ is an n-tuple of equivalence relations $\approx_i \subseteq S \times S$ satisfying the following conditions:

(i) for all $s, s' \in S$, if $s \rightarrow s'$, then there exists a unique i such that $s \not\approx_i s'$. [we can therefore define $\Rightarrow \subseteq S \times \{1, ..., n\} \times S$ by $s \overset{i}{\Rightarrow} s'$ iff $s \rightarrow s'$ and $s \not\approx_i s'$],

(ii) for all $s_1, s_2, s_3 \in S$ and $i \neq j \in \{1, ..., n\}$, if $s_1 \overset{i}{\Rightarrow} s_2$ and $s_1 \overset{j}{\Rightarrow} s_3$ then there exists $s_4 \in S$, such that $s_2 \overset{j}{\Rightarrow} s_4$ and $s_3 \overset{i}{\Rightarrow} s_4$, and

(iii) for all $s_1, s_2, s_3 \in S$ and $i \in \{1, ..., n\}$, if $s_1 \overset{i}{\Rightarrow} s_2$ and $s_1 \overset{i}{\Rightarrow} s_3$ and $s_2 \neq s_3$, then $s_2 \not\approx_i s_3$.

□

Condition (i) refers to the locality of event occurrences in the systems we wish to study and to the fact that observers see only one event occurrence at a time (this condition can be relaxed, and we will discuss that later on). In a sense, this ensures that transitions in KTSs correspond to event occurrences in ACSAs.

Note that we still have considerable flexibility in the granularity of transitions. We can have the situation as in Figure 4: where transitions $s_0 \rightarrow s_1$ and $s_1 \rightarrow s_2$ can mean "add 1 to local variable y", whereas $s_0 \rightarrow s_2$ is "add 2 to local variable y".

Condition (ii) ensures *confluence*. If both an i-action and a j-action are enabled at a state, $i \neq j$, then they can be performed in any order. Such 'forward-diamond conditions' are typical in models of 'true' concurrency.

Condition (iii) asserts that every agent knows the effect of making a local choice. This corresponds with the intuition that the two i-transitions enabled at the same state are in local conflict.

In the previous section, we discussed how a transition system with equivalence relations can be associated with an ACSA. As may be expected, it is a KTS, as the following proposition asserts:

Proposition 4 : Let $\mathcal{E} = (E, \leq, \eta)$ be a finitary n-ACSA, and let C denote the set of all finite configurations of $\mathcal{E}$. Define the structure $K = (C, \rightarrow, \emptyset, Eq_n)$ by:

(a) $c \rightarrow c'$ iff there exists $e \in E$ such that $c' = c \oplus \{e\}$, and

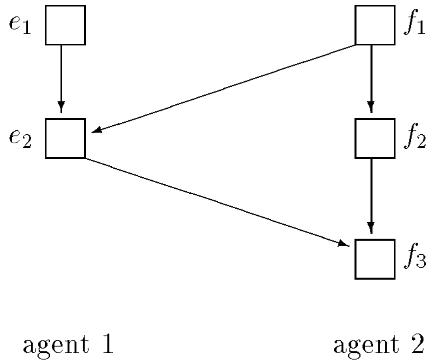

Figure 5: 2-ACSA

$$
\begin{array}{lll}
s_0 = \emptyset & s_3 = \{e_1\} & s_6 = \{e_1, e_2, f_1\} \\
s_1 = \{f_1\} & s_4 = \{e_1, f_1\} & s_7 = \{e_1, e_2, f_1, f_2\} \\
s_2 = \{f_1, f_2\} & s_5 = \{e_1, f_1, f_2\} & s_8 = \{e_1, e_2, f_1, f_2, f_3\}
\end{array}
$$

$$
\begin{tikzcd}
s_0 \arrow[r] \arrow[d] & s_1 \arrow[r] \arrow[d] & s_2 \arrow[d] & \\
s_3 \arrow[r] & s_4 \arrow[r] \arrow[d] & s_5 \arrow[d] & \\
& s_6 \arrow[r] & s_7 \arrow[r] & s_8
\end{tikzcd}
$$

Figure 6: KTS associated with the 2-ACSA from Figure 5. Horizontal arrows are $\overset{2}{\Rightarrow}$, vertical arrows are $\overset{1}{\Rightarrow}$.

(b) $Eq_n = (\approx_1, ..., \approx_n)$, where $\approx_i \subseteq C \times C$ is given by: $c \approx_i c'$ iff $c \cap E_i = c' \cap E_i$, for all $i \in \{1, ..., n\}$.

Then K is a KTS. $\square$

Consider the 2-ACSA in Figure 5 :

Figure 6 shows the transition system associated with it.

Note that in Figure 6, s_5 is the latest state where agent 1 could have sent the message to agent 2 (event occurrence e_2), and s_8 is the earliest state where agent 2 could receieve that message. We also have $s_5 \overset{1}{\Rightarrow} s_7 \overset{2}{\Rightarrow} s_8$ but there is no state s such that $s_5 \overset{2}{\Rightarrow} s \overset{1}{\Rightarrow} s_8$. A similar remark applies for $s_3 \overset{2}{\Rightarrow} s_4 \overset{1}{\Rightarrow} s_6$ in the context of the message sent from agent 2 to agent 1 (event occurrence f_1). We thus have a situation where $s_5 \approx_2 s_7$ but the capabilities of agent 2 are different in the two states!

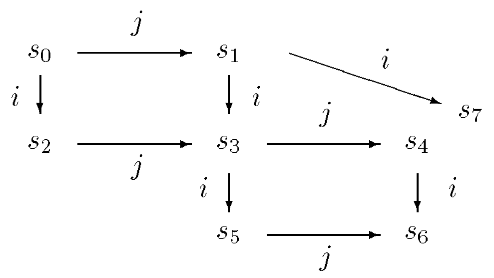

Figure 7: KTS event types: concurrency $((s_0, s_1)$ and $(s_0, s_2))$; local conflict: $((s_0, s_2)$ and $(s_1, s_7))$; conflict: $((s_5, s_6)$ and $(s_1, s_7))$; local causality: $((s_0, s_2)$ and $(s_3, s_5))$ and remote causality: $((s_1, s_3)$ and $(s_5, s_6))$

This brings us to the crucial point: situations where we have $s_1 \stackrel{i}{\Rightarrow} s_2 \stackrel{j}{\Rightarrow} s_3$ but without any s such that $s_1 \stackrel{j}{\Rightarrow} s \stackrel{i}{\Rightarrow} s_3$ represent messages from agent i to j.

Now we will identify 'event types' specified in the KTS and then we will proceed to associate an ACSA with a given KTS. Consider Figure 7, where we depict four different situations that occur in KTSs, which are intuitively interpreted as concurrency, conflict (local as well as inherited), local causality and remote causality. When the (cyclic) KTS is unfolded, these event types yield event occurrences. To this end we look at paths in the KTS.

For the rest of the section fix a KTS $\mathcal{K} = (S, \rightarrow, s_0, Eq_n)$. Define $R_{\mathcal{K}} \subseteq S^*$, the *runs* of the KTS, to be the set of sequences $t_1...t_k, k \geq 0$, where $t_1 = s_0$, and for $1 \leq l < k, t_l \rightarrow t_{l+1}$. We use δ, δ' etc to range over elements of R (we will often omit the subscript K, when the context is clear). Usually we will be interested only in non-null runs in the KTS. Let $R^+ \stackrel{\text{def}}{=} \{\delta \in R \mid |\delta| > 1\}$; these have at least one state transition. Define $\tau : R^+ \rightarrow \{1, ..., n\}$ by: $\tau(\delta) = i$ where $\delta = \delta' s s'$ and $s \stackrel{i}{\Rightarrow} s'$. We use the notation $\preceq$ to denote the prefix ordering on sequences.

Let $\delta = t_1...t_l \stackrel{i}{\Rightarrow} t_{l+1} \stackrel{i_1}{\Rightarrow} ... \stackrel{i_q}{\Rightarrow} t_p \stackrel{j_1}{\Rightarrow} t_{p+1} ... \stackrel{j_r}{\Rightarrow} t_{m-1} \stackrel{j}{\Rightarrow} t_m ... t_n$ be a run of the KTS. We will call a pair of transitions $t_l \stackrel{i}{\Rightarrow} t_{l+1}$ and $t_{m-1} \stackrel{j}{\Rightarrow} t_m$ *concurrent* iff there exist a run of the KTS, $\delta' = t_1...t_l \stackrel{j_1}{\Rightarrow} u_{p+1} ... \stackrel{j_r}{\Rightarrow} u_{m-1} \stackrel{j}{\Rightarrow} u_m$ such that for all v, $p+1 \leq v \leq m, t_v \neq u_v$, and sets $I_1 = \{i, i_1, i_2, ..., i_q\}$ and $I_2 = \{j_1, j_2, ..., j_r, j\}$ are disjoint and $u_{p+1} \approx_{j_1} t_{p+1}$, $u_{p+2} \approx_{j_2} t_{p+2}$,...,$u_m \approx_j t_m$.

Note that this implies that there is a 'grid' of transitions in KTS between t_l and t_m.

Consider a run $\delta \in R^+$, say, $t_1...t_k (k > 1)$. Let $\tau(\delta) = i$ and let $j \neq i$. In such a case, we say that the subsequence $t_1...t_l$, $(1 < l < k)$ is the *j-predecessor* of δ if $t_{l-1} \stackrel{j}{\Rightarrow} t_l$ is maximal

j-transition in δ not concurrent with $t_{k-1} \stackrel{i}{\Rightarrow} t_k$. If no such l exists, define the j-predecessor of δ to be the null sequence. Note that the j-predecessor is defined now for all $j \neq \tau(\delta)$.

The intuition behind the above definition should be obvious: within a run, this helps us identify event occurrences which constitute receipt of messages and match them with corresponding send occurrences.

As an example suppose $\delta = s_0 \stackrel{j}{\Rightarrow} s_1 \stackrel{k}{\Rightarrow} s_2 \stackrel{i}{\Rightarrow} s_3$. Assume i, j, k are distinct. If there exists s such that $s_1 \stackrel{i}{\Rightarrow} s$ and $s \approx_i s_3$, then $s \stackrel{k}{\Rightarrow} s_3$. Then the transitions (s_1, s_2) and (s_2, s_3) are concurrent and cannot be interpreted as a send-receive pair. Otherwise, we have the lack of "forward-confluence" referred to above, which we interpret as message passing. On the other hand, we can have $s_1 \stackrel{i}{\Rightarrow} s$, but no $s_0 \stackrel{i}{\Rightarrow} s'$ such that $s \approx_i s' \approx_i s_3$. This again leads us to interpret (s_0, s_1) as a send and (s_2, s_3) as a receive. When neither s nor s' exists, we can think of a message from j to i routed through k.

Proposition 5 : Suppose $\delta_1 \preceq \delta_2, \tau(\delta_1) \neq i$ and $\tau(\delta_2) \neq i$. Then the i-predecessor of δ_1 is also a prefix of the i-predecessor of δ_2. □

Let $[s]_i \stackrel{\text{def}}{=} \{s' \in S | s \approx_i s'\}, S_i \stackrel{\text{def}}{=} \{[s]_i | s \in S\}, 1 \leq i \leq n$. Define $L_i \subseteq S_i^*$, the *local i-runs* of the KTS, to be the set of sequences $x_1 ... x_k, k \geq 0$, where $s_0 \in x_1$, and for $1 \leq l < k$, there exist $s \in t_l, s' \in t_{l+1}$ such that $s \stackrel{i}{\Rightarrow} s'$. A *local run* is a local i-run, for some $1 \leq i \leq n$. We use ρ, ρ' etc. to range over elements of local runs. The following proposition asserts that an i-run cannot also be (usually) a j-run: the exceptions are, the null sequence and the singleton sequence when $\stackrel{i}{\Rightarrow}$ is empty.

Proposition 6 : Suppose $\rho \in L_i, |\rho| > 1$. Then $\rho \notin L_j$, for every $j \neq i$.

Proof : Suppose $\rho \in L_i, \rho = x_1 x_2 \rho'$. Let $s \in x_1, s' \in x_2$ such that $s \stackrel{i}{\Rightarrow} s'$. Clearly, $s \in [s_0]_i, s' \notin [s_0]_i$ and for $j \neq i, \{s, s'\} \subseteq [s_0]_j$. Therefore there can be no $\rho'' \in L_j$ starting with x_1. □

Define the projection maps $\phi_i : R \rightarrow L_i$ as follows: let $\delta = t_1 ... t_k, k \geq 0$.

$\phi_i(\delta) =$ null, if there is no $1 \leq l < k$ such that $s_l \stackrel{i}{\Rightarrow} s_{l+1}$, and

$\phi_i(\delta) = [s]_i [s']_i \phi_i(\delta_2)$, if $\delta = \delta_1 s s' \delta_2, s \stackrel{i}{\Rightarrow} s'$ and $\phi_i(\delta_1) =$ null.

Let the map $\epsilon : R^+ \rightarrow L_1 \times ... \times L_n$ be defined as follows: $\epsilon(\delta) = (\phi_1(\delta_1), ..., \phi_n(\delta_n))$, where $\delta_i = \delta$, if $i = \tau(\delta)$, and δ_i is the i-predecessor of δ otherwise. Clearly the map ϵ is well-defined. We use ϵ to associate an ACSA with the KTS.

Proposition 7 : Let $\mathcal{K} = (S, \rightarrow, s_0, Eq_n)$ be a KTS. Define the structure $\mathcal{E} = (E, \leq, \eta)$ by:

$E \stackrel{\text{def}}{=} \{\epsilon(\delta) | \delta \in R^+ |\}$,

$\leq \stackrel{\text{def}}{=} \{(\epsilon(\delta_1), \epsilon(\delta_2)) |$ there exists $\delta \preceq \delta_2, \epsilon(\delta) = \epsilon(\delta_1)$, and either $(\tau(\delta) = \tau(\delta_1)$, or (there exists δ' such that $\tau(\delta) = \tau(\delta') = j \neq \tau(\delta_2), \delta \preceq \delta' \preceq \delta_2$ and δ' is the j-predecessor of δ_2) $\}$, and

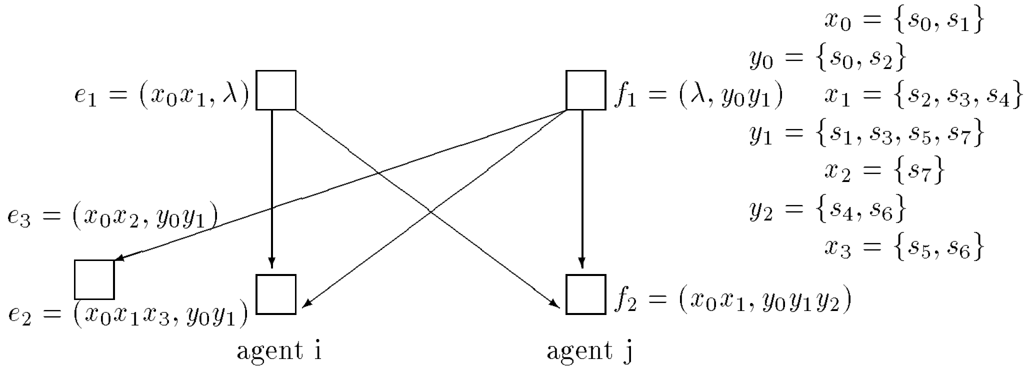

Figure 8: ACSA associated with the KTS from Figure 7.

$\eta(\epsilon(\delta)) \stackrel{\text{def}}{=} \tau(\delta)$.

Then $\mathcal{E}$ is an n-ACSA.

Proof : Reflexivity and antisymmetry of $\leq$ are trivial. To prove transitivity, suppose that $e_1 \leq e_2 \leq e_3$ and let $e_1 = \epsilon(\delta_1)$, $e_2 = \epsilon(\delta_2)$, $e_3 = \epsilon(\delta_3)$, $\delta_1 \preceq \delta_2 \preceq \delta_3$. Clearly $\delta_1 \preceq \delta_3$, so we are done if $\tau(\delta_1) = \tau(\delta_3)$.

Hence suppose that $\tau(\delta_1) = i \neq j = \tau(\delta_3)$. Let δ_3' be the i-predecessor of δ_3. We now have two cases: $\tau(\delta_2) = i$. Then $\delta_1 \preceq \delta_2 \preceq \delta_3'$, and we are done. Now consider the case when $\tau(\delta_2) \neq i$ and let δ_2' be the i- predecessor of δ_2. Since $\delta_1 \preceq \delta_2'$ and by Proposition 5 above, $\delta_2' \preceq \delta_3'$, we get $\delta_1 \preceq \delta_3'$, as required.

To check that $\leq$ is backwards linear within agents, consider $e_1 \leq e_3$ and $e_2 \leq e_3$ such that $\eta(e_1) = \eta(e_2) = i$, say. Let $e_1 = \epsilon(\delta_1)$, $e_2 = \epsilon(\delta_2)$, $e_3 = \epsilon(\delta_3)$ and $\delta_1 \preceq \delta_3, \delta_2 \preceq \delta_3$. Clearly $\delta_1 \preceq \delta_2$ or $\delta_2 \preceq \delta_1$. By definition of $\leq$, we see that $e_1 \leq e_2$ in the former case, and that $e_2 \leq e_1$ in the latter. Thus $\downarrow e_3 \cap E_i$ is ordered by $\leq$, as required.

$\square$

We refer to the structure $\mathcal{E}$ as the ACSA associated with $\mathcal{K}$ and denote it $\mathcal{E}_{\mathcal{K}}$.

In Figure 8, we have an example of the ACSA associated with the KTS from Figure 7.

4 Forth and Back

Given an ACSA $\mathcal{E}$, we can associate a KTS $\mathcal{K}_{\mathcal{E}}$ with it, and with this KTS we can associate an ACSA $\mathcal{E}_{\mathcal{K}_{\mathcal{E}}}$. What is the relationship between $\mathcal{E}$ and $\mathcal{E}_{\mathcal{K}_{\mathcal{E}}}$?

Theorem 8 : Let $ES = (E, \leq, \eta)$ be a finitary n-ACSA, and let $\mathcal{E}_{\mathcal{K}_\mathcal{E}} = (E', \leq', \eta')$. Then there is a bijection $f : E \rightarrow E'$ such that :

$\forall e, e' \in E, e \leq e'$ iff $f(e) \leq' f(e')$ and

$\forall e \in E, \eta(e) = \eta'(f(e))$.

In other words, $\mathcal{E}_{\mathcal{K}_\mathcal{E}}$ is isomorphic to $\mathcal{E}$.

Proof : Let $\mathcal{E} = (E, \leq, \eta)$ be the given ACSA, and fix $\mathcal{K}_\mathcal{E}$ to be $\mathcal{K}$. Let C denote the set of finite configurations of $\mathcal{E}$. For $c \in C$, let $[c]_i$ stand for the equivalence class of c under the relation $\approx_i$. Given $e \in E, \eta(e) = i$, consider the set $\{[\downarrow e']_i | e' \leq e, \eta(e') = i\}$; since $\mathcal{E}$ is finitary and backwards linear within agents, this set can be written as a finite sequence of elements $[\downarrow e_1]_i, ..., [\downarrow e_k]_i$, where for $j \in \{1, ..., k-1\}, e_j < e_{j+1}$ and $e_k = e$; call this sequence, prefixed by $\{\emptyset\}$, the local history of i at e, denoted $lh(e)$. Now consider the tuple $< \rho_1, ..., \rho_n >$, where for $j \in \{1, ..., n\}, \rho_j$ is the null sequence if $\downarrow e \cap E_j = \emptyset$, and otherwise it is $lh(e')$, where e' is the maximal j-event in $\downarrow e$. Call this tuple the i-view at e.

We claim that the i-view at e is indeed an i-event occurrence in $\mathcal{K}$ and hence a member of E'. To see this, observe firstly that for any $e' \in E$, $lh(e')$ is a local j-run in $\mathcal{K}$ where $\eta(e') = j$, and $|lh(e')| > 1$. We only need to check that every such i-view is generated as $\epsilon(\delta)$ for some run δ in $\mathcal{K}$.

A *schedule* of a configuration $c \in C$ is a sequence $e_1 ... e_k$ such that $l \neq m$ implies $e_l \neq e_m, c = \{e_1, ..., e_k\}$, and if $e_l \leq e_m$, then $l \leq m$. Note that every schedule (of any configuration) corresponds to a run in K, and conversely that every run from $\emptyset$ to a configuration $c \in C$ defines a schedule of c.

Let $e_1 ... e_k$ be a schedule of $\downarrow e_k, \eta(e_k) \neq j$. Suppose $\downarrow e_k \cap E_j \neq \emptyset$ and e_l is the maximal j-event occurrence in $\downarrow e_k$. It can be easily seen that the schedule $e_1 ... e_l$ (of $\downarrow e_l = c'$, say) is the j-predecessor of $e_1 ... e_k$: since $e_l \in \downarrow e_k$, e_l and e_k cannot be concurrent. This shows that $\epsilon(\delta)$ is exactly the i-view at e_k, where $\delta = \emptyset\{e_1\}...\{e_1, ..., e_k\}, e_1 ... e_k$ is a schedule of $\downarrow e_k$ and $\eta(e_k) = i$.

We can thus meaningfully define the map $F : E \rightarrow E'$: given by $F(e) =$ the $\eta(e)$-view at e.

To prove that F is injective, suppose $e_1 \neq e_2$. If $\eta(e_1) = \eta(e_2) = i$, clearly $lh(e_1) \neq lh(e_2)$ and hence i-view at e_1 is distinct from the i-view at e_2. Otherwise, suppose $\eta(e_1) = i \neq j = \eta(e_2)$, but that the i-view at e_1 is identical to the j-view at e_2. This means that e_1 is the maximal i-event in $\downarrow e_2$, and that e_2 is the maximal j-event in $\downarrow e_1$. Then in particular, we get $e_1 \leq e_2, e_2 \leq e_1$ and $e_1 \neq e_2$, contradicting the antisymmetry of $\leq$. Thus F is injective.

Now consider $e' \in E', \eta'(e') = i$. Then e' is an i-event occurrence in $\mathcal{K}_\mathcal{E}$ say, $< \sigma_1, ..., \sigma_n >$. Since $|\sigma_i| > 1$, it is of the form $\sigma x x'$, where there exist $c, c' \in C, c \in x, c' \in x'$ and for some $e \in E_i, c' = c \oplus \{e\}$. It can then be easily checked that e' is indeed the i-view at e, that is $F(e) = e'$, proving surjectivity of F.

The fact that $\eta(e) = \eta'(F(e))$ is trivial. Now consider $e_1, e_2 \in E$ such that $e_1 \leq e_2$. Then every schedule δ_2 for $\downarrow e_2$ includes as a prefix a schedule δ_1 for $\downarrow e_1$. Since $\delta_1 \preceq$

$\delta_2, \epsilon(\delta_1) \leq' \epsilon(\delta_2)$, that is, $F(e) \leq' F(e')$, as required.

On the other hand, suppose $e_1, e_2 \in E$ such that $F(e_1) \leq' F(e_2)$. Again let $F(e_1)$ be an i-event occurrence in $\mathcal{K}_{\mathcal{E}}$ and $F(e_2)$ a j-event occurrence. Then if $F(e_2) =< \rho_1, ..., \rho_n >$, then ρ_j is of the form $\rho x x'$ where there exist $c \in x, c' \in x'$ such that $c' = c \oplus \{e_2\}$, and ρ_i is of the form $\rho' x_1 x_2 \rho'' x_3 x_4$, where there exist $c_l \in x_l, l \in \{1, ..4\}$ such that $c_2 = c_1 \oplus \{e_1\}, c_2 \subseteq c_3, c_4 = c_3 \oplus \{e_3\}$, and e_3 is the i-maximal event in $\downarrow e_2$. Thus we get $e_1 \leq e_3 \leq e_2$, and thus $e_1 \leq e_2$.

Thus F is order-preserving too, and we have the result. □

5 Back and Forth

Given a KTS $\mathcal{K}$, we can associate an ACSA $\mathcal{E}_{\mathcal{K}}$ with it, and with this ACSA we can associate a KTS $\mathcal{K}_{\mathcal{E}_{\mathcal{K}}}$. What is the relationship between $\mathcal{K}$, and $\mathcal{K}_{\mathcal{E}_{\mathcal{K}}}$?

Before we answer this question, we define a notion of simulation between transition systems, which is a kind of unfolding.

Def 9 : Let $\mathcal{K} = (S, \rightarrow, s_0, Eq_n)$ and $\mathcal{K}' = (S', \rightarrow', s'_0, Eq'_n)$ be KTSs. We say that TS' is a *simulation* of TS iff there exists a *surjective* map $f : S' \rightarrow S$: such that

(i) $f(s'_0) = s_0$,

(ii) for $s'_1, s'_2 \in S'$, if $s'_1 \rightarrow' s'_2$ then $f(s'_1) \rightarrow f(s'_2)$,

(iii) for $s_1, s_2 \in S$, if $s_1 \rightarrow s_2$ and $f(s'_1) = s_1$, then there exists $s'_2 \in S'$ such that $s'_1 \rightarrow' s'_2$ and $f(s'_2) = s_2$, and

(iv) for $s'_1, s'_2 \in S'$, if $s'_1 \approx'_i s'_2$ then $f(s'_1) \approx_i f(s'_2)$.

□

For the rest of this section, fix a KTS $\mathcal{K} = (S, \rightarrow, s_0, Eq_n)$, and $\mathcal{E}_{\mathcal{K}} = (E, \leq, \eta)$.

Before we proceed with the "back and forth" argument, we need some technical results.

Recall that R denotes the set of all runs of the KTS. Define the map $g : R \rightarrow 2^E$ by: $g(s_0) = \emptyset$, and for $\delta \in R^+, g(\delta) = \{\epsilon(\delta') | \delta' \preceq \delta, \delta' \in R^+\}$.

Proposition 10 : Suppose $\delta \in R$. Then $g(\delta) \in C$. Further,

(a) $|g(\delta)| = |\delta| - 1$, and

(b) if δ_1 is a prefix of δ_2, then $g(\delta_1) \subseteq g(\delta_2)$.

□

Proposition 11 : Suppose $g(\delta_1) = g(\delta_2)$, then there exists s such that δ_1 is of the form δs and δ_2 is of the form $\delta' s$.

$$\begin{tikzcd}
t_m = s_m \arrow[r, "i"'] \arrow[d, "j"'] & s_{m+1} \arrow[d, "j"] \\
t_{m+1} \arrow[r, "i"'] \arrow[d, "j_1"'] & u_{m+1} \arrow[d, "j_1"] \\
\vdots & \vdots \\
t_{p-1} \arrow[r, "i"'] \arrow[d, "j_r"'] & u_{p-1} \arrow[d, "j_r"] \\
t_p \arrow[r, "i"'] & t_{p+1} = u_p
\end{tikzcd}$$

Figure 9:

Proof : Suppose $g(\delta_1) = g(\delta_2) = c$. Then $|\delta_1| = |\delta_2| = k$, say. When $k = 1$, the result is obvious and the required $s = s_0$. So suppose $k > 1$. Let $\delta_1 = s_1 \stackrel{i_1}{\Rightarrow} s_2 \stackrel{i_2}{\Rightarrow} ...s_k$ and $\delta_2 = t_1 \stackrel{j_1}{\Rightarrow} t_2 \stackrel{j_2}{\Rightarrow} ...t_k$. Clearly, $s_1 = t_1 = s_0$, the initial state of the KTS.

Let m be the latest index of agreement between δ_1 and δ_2, i.e. for $1 \leq l \leq m, s_l = t_l$ and $s_{m+1} \neq t_{m+1}$. Since the first state of both runs is $s_0, m > 0$. Thus $0 \leq k - m < k$. We show the result by induction on $k - m$. The base case, when $m = k$ is trivial, since then the required $s = s_k = t_k$. Now assume by induction hypothesis that the result holds for runs with $m \leq k$. Now consider runs δ_1 and δ_2 such that $m < k$.

To fix notation, let $\delta = s_1 s_2 ... s_m (= t_1 t_2 ... t_m), c' = g(\delta), s_m \stackrel{i}{\Rightarrow} s_{m+1}, s_m \stackrel{j}{\Rightarrow} t_{m+1}$. Further let $c'_1 = g(\delta s_{m+1}) = c' \oplus \{e_1\}, c'_2 = g(\delta t_{m+1}) = c' \oplus \{e_2\}$. That such configurations exist is assured by the construction of g. Clearly, e_1 and e_2 are respectively i and j-event occurrences. Note that by monotonicity of g, $e_1 \in c'_1 \subseteq c, e_2 \in c'_2 \subseteq c$, thus $\{e_1, e_2\} \subseteq c$.

Now suppose $i = j$. If $e_1 = e_2$, then $[s_{m+1}]_i = [t_{m+1}]_i$, and we have $s_m \stackrel{i}{\Rightarrow} s_{m+1}, s_m \stackrel{i}{\Rightarrow} t_{m+1}$ and $s_{m+1} \approx_i t_{m+1}$, therefore $s_{m+1} = t_{m+1}$, contradicting the fact that the latest index of agreement is $m < k$. Therefore $e_1 \neq e_2$. But then e_1 and e_2 are in immediate conflict in E, contradicting the fact that $\{e_1, e_2\} \subseteq c$.

Thus we find that $i \neq j$. Let p be the smallest index such that $t_p \stackrel{i}{\Rightarrow} t_{p+1}$. Clearly, $m < p < k$. Now for each $l : m < l \leq p + 1$, let $g(\delta t_{m+1} ... t_l) = c_l$ and let $c_{l+1} = c_l \oplus \{e_l\}$. By definition of KTSs, we can find a sequence of states (see Figure 9) $u_{m+1} ... u_p$ such that for every $l : m < l \leq p, t_l \stackrel{i}{\Rightarrow} u_l$. Further $s_{m+1} \stackrel{j}{\Rightarrow} u_{m+1}$ and for all $m < l < p, u_l \stackrel{j'}{\Rightarrow} u_{l+1}$ iff $t_l \stackrel{j'}{\Rightarrow} u_{l+1}$. Now, for $m < l \leq p$, let $d_l = g(\delta s_{m+1} u_{m+1} ... u_l)$. From the fact that $s_m \approx_i t_{m+1} \approx_i ... \approx_i t_p$ and $s_{m+1} \approx_i u_{m+1} \approx_i ... \approx_i u_p$, we get, for all $m < l < p, d_l = c_l \oplus \{e_1\}$. Now let $e' = c_{p+1} - c_p$. Thus we have two i-event occurrences e_1, e' enabled at c_p. If $e_1 \neq e'$, they are in (immediate) conflict, and hence $e_1 \notin d$, for every configuration $d : c_{p+1} \subseteq d$. But by monotonicity of g, we have $c_{p+1} \subseteq g(\delta_2) = c$ and $e_1 \in c$, a contradiction. Therefore $e_1 = e'$ implying $u_p \approx_i t_{p+1}$ which is possible only if $u_p = t_{p+1}$.

Now let $\delta_2' = \delta_1 s_{m+1} u_{m+1} ... u_p t_{p+2} ... t_k$. Since $g(\delta_1 s_{m+1} u_{m+1} ... u_p) = g(\delta t_{m+1} ... t_p u_p)$, we also have $g(\delta_2) = g(\delta_2') = c$. But now consider the two runs δ_1, δ_2': their g-image is the same, and their latest index of agreement is $m + 1$; but then by induction hypothesis, $s_k = t_k$, which is what we set out to prove. □

Corollary 1 : Suppose $c \in C$. Then there exists $\delta \in R$ such that $g(\delta) = c$.

Proof : If $c = \emptyset$, then $g(s_0) = c$. Inductively assume that for every configuration c such that $|c| < k$, there exists δ such that $g(\delta) = c$. Now suppose $c \in C, |c| = k, k > 0$. There exists a configuration c' and $e \in c$ such that $c = c' \oplus \{e\}$. By inductive assumption, there exists δ' such that $g(\delta') = c'$. Let $e = \epsilon(\delta), \delta = \delta_1 s s'$. But then $g(\delta_1 s) = c'$, hence by the previous proposition, δ' is of the form δs. Thus $g(\delta s s') = c$, as required. □

Theorem 12 : Let $\mathcal{K}$ be a KTS. Then $\mathcal{K}_{\mathcal{E}_\mathcal{K}}$ is a simulation of $\mathcal{K}$.

Proof : Let $\mathcal{K} = (S, \rightarrow, s_0, Eq_n)$, $\mathcal{E}_\mathcal{K} = (E, \leq, \eta)$ and $\mathcal{K}_{\mathcal{E}_\mathcal{K}} = (S', \rightarrow', s_0', Eq_n')$. Clearly S' is the set C of finite configurations of $\mathcal{E}$.

Define $f : C \rightarrow S$ by: $f(\emptyset) = s_0$, and for all nonempty configurations $c, f(c) = s$, where there exists a path $\delta s \in R^+$ such that $g(\delta s) = c$. The previous proposition and corollary together ensure that f is indeed well-defined.

Now if $c' = c \oplus \{e\}, f(c) = s, f(c') = s'$, then by an argument similar to the one above, we can check that $e = \epsilon(\delta s s')$, and hence $s \rightarrow s'$. On the other hand, if $s \rightarrow s'$ in K, there is a run $\delta s s'$ such that $g(\delta s s') = g(\delta s) \oplus \{\epsilon(\delta s s')\}$. That is, when $f(c) = s$, there exists $c', f(c') = s'$ such that $c \rightarrow' c'$.

Suppose $c \approx_i' c'$, that is $c \cap E_i = c' \cap E_i$. Let $g(\delta s) = c, g(\delta' s') = c'$. We find that $\phi_i(\delta s) = \phi_i(\delta' s')$, and hence $s \approx_i s'$, as required. □

6 Discussion

It is interesting to explore the implications of condition (i) in the definition of knowledge transition systems. Condition (i) asserts that every state is a knowledge state and hence any change in state is due to the fact that the knowledge of some agent (in fact, a unique one) has changed. In this view, there is no "state of the real world", but only what is got by pooling the information available to all the agents.

One technical reason for such a condition is the following: we wish to see every state change as an event in the system, and if the new state is indistinguishable from the old one for every agent in the system, this means an event occurrence unobservable to all agents. In such a situation, we have taken the attitude that events outside the system might as well not exist. However, we could indeed drop this stringent condition and work with external events; this means a similar change in the definition of n-ACSAs: the funtionality of the naming function η is from E to $\{1, 2, ..., n\} \cup \{*\}$, where $\eta(e) = *$ can be used to denote that e is an external event.

The other aspect is that of uniqueness: why should a state change not be accompanied by knowledge change for several agents ? In the previous section, we have utilized uniqueness principally for convenience in extracting *asynchronously* communicating sequential agents, where every internal event occurrence is associated with only one agent. We can in fact, generalize these structures to allow *shared* events, as proposed in [LRT]:

Def 13 : A system of n Communicating Sequential Agents (n-CSA) is a triple C=$(E, \leq, \eta)$ where

(i) E is a set of *event occurrences*,

(ii) $\leq \subseteq E \times E$ is a partial order called the *causality relation*, and

(iii) $\eta : E \rightarrow 2^{\{1,...,n\}}$ is a *naming function* such that $\forall e \forall i \in \{1, ..., n\}$ $\{e' | e' \leq e\} \cap \{e' | i \in \eta(e')\}$ is totally ordered by $\leq$.

$\square$

This gives us a considerable generality to discuss systems with both the possibilities of synchronous as well as asynchronous communication.

With a view to relating KTS with n-CSAs, we can drop the uniqueness conditions. However, the confluence and other conditions become more difficult to express now. We need to generalize the relation $\stackrel{i}{\Rightarrow}$ to $\stackrel{u}{\Rightarrow}$, where $u \subseteq \{1, 2, ..., n\}$, with the interpretation that $s \stackrel{u}{\Rightarrow} s'$ means that for all $i \in u$, $s \not\approx_i s'$, and for all $j \notin u$, $s \approx_j s'$ (along with the condition $s \rightarrow s'$). Then the confluence condition is specified for $s \stackrel{u}{\Rightarrow} s_1, s \stackrel{v}{\Rightarrow} s_2$, where $u \cap v = \emptyset$. On the other hand, the local choice condition is now for $s \stackrel{u}{\Rightarrow} s_1, s \stackrel{v}{\Rightarrow} s_2$, where $u \cap v \neq \emptyset$.

With such modifications on both sides we believe that (the generalized) KTSs can be related with n-CSAs as before. However, the system can now have many "major" state transitions which in a sense 'bypass' many paths in the system, and this creates many technical difficulties. We believe that a result similar to the one in the previous section can be proved also for this class.

Another question relates to the fixed number n of agents in the system - is that crucial? It can be easily checked that the results in the previous section can be easily extended to systems with countably many agents. However, when we allow shared events as proposed above, we need to ensure that η maps events only to finite sets of agents, and we need conditions to ensure that every state change in the KTS causes knowledge change for only finitely many agents.

In a different direction, we can relate knowledge transition systems to "communicating" transition systems. We can think of these as automata making local transitions, except that there can be dependencies between transitions across systems. Every KTS can be easily "decomposed" into such automata, and we can associate n-ACSAs with the automata so that a simulation of the original KTS can be recovered.

Def 14 : A *system of n communicating transition systems* is an $(n+1)$- tuple CA, $CA = (TS_1, ..., TS_n, Comm)$, where for each i in $\{1, 2, ..., n\}$, $TS_i = (X_i, \rightarrow_i)$ is a local transition system with local states X_i and local transition system $\rightarrow_i \subseteq X_i \times X_i$. $Comm$ is the communication constraint, $Comm \subseteq (X \times X) \times (X \times X)$, where $X = \cup_i X_i$ and $(x_1, x_2) Comm (x_3, x_4)$ implies that for some $i \neq j$, $x_1 \rightarrow_i x_2$, and $x_3 \rightarrow_j x_4$.

Now suppose we are given a KTS $(S, \rightarrow, s_0, Eq_n)$, we can associate a CA with it as follows. Define:

$X_i = \{x | x$ is an equivalence class under $\approx_i\}$,

$x \rightarrow_i y$ iff there exist $s \in x$, $s' \in y$ such that $s \stackrel{i}{\Rightarrow} s'$, and

$(x, y)Comm(x', y')$ iff there exist $s_0 \in x$, $s_1 \in y \cap x'$ and $s_2 \in y'$ such that $s_0 \stackrel{i}{\Rightarrow} s_1 \stackrel{j}{\Rightarrow} s_2$, and there is no $s_3 \in S$ such that $s_0 \stackrel{j}{\Rightarrow} s_3 \stackrel{i}{\Rightarrow} s_2$, where x, y are i-equivalence classes and x', y' are j-equivalence classes, $i \neq j$.

$\square$

The unfolding of CA to obtain n-ACSA's is straightforward. In this manner, we can relate CA and KTSs.

7 Conclusion

We have argued that the notion of partially ordered time in distributed systems is in a sense dual to the notion of knowledge change of agents in the system. We have attempted to formalize this (widely-believed) result. Further, while it is customary to assert that "knowledge is not lost by receiving messages" and that "knowledge is not gained by sending messages", here we utilize such information to conclude whether an event occurrence constitutes an internal event, sending a message or receiving a message.

This setting seems to be appropriate for category-theoretical analysis. Note that in knowledge transition systems, each state has an S5-Kripke structure in it, and transitions refine the information partitions and thus modify the structures. Further, we hope to place relationships between KTSs and ACSAs on firmer foundations by studying categories of knowledge transition systems and categories of ACSAs. We would like to study the category of KTSs (with KTSs as objects and simulations as morphisms) on one hand and the category of ACSAs (with label- preserving order-preserving maps as morphisms) on the other: the question we study in this paper then becomes the following – given the functor from the category of ACSAs to the category of KTSs, obtain the left adjoint of that functor. Our results about isomorphism in Section 4 and simulation in Section 5 then constitute a co-reflection.

We would like to study knowledge transition systems using a propositional modal logic which contains the usual S5 modalities K_i for $i \in \{1, ..., n\}$, as well as the temporal modalities $\bigcirc$ and $\Diamond$. Such a logic is easily defined, and seen to be decidable too. However, obtaining a complete axiomatization seems to be a nontrivial problem.

Another interesting question relates to viewing KTSs as automata. In the last section we mentioned communicating transition systems, but to make sense of these as automata, we need to introduce action-labelled transitions. The alphabet of actions can then be partitioned n-ways. We can then consider finite state KTSs as acceptors of regular languages with some notion of communication dependency (in the same way as Zielonka automata [Zie] accept the regular trace languages of Mazurkiewicz [Maz]). We thus have hopes of KTSs providing a transition system model of message passing.

References

[CM] Chandy, M., and Misra, J., "How processes learn", *Distributed Computing*, vol. 1, #1, pp. 40-52.

[DM] Dwork, C., and Moses, Y., "Knowledge and common knowledge in a byzantine environment: Crash failures", Inf and Comp, vol 88 (1990) pp 156-186.

[HF] Halpern, J., and Fagin, R., "Modelling knowledge and action in distributed systems", *Distributed Computing*, vol 3, #4, 1989, 159-177.

[HM] Halpern, J., and Moses, Y., "Knowledge and Common Knowledge in a Distributed Environment", *JACM*, vol. 37, pp. 549-578.

[HZ] Halpern, J., and Zuck, L., "A little knowledge goes a long way: simple knowledge-based derivations and correctness proofs for a family of protocols", *Proc. 6th ACM Symp. on Principles of Distributed Computing*, 1987, pp. 269-280.

[Lam] Lamport, L., "Time, Clocks and the Ordering of Events in a Distributed System", *CACM* , vol. 21, #7, July 1978, pp. 558-565.

[LRT] Lodaya, K., and Ramanujam, R., and Thiagarajan, "Temporal Logics for Communicating Sequential Agents: I", Intl Jnl on Found. of Comp Sci, vol 3, #2, 1992, 117-159.

[Ma] Mazer, M., "A link between knowledge and communication in faulty distributed systems (Preliminary Report)", *TARK III*, 1990, pp. 289-304.

[Maz] Mazurkiewicz, A., "Basic notions of trace theory", LNCS 354 (1989), pp 285-363.

[NPW] Nielsen, M., Plotkin, G., and Winskel, G., "Petri Nets, Event Structures and Domains, Part I", *Theoretical Computer Science*, vol 13, #1, 1980, 86-108.

[PK] Parikh, R., and Krasucki, P., "Levels of knowledge in distributed systems", *Sadhana*, vol. 17, Part. 1, March 1992, pp. 167-191.

[PR] Parikh, R., and Ramanujam, R., "Distributed Processes and the Logic of Knowledge", *Logics of Programs* Springer LNCS #193, pp 256-268.

[Pra] Pratt, V., "The duality of time and information", Proceedings of CONCUR 92, Springer LNCS 630, pp 237-253.

[Zie] Zielonka, W., "Notes on finite asynchronous automata", RAIRO-Inf. Theor. et Appli., vol 21 (1987) pp 99-135.