Toward Distributed Declarative Control of Networked Cyber-Physical Systems (Full Version)

Mark-Oliver Stehr, Minyoung Kim, and Carolyn Talcott

SRI International,
`stehr,mkim,clt@csl.sri.com`

Abstract. Networked Cyber-Physical Systems (NCPS) present many challenges that are not suitably addressed by existing distributed computing paradigms. They must be reactive and maintain an overall situation awareness that emerges from partial distributed knowledge. They must achieve system goals through local, asynchronous actions, using (distributed) control loops through which the environment provides essential feedback. Typical NCPS are open, dynamic, and heterogeneous in many dimensions, and often need to be rapidly instantiated and deployed for a given mission.
To address these challenges, we pursue a declarative approach to provide an abstraction from the high complexity of NCPS and avoid error-prone and time-consuming low-level programming. A longer-term goal is to develop a distributed computational and logical foundation that supports a wide spectrum of system operation between autonomy and cooperation to adapt to resource constraints, in particular to limitations of computational, energy, and networking resources. Here, we present first steps toward a logical framework for NCPS that combines distributed reasoning and asynchronous control in space and time. The logical framework is based on partially ordered knowledge sharing, a distributed computing paradigm for loosely coupled systems that does not require continuous network connectivity. We present general theoretical results and also illustrate our approach with a simulation prototype of our logical framework in the context of networked mobile robot teams that operate in an abstract instrumented cyber-physical space with sensors.

1 Introduction

A growing number and variety of devices can sense and affect their environment. Some are fairly simple, such as radio-frequency identification (RFID) access control, some quite sophisticated, such as mobile robots with localization and sensing capabilities. This opens up an opportunity for a new generation of *Networked Cyber-Physical Systems* (NCPS). Such systems can provide complex, situation-aware, and often critical services in applications such as distributed surveillance, crisis response, medical systems, self-assembling structures or systems, networked space/satellite missions, or distributed critical infrastructure monitoring and control. General principles and tools are urgently needed for building robust, effective NCPS using individual cyber-physical devices as building blocks. In this paper, we present a declarative approach to NCPS based on a logical framework that supports distributed reasoning and can interact with the physical world asynchronously through observations and goal-oriented control.

Cyber-physical systems are often deployed in challenging environments with a wide spectrum of networking characteristics. They should be able to take advantage of opportunities for communication as they arise and must be robust in spite of delays and disruptions due to, for example, mobility, failures, or nodes entering and leaving the system. Topology changes (network partitioning in the extreme case) can happen continuously, so that even the common assumption of globally stable periods, which is crucial for many distributed algorithms, is not realistic. Hence, the logical framework is developed on top of a loosely coupled distributed computing model based on the *partially ordered knowledge-sharing* paradigm that has been used in our earlier work on disruption-tolerant networking [20]. Knowledge (as opposed to, say, a packet) is a semantically meaningful unit of information that can be stored, processed, aggregated, and communicated to other nodes. A unique feature of the knowledge-sharing model is that it is parameterized by an application-dependent partial order on knowledge that is available to all nodes and provides an abstraction of the knowledge semantics. In this model, the network topology can change continuously, communication can be unreliable, and no bounds are assumed on communication delays. As a consequence, any form of communication between nodes is acceptable. Similar to data mules for sensor networks [7] or message ferrying in delay- and disruption-tolerant networks [9], each node can cache knowledge for extended periods of time and hence can exploit the (possibly mobile) network dynamics to share knowledge without relying on an end-to-end path at any point in time.

The declarative view of NCPS enables us to recast information collection, control, and decision problems as logical problems that are primarily centered around the duality of two kinds of knowledge: facts and goals. Facts can represent sensor readings at specific locations and other information that is derived by possibly distributed computation. Goals can represent queries for information or requests to the system or individual components to perform certain actions. Although there are cases where a goal can be directly satisfied by a single action, it is more often the case that distributed actions are needed, sometimes in a coordinated manner that requires cooperation across multiple nodes. In our logical framework, both facts and goals will be treated on an equal footing together with corresponding communication and reasoning rules. The distributed, dynamic, and interactive nature of the underlying systems is rarely considered in logical frameworks, which are traditionally designed as closed systems. Our framework is flexible enough to take into account the heterogeneous resources and capabilities at each node. Each node cooperates with its neighbors and interacts with its environment by sensing and affecting, driven by facts and goals. Overall system goals are refined to goals achievable by an individual node or device. Reasoning takes place locally at each node in the network as well as in cooperation (and competition) with other nodes. Seamless integration of cooperation and autonomy ensures that there is no need to rely on the existence or connectivity of other nodes, so that the local operation can always proceed, although possibly in a less optimal fashion. Solutions can be shared and composed oportunistically without being subject to any rigid or hierarchical flow constraints.

Recent applications of cognitive and more specifically declarative techniques in communication and networking, [5] being a noteworthy example, have attracted a lot of interest, but a declarative treatment of NCPS still remains a challenge. To study this problem we use *self-organizing mobile robots* as an example capturing many of the challenges of NCPS [7]. This example is inspired by previous work at SRI, in particular the Centibots project [17] that has developed a team-based hierarchical planning approach to accomplish a mission such as collaborative mapping, and the Commbots project [10], where the mission objective is to improve network connectivity by distributed control of robotic routers.

This paper contains the following contributions. In Section 2, we informally introduce the knowledge-based distributed computing model with a partial-order based knowlege semantics that is slightly more general than what we used in previous work. More formally, we develop in Section 3 our distributed logical framework for NCPS together with its key properties, such as soundness, completeness, and termination. Section 4 ilustrates the key ideas and a prototypical implementation by means of an abstract simulation of a networked mobile robot team operating in an instrumented cyber-physical space.

2 Knowledge Sharing as a Basis for Distributed Computing

In this paper we use a refinement of a distributed computing model based on asynchronous *knowledge sharing* that we have used in earlier work [20] as the basis for disruption-tolerant networking. The knowledge-sharing model can make explicit the structure of a distributed computation in space and time, and hence is less abstract than many other models of distributed computing, e.g. those abstracting from the network topology by assuming direct end-to-end channels.

In a nutshell, we assume a networked cyber-physical system with a finite set of so called *cyber-nodes* that provide *computing* resources, can have volatile and/or persistent *storage*, and are all equipped with *networking* capabilities. Cybernodes can have additional devices such as *sensors* and *actuators*, through which they can observe and control their environment, but only to a limited degree (including possibly their own physical state, e.g. their orientation/position). Cyber-nodes can be fixed or mobile, and for the general model no assumption is made about the computing or storage resources or about the network and the communication capabilities or opportunities that it provides. Hence this model covers a broad range of *heterogeneous* technologies (e.g. wireless/wired, unicast/broadcast) and potentially challenging environment conditions, where networking characteristics can range from high-quality persistent connectivity to intermittent/episodic connectivity. The cyber-physical system is *open* in the sense that new nodes can join and leave the network at any time. Permanent or temporary communication or node *failures* are admitted by this model. As a consequence, many forms of network dynamics including partitioning, merging, message ferrying, group mobility, etc. are possible.

In the following, we give in informal characterization of an individual cyber-node that will be sufficient for the our purposes. Each cyber-node has a unique *name* and a *local clock*, which increases monotonically by at least one unit in each instruction and is only loosely synchronized with other nodes in the network if admitted by the networking conditions. We also assume that each node has access to a source of randomness (e.g. a fair coin), with the idea that typical applications of this model make heavy use of randomization techniques.

Locally, each cyber-node uses a sequential and universal computation model. The model is based the dual notions of local events and distributed knowledge. Two key services are provided by each node. First, timed events can be posted, i.e. scheduled to be executed at any local time (possibly randomized) in the future. Second, knowledge can be posted, i.e. submitted for dissemination in the network. All local computation is event-based, where corresponding to the two services above, events can be either *timed events* or *knowledge events*, with the latter representing the reception of a new unit of knowledge. Similar to existing middleware framworks for messaging or group communication, knowledge dissemination can take place independently in different logical *cyber-spaces*, but a unit of knowledge is a more state-like entity that should not be confused with the notion of a message. Furthermore, no reliability, delivery order, or atomicity guarantees are provided to the applications, because they would serverely limit the scalability of the model in terms of the network size.

Distributed knowledge sharing is asynchronous and each node can use some of its storage as a cache, which we also refer to as a *knowledge base*. Network caching allows the system to support communication even if no end-to-end path exists at a single point in time. Different from a shared-memory model, distributed knowledge sharing allows each node to have its own (typically partial and delayed) view of the distributed state of knowledge. Different from an asynchronous message-passing model, knowledge is not directed towards a particular destination. Instead each node decides based on the knowledge content (or its embedded type) if it wants to use the unit of knowledge that it receives.

Epidemic and (spatial) gossiping techniques can be used to implement knowledge sharing, but unlike gossiping, which is based on the exchange of cache summaries, knowledge sharing can also be implemented by single-message protocols based on unidirectional communication [20]. On the other hand, epidemic computing covers a very broad class of algorithms, whereas distributed knowledge sharing is a more restricted model that makes specific use of the abstract semantics of knowledge that is given in a very specific way, namely in terms of an equivalence relation and a partial order. The consideration of the partial-order semantics of knowledge by intermediate nodes is of key importance for scalable implementations and also the reason why knowledge sharing is fundamentally different from asynchronous/unreliable or even epidemic/probabilistic broadcast.

To partially capture the semantics of knowledge for the purpose of distributed knowledge sharing, we assume an application-specific partial order $\leq$ on all knowledge items together with its induced equivalence relation. We refer to $\leq$ as the *subsumption order* given that the intuitive meaning of $K \leq K'$ is that K' contains at least the information contained in K. With this interpretation the induced equivalence $K \equiv K'$, defined as $K \leq K'$ and $K \geq K'$, means that K and K' have the same semantics, even if they are represented in different ways. In this situation, the knowledge-sharing model may (but does not have to) discard K' without delivering it to the application, if K has already been delivered already. In addition to $\leq$, we assume an application-specific strict partial order $\prec$ that is compatible with $\leq$ and we refer to as *replacement order*, with the intuition that $K \prec K'$ means that K' replaces/overwrites K, and hence if K has not been delivered yet to the application, the knowledge-sharing model may (but does not have to) discard it, if K' has already been received.

The distributed knowledge-sharing model can be specialized by imposing local and global resource bounds as well by more specific environment (and hence network) models. This paper, however, will not impose such restrictions, because the logical framework that we are presenting here should be applicable to a wide variety of cyber-physical systems.

3 Distributed Logic and Cyber-Inference

Declarative control aims at providing the user with a logical view of the cyber-physical system so that user objectives can be conveyed to the system, which then will make its best effort to realize those objectives. Such objectives are given in the context of the current system state (which is only approximately and partially observable). Furthermore, the user objectives can be part of a larger set of objectives (e.g., including system policies and objectives from other users), which we simply refer to as the system goal. Declarative control is the process of continuous adaptation of the system to transition to a state that satisfies the system goal, which in turn can continue to change based on feedback from the environment.

The purpose of logic in this context is many-fold. First of all, it provides a language to express and communicate system goals. Dually, it allows expressing and communicating facts about the current system state. In both cases, communication includes communication with the users but also communication among the components of the system themselves. At the level of an individual cyber-physical component, the logic provides a declarative interface for goal-oriented control and feedback through observations that are represented as logical facts. Finally, it provides a framework for inference and computation, which allows facts and goals to interact with each other and form new facts or goals.

Aiming at a solution to declarative control that covers the entire spectrum between cooperation and autonomy and makes opportunistic use of networking resources, it is clear that the logic needs to be inherently distributed. The opportunity to exchange knowledge with other nodes should lead to cooperation, and the absence of such opportunities should lead to more autonomous behavior. In the following we present a simple distributed inference system that accomplishes this goal. We use Horn clause logic [18] to illustrate our approach, which we expect to generalize to more expressive logics.

For the following abstract logical treatment, we assume a networked cyber-physical system S with a finite set of cyber-nodes N. We assume a time-dependent network and environment model, in which two cyber-nodes have the capability to communicate (uni- or bidirectionally) whenever the network conditions admit it and where each cyber-node can have (not necessarily the same) sensors and actuators. The sensors can generate observations at arbitrary time points. The actuators are driven by goals, which they can either attempt to satisfy immediately or in a continuous asynchronous process with (partial) feedback provided through observations. Instead of imposing restrictive conditions on S we generally allow S to operate under arbitrary conditions. As a consequence, a guarantee that goals are achieved is not assumed in this model. In the following, we use x and y to range over cyber-nodes and t to range over the time domain, for which we use natural numbers in this paper.

For the following, we also assume a fixed signature Σ and a fixed finite theory Ω over Σ in Horn clause logic that is shared by all nodes of the cyber-physical system. We assume that Σ contains built-in constants for natural numbers and names of cyber-nodes. Additional *built-in functions*, and *built-in predicates* can be included in Σ. Nonpropositional and propositional constants are identified with corresponding functions and predicates, respectively, of arity zero. To account for the temporal and distributed character of cyber-physical systems, we assume that the signature Σ contains a distinguished set of predicates (distinct from built-ins) that we refer to as *cyber-predicates*, i.e., predicates that define the interface of the logic with the outside world (i.e., cyber-physical devices and users), and we use p_c to range over such predicates in the following.

Using standard terminology, we assume an countably infinite set of variables $\mathcal{V}$, a set of *terms* $\mathcal{T}(\Sigma, \mathcal{V})$, and a set of *atoms* $\mathcal{A}(\Sigma, \mathcal{V})$, i.e., atomic propositions, over these variables. Using e to range over terms, atoms are of the form $p(e_1, \ldots, e_n)$ if p is an n-ary predicate. As usual, we define ground terms $\mathcal{T}(\Sigma)$ and ground atoms $\mathcal{A}(\Sigma)$ as terms and atoms without variables, respectively. In the following, we use P and Q to range over atoms. We furthermore assume that all clauses in Ω are uniquely labeled and definite, i.e., of the form $l : P_1, \ldots, P_n \Rightarrow Q$ with a unique label l, where Q is not the application of a built-in predicate. As usual, an implicit universal quantification over all variables is assumed. We use $\vdash$ to denote the *standard derivability* in Horn clause logic with all the built-ins in Σ, i.e., if Φ is a set atoms, $\Phi \vdash Q$ means that Q can be derived from Φ by means of the clauses in Ω. If in addition, Q is generated by a forward or backward clause we write $\Phi \vdash_{\mathrm{f}} Q$ or $\Phi \vdash_{\mathrm{b}} Q$, respectively.

For our proof system, we assume that $\Omega = \Omega_\mathrm{f} \cup \Omega_\mathrm{b}$, where Ω_f and Ω_b are sets of clauses that we refer to as *forward and backward clauses*, respectively. The set of *facts* is simply defined as the set of all ground atoms. Hence, all predicates are allowed to occur in facts. The set of *goal predicates* is any set of predicates that includes at least the built-in predicates and all predicates that apear in the conclusion of a backward clause, i.e., a clause from Ω_b. Certain cyber-predicates can be included in this set if they are intended to form goals. The set of *goals* can be any set of (not necessarily ground) atoms that are applications of goal predicates and satisfy the following closure properties: **(1)** If G is a goal then $\sigma(G)$ is a goal, i.e., we have closure under (not necessarily ground) substitutions. **(2)** If there is a clause of the form $l : P_1, \ldots, P_n \Rightarrow Q$ in Ω_f, $j \in 1, \ldots, n$, P_j is the application of a goal predicate, $\sigma(P_i)$ is a fact for each $i \in 1, \ldots, j-1$, then $\sigma(P_j)$ is a goal. **(3)** If there is a clause of the form $l : P_1, \ldots, P_n \Rightarrow Q$ in Ω_b, P_j is the application of a goal predicate, $\sigma(P_i)$ is a fact for each $i \in 1, \ldots, j-1$, and $\sigma(Q)$ is a goal, then $\sigma(P_j)$ is a goal.

With this user-defined notion of goals, we now formulate our *variable restriction* that we generally assume in the following: **(1)** For each forward clause $l : P_1, \ldots, P_n \Rightarrow Q$ in Ω_f, each variable in Q appears in at least one of the atoms $P_1, \ldots, P_n$. **(2)** For each backward clause $l : P_1, \ldots, P_n \Rightarrow Q$ in Ω_b, if $\sigma(Q)$ is a goal, then each variable in $\sigma(Q)$ appears in at least one of the atoms $\sigma(P_1), \ldots, \sigma(P_n)$. Generally, the idea is that the set of goals is defined for a given application such that this variable restriction is satisfied.

Note that we neither require that every instance of the conclusion of a backward clause is a goal, nor do we exclude the possibility that a goal appears as an instance of the conclusion of a forward clause. From now on, we will use F and G to range over facts and goals, respectively. It is important that we do not require that every atom (in a clause) is a goal, because some atoms may only become sufficiently constrained after some of their variables have been instantiated. This remark particularly applies to built-in goals defined next.

A fact or goal that is the application of a built-in predicate is called a *built-in fact* or *built-in goal*, respectively. Given a built-in goal G, we say that $\sigma(G)$ is a *solution* of G iff $\sigma(G)$ is ground (hence becomes a fact) and $\vdash \sigma(G)$. We generally assume that all built-in goals G are finitary, where a goal G is said to be *finitary* iff it has a finite set of solutions. Note that this condition implies that $\vdash G$ can only hold for a built-in goal G if G is also a fact. All cyber-predicates p_c are explicitly time dependent and form atoms $p_c(t, \ldots)$, where t is a natural number denoting its timestamp, i.e., its time of creation at the creating node. A *cyber-fact* or *cyber-goal* is any fact or goal, respectively, of this form. Note that this does not preclude cyber-facts and cyber-goals from having additional temporal attributes or constraints that can be specified in the remaining arguments, e.g., the time of obervation in the case of a cyber-fact, or a deadline in the case of a cyber-goal.

Next we assume that the set of atoms is equipped with two reflexive *subsumption relations* $\leq_\mathrm{ff}$ and $\leq_\mathrm{gg}$, and four irreflexive *replacement relations* $\prec_\mathrm{ff}$, $\prec_\mathrm{gg}$, $\prec_\mathrm{gf}$, $\prec_\mathrm{fg}$, where the subscripts f and g indicate if the corresponding side is restricted to facts or goals, respectively. We assume that subsumption and replacements relations do not relate built-in atoms and we assume that they are linear and closed under renaming of variables. Here, we say that a relation R is *linear* iff $P\ R\ P'$ implies that each variable occurs at most once in (P, P'), and we say that R is *closed under renaming of variables* iff $P\ R\ P'$ implies $\sigma(P)\ R\ \sigma(P')$ for all variable renamings σ. Note that this implies that the name of a variable is irrelevant on both sides of these relations. The replacement relations cannot be arbitrary, and at a minimum must be extensible to a strict partial order, a condition formulated below, when the replacement ordering is introduced.

The logical state of a cyber-node is of the form $\Gamma \vdash \Delta\ @\ t, x$, where x is the unique name of the node, t is a natural number representing its local time, Γ is a finite set of derived facts, and Δ is a finite set of derived goals, where derived facts and goals are distinct objects with underlying facts and goals (see below) and explicit information about how they are derived. With an appropriate initial state, the logical state will satisfy the invariant that t is larger than any timestamp (but not necessarily any time) that is contained in Γ and Δ, meaning that t is always fresh and can be used as a timestamp for new atoms. It is worthwhile to point out that the interpretation of $\Gamma \vdash \Delta\ @\ t, x$ is nonstandard, and quite different from sequent calculus [3] even without t, x. Intuitively, Γ is a set of facts that is continuously growing through observations or by inference, while Δ is a set of goals that the system is trying to satisfy without necessarily aiming to satisfy all of them. To reflect the reality of cyber-physical systems, inference is a continuous typically nonterminating process, and goals and facts can change at any time due to environment and user interaction. Since reasoning is a distributed process in space and time and the world can change during this process, an approach quite different from traditional work in formal specification, logic programming, or automated deduction is needed.

Derived atoms, i.e., derived facts or goals, are represented as *knowledge* in the underlying knowledge-sharing model. Derived facts and derived goals are defined subsequently as objects of the form $f : F$ and $g : G$, respectively, where f and g are refered to as fact derivations and goal derivations, respectively, that will be used to keep track of how F and G were constructed. In the following, we use f and g to range over fact and goal derivations, respectively, and d to range over derivations of both kinds. Recall that F is always a ground atom, but a goal G can contain variables, which are implicitly existentially quantified. Different from standard approaches with explicit proof objects, we do not only define explicit derivations for facts but also for goals, which may turn out not to be solvable. We will also see that a goal does not have to be solved to be useful, but as a cyber-goal can still have an impact on the environment, which may be observable through cyber-facts.

A *derived atom* can be either a derived fact or a derived goal, where the set of *(atomic) derived facts* and *(atomic) derived goals* together with their *derivations* is defined by the following mutually inductive definition: **(1)** $\mathtt{B}_\sigma(G) : \sigma(G)$ is a (atomic) derived fact for each built-in goal G with a solution $\sigma(G)$, **(2)** $\mathtt{O}(F) : F$ is a (atomic) derived fact, also called an *observation*, for each cyber-fact F, **(3)**

$\mathtt{C}(G) : G$ is a (atomic) derived goal, also called a *control*, for each cyber-goal G; **(4)** $l_\sigma(f_1, \ldots, f_n) : \sigma(Q)$ is a derived fact, if there is a clause of the form $l : P_1, \ldots, P_n \Rightarrow Q$ in Ω_f, $\sigma(Q)$ is a fact, and $f_i : \sigma(P_i)$ are derived facts; **(5)** $l_\sigma^{-1}(f_1, \ldots, f_{j-1}) : \sigma(P_j)$ is a derived goal, if $j \in 1, \ldots, n$ and there is a clause of the form $l : P_1, \ldots, P_n \Rightarrow Q$ in Ω_f with goal $\sigma(P_j)$ and $f_i : \sigma(P_i)$ are derived facts; **(6)** $l_\sigma(f_1, \ldots, f_n; g') : \sigma(Q)$ is a derived fact, if there is a clause of the form $l : P_1, \ldots, P_n \Rightarrow Q$ in Ω_b, $\sigma(Q)$ is a fact, $f_i : \sigma(P_i)$ are derived facts, and $g' : G'$ is a derived goal with $\sigma(G') = \sigma(Q)$; and **(7)** $l_\sigma^{-1}(f_1, \ldots, f_{j-1}; g') : \sigma(P_j)$ is a derived goal, if $j \in 1, \ldots, n$ and there is a clause of the form $l : P_1, \ldots, P_n \Rightarrow Q$ in Ω_b with goal $\sigma(P_j)$ and $f_i : \sigma(P_i)$ are derived facts and $g' : G'$ is a derived goal with $\sigma(G') = \sigma(Q)$. In this definition and in the following, we identify derived goals that are related by consistent renaming of their variables (recall that they are implicitly bound). Different from facts and goals, derived facts and goals are disjoint sets. Given a derived atom $d : P$, it is easy to see from the above definition that P is uniquely determined by d. Hence, d uniquely determines if $d : P$ is a derived fact or a derived goal, respectively, something that is not determined by P alone.

We say that $d : P$ is an *immediate subderivation* of $d' : P'$, written $d : P \rhd d' : P'$, iff d' is of the form $L(\ldots, d, \ldots)$, where L any of the above constructors of derivations. We also say that $d : P$ is a *proper subderivation* or a *subderivation* of $d' : P'$, written $d : P \rhd^+ d' : P'$ or $d : P \rhd^* d' : P'$, respectively, relations that are defined as the transitive closure or the reflexive, transitive closure, respectively, of $\rhd$. Using K to range over derived atoms (i.e., units of knowledge) and $\mathcal{K}$ to range over sets of derived atoms, we inductively define the *knowledge derivation* relation $\vdash$ by the following conditions: **(1)** $\mathcal{K} \vdash K'$ if $K' \in \mathcal{K}$. **(2)** $\mathcal{K} \vdash K'$ if $\mathcal{K} \vdash K$ for all $K \rhd K'$. **(3)** $\mathcal{K} \vdash \mathtt{B}_\sigma(G) : \sigma(G)$ for each built-in goal G with a solution $\sigma(G)$.

Next we lift the subsumption and replacement relations to a quasi-order and a strict partial order, respectively, on on derived facts and goals. The *subsumption order* $\leq$ on derived atoms is a quasi-order that is inductively defined such that the following conditions hold: **(1)** $f : F \leq f' : F'$ if $F \leq_\mathrm{ff} F'$; **(2)** $g : G \leq g' : G'$ if $G \leq_\mathrm{gg} G'$. We also define the induced *subsumption equivalence* $K \equiv K'$ as $K \leq K' \wedge K \geq K'$, and the *strict subsumption order* $K < K'$ as $K \leq K' \wedge K \not\equiv K'$. The *replacement order* $\prec$ is then a transitive relation on derived atoms that is inductively defined such that: **(1)** $K < K'$ implies $K \prec K'$, i.e., $\prec$ extends $<$; **(2)** $K \leq K'$, $K' \prec K''$, and $K'' \leq K'''$ implies $K \prec K'''$, i.e., $\prec$ is compatible with $\leq$; **(3)** $f : F \prec f' : F'$ if $F \prec_\mathrm{ff} F'$; **(4)** $g : G \prec g' : G'$ if $G \prec_\mathrm{gg} G'$; **(5)** $g : G \prec f' : F'$ if $G \prec_\mathrm{gf} F'$; **(6)** $f : F \prec g' : G'$ if $F \prec_\mathrm{fg} G'$. Note that in all these relations, derivations are only used to distinguish between derived facts and goals, but no use is made of the structure of the derivation. Derivations can be used for proof-theoretic purposes or for tracing in practice, but are not generally required for an implementation of the logical framework. We obviously require that the replacement order is a strict partial order, which leads to corresponding restrictions on the user-defined replacement relations.

We also require the following *ordering consistency* condition between the two orderings: If $K_1 \leq K_1' \prec K_2 \leq K_2'$ and $K_1 \leq K_2'$ then $K_1' \leq K_2$.

A configuration of a cyber-physical system S is a set of local states $\Gamma \vdash \Delta$ @ t, x, one for each cyber-node x of S. Given a configuration c containing $\Gamma \vdash \Delta$ @ t, x, we write $\mathcal{F}_x(c)$ and $\mathcal{G}_x(c)$ to denote Γ and Δ, respectively. The proof system defines a labeled transition relation $\rightarrow$ on configurations of the cyber-physical system S in the following sense: For configurations c and c', we have $c \rightarrow_r c'$ iff there exist an instance r of a proof rule such that c contains the premises of r, and c' is obtained by an update of c with the conclusion, i.e., by replacing $\Gamma \vdash \Delta$ @ t, x by the conclusion $\Gamma' \vdash \Delta'$ @ t', x. In this case, we also say that r is *applicable* in c.

For readability, we have omitted an implicit side condition $t < t'$ in all proof rules ($t_x < t'_x$ in the communication rules), meaning that time increases monotonically at least by one unit in each step. Furthermore, we view each derived fact or derived goal as a singleton set and use the comma operator to denote set union. Derived goals and derived facts that are subsumption-equivalent are identified. If the comma operator is used in the premise of a proof rule, we always assume that it denotes the union of disjoint sets, i.e., Γ, K implies $K \notin \Gamma$ if it occurs in a premise. For a set $\mathcal{K}$ of derived facts or goals, we write $K \prec \mathcal{K}$ if there exist $K' \in \mathcal{K}$ such that $K \prec K'$, respectively. In the context of a proof rule that has a premise $\Gamma \vdash \Delta$ @ t, x we say that K is *fresh* (at x) if neither $K \in \Gamma, \Delta$ nor $K \prec \Gamma, \Delta$. In the condition of proof rules we use σ to range over all (not necessarily ground) substitutions that satisfy the condition of the proof rule. Some conditions in the reasoning rules will further restrict σ to most general substitutions (MGS).

To give some intuitive explanation of the proof rules in Figure 1, the control rule represents the addition of a new user-level objective to the set of system goals. The observation rule captures the generation of information from the environment. This can happen spontaneously or can be triggered by a goal that a cyber-device attempts to satisfy. The communication rules allow cyber-nodes to exchange facts or goals by means of asynchronous communication. The nature of communication, i.e., whether it is uni-/bidirectional, unicast, multicast, or broadcast remains unspecified. The replacement rules are used to overwrite subsumed and obsolete facts and goals. In this paper, we assume only a loose form of logical time synchronization that satisfies the minimal monotonicity requirement formulated in the communication rules. Of course, this does not preclude implementations with time synchronization that takes place even when no knowledge is exchanged. The forward and backward rules implement forward and backward reasoning. The first forward rule applies an instance of a Horn clause from the underlying theory if all conditions are available as facts, generating a new fact $\sigma(Q)$ corresponding to the conclusion in this process. The second forward rule covers the case where the available facts are not sufficient to apply the clause so that a new subgoal $\sigma(P_j)$ needs to be generated for a missing fact. The backward rules are analogous to the two forward rules, but apply a Horn clause in a goal-directed way, by first unifying the conclusion with an existing goal. Again,

$$\frac{\Gamma \vdash \Delta \ @\ t, x}{\Gamma \vdash \Delta, \mathtt{C}(G) : G \ @\ t', x} \quad \text{if } G = p_c(t, \ldots) \text{ is a cyber-goal} \qquad \textsf{(Control)}$$

$$\frac{\Gamma \vdash \Delta \ @\ t, x}{\Gamma, \mathtt{O}(F) : F \vdash \Delta \ @\ t', x} \quad \text{if } F = p_c(t, \ldots) \text{ is a cyber-fact} \qquad \textsf{(Observation)}$$

$$\frac{\Gamma, f : F \vdash \Delta \ @\ t, x}{\Gamma \vdash \Delta \ @\ t', x} \quad \text{if } f : F \prec \Gamma, \Delta \qquad \textsf{(Replacement1)}$$

$$\frac{\Gamma \vdash \Delta, g : G \ @\ t, x}{\Gamma \vdash \Delta \ @\ t', x} \quad \text{if } g : G \prec \Gamma, \Delta \qquad \textsf{(Replacement2)}$$

$$\frac{\Gamma_x \vdash \Delta_x \ @\ t_x, x \quad \Gamma_y, f : F \vdash \Delta_y \ @\ t_y, y}{\Gamma_x, f : F \vdash \Delta_x \ @\ t'_x, x} \qquad \textsf{(Communication1)}$$
if $x \neq y$, $t'_x \geq t_y$, and $f : F$ is fresh at x.

$$\frac{\Gamma_x \vdash \Delta_x \ @\ t_x, x \quad \Gamma_y \vdash \Delta_y, g : G \ @\ t_y, y}{\Gamma_x \vdash \Delta_x, g : G \ @\ t'_x, x} \qquad \textsf{(Communication2)}$$
if $x \neq y$, $t'_x \geq t_y$, and $g : G$ is fresh at x

$$\frac{\Gamma \vdash \Delta, g : G \ @\ t, x}{\Gamma, \mathtt{B}_\sigma(g) : \sigma(G) \vdash \Delta, g : G \ @\ t', x} \qquad \textsf{(Built-in)}$$
if G is a built-in goal with a solution $\sigma(G)$ such that $\mathtt{B}_\sigma(g) : \sigma(G)$ is fresh.

$$\frac{\Gamma, f_1 : \sigma(P_1), \ldots, f_n : \sigma(P_n) \vdash \Delta \ @\ t, x}{\Gamma, f_1 : \sigma(P_1), \ldots, f_n : \sigma(P_n), f : \sigma(Q) \vdash \Delta \ @\ t', x} \qquad \textsf{(Forward1)}$$
if $l : P_1, \ldots, P_n \Rightarrow Q$ is a clause from Ω_f,
$f = l_\sigma(f_1, \ldots, f_n)$, $\sigma(Q)$ is a fact, and $f : \sigma(Q)$ is fresh.

$$\frac{\Gamma, f_1 : \sigma(P_1), \ldots, f_{j-1} : \sigma(P_{j-1}) \vdash \Delta \ @\ t, x}{\Gamma, f_1 : \sigma(P_1), \ldots, f_{j-1} : \sigma(P_{j-1}) \vdash \Delta, g : \sigma(P_j) \ @\ t', x} \qquad \textsf{(Forward2)}$$
if $l : P_1, \ldots, P_n \Rightarrow Q$ is a clause from Ω_f,
$g = l_\sigma^{-1}(f_1, \ldots, f_{j-1})$, σ is a MGS, $\sigma(P_j)$ is a goal, and $g : \sigma(P_j)$ is fresh.

$$\frac{\Gamma, f_1 : \sigma(P_1), \ldots, f_n : \sigma(P_n) \vdash \Delta, g' : G' \ @\ t, x}{\Gamma, f_1 : \sigma(P_1), \ldots, f_n : \sigma(P_n), f : \sigma(Q) \vdash \Delta, g' : G' \ @\ t', x} \qquad \textsf{(Backward1)}$$
if $l : P_1, \ldots, P_n \Rightarrow Q$ is a clause from Ω_b,
$f = l_\sigma(f_1, \ldots, f_n; g')$, $\sigma(Q) = \sigma(G')$,
$\sigma(Q)$ is a fact, and $f : \sigma(Q)$ is fresh.

$$\frac{\Gamma, f_1 : \sigma(P_1), \ldots, f_{j-1} : \sigma(P_{j-1}) \vdash \Delta, g' : G' \ @\ t, x}{\Gamma, f_1 : \sigma(P_1), \ldots, f_{j-1} : \sigma(P_{j-1}) \vdash \Delta, g' : G', g : \sigma(P_j) \ @\ t', x} \qquad \textsf{(Backward2)}$$
if $l : P_1, \ldots, P_n \Rightarrow Q$ is a clause from Ω_b,
$g = l_\sigma^{-1}(f_1, \ldots, f_{j-1}; g')$, σ is a MGS such that $\sigma(Q) = \sigma(G')$,
$\sigma(P_j)$ is a goal, and $g : \sigma(P_j)$ is fresh.

$$\frac{\Gamma \vdash \Delta \ @\ t, x}{\Gamma \vdash \Delta \ @\ t', x} \qquad \textsf{(Sleep)}$$

Fig. 1. Proof Rules of our Distributed Logical Framework for NCPS

if the present facts are not sufficient to cover all conditions of a clause, a new subgoal is generated. In both forward and backward rules, the selection of facts, goals, and rules is entirely nondeterministic, and many strategies are possible as long as they satisfy some local weak fairness requirements (see below). Finally, the sleep rule allows the system to wait and hence slow down the reasoning for an arbitrary amount of time, e.g., to save energy, wait for new knowledge, or use the resources for other purposes.

An execution of the networked cyber-physical system S is a finite sequence $\pi = c_0, r_0, c_1, r_1, c_2, \ldots, c_n$ or an infinite sequence $\pi = c_0, r_0, c_1, r_1, c_2, \ldots$ of configurations such that $c_i \rightarrow_{r_i} c_{i+1}$ for all i. We say that i is a *step* of π iff $r_i \in \pi$. We say that a rule r is *applied* in π at j iff $r = r_j$.

For the following definition and the subsequent weak fairness properties, we identify instances of proof rules that only differ in Γ, Δ, t, t' (or the corresponding indexed variables). We say that r is *permanently applicable* in π at i iff r is applicable in all steps $j \geq i$ of π.

An execution is *logically fair* iff each instance of a *reasoning rule*, i.e., either a built-in, forward, or backward rule, that is permanently applicable at i is applied at some $j \geq i$. Similarly, an execution is *replacement fair* iff each instance of a *replacement rule* that is permanently applicable at i is applied at some $j \geq i$. An execution is *communication fair* iff each instance of a *communication* rule that is permanently applicable at i is applied at some $j \geq i$. Note that an applicable communication rule can lose applicability if the conclusion has been reached already, which means that direct communication between each pair is not required if the information can be exchanged over multiple hops by other instantiations of the communication rules.

An execution π is *locally fair* iff it is logically fair and replacement fair. An execution is *globally fair* iff it is locally fair and communication fair.

For a given execution π and node x, we denote by $\mathcal{F}_x^O(\pi)$ all derived facts of the form $\mathtt{O}(F) : F$ generated in π by the observation rule and by $\mathcal{G}_x^C(\pi)$ all derived goals of the form $\mathtt{C}(G) : G$ generated in π by the control rule. We define $\mathcal{F}^O(\pi)$ as the union of $\mathcal{F}_x^O(\pi)$ over all nodes x, and define $\mathcal{G}^C(\pi)$ correspondingly.

We now state some general properties of finite of infinite executions $\pi = c_0, r_0, c_1, r_1, c_2, \ldots$ that start with an initial configuration c_0 where each node has an empty set of facts and goals. Most of these properties are achieved independent of the network conditions, i.e., independent of how much information can be exchanged using the communication rules.

We first observe that the only source of non-monotonicity are the replacement rules. It is particularly noteworthy that goals are not automatically removed once they are solved, but they can remain active (until they are replaced), because they may lead to further solutions (expressed as facts) due to additional reasoning and/or due to observed changes in the environment.

Monotonicity: If π does not apply any replacement rules then for all steps i, j of π with $i \leq j$ we have $\mathcal{F}_x(c_i) \subseteq \mathcal{F}_x(c_j)$ and $\mathcal{G}_x(c_i) \subseteq \mathcal{G}_x(c_j)$ for every cyber-node x.

The distributed proof system maintains the following invariant property which implies logical soundness relative to the underlying logic. Below we write $\pi_{|i}$ to denote the prefix $c_0, r_0, c_1, r_1, c_2, \ldots, c_i$ of π. We also use $\mathcal{K} \vdash Q$ to denote $\Phi(\mathcal{K}) \vdash Q$ if $\Phi(\mathcal{K})$ is the set of atoms of $\mathcal{K}$ (i.e., ignoring derivations).

Soundness: For every step i of π, and for each $f : F \in \mathcal{F}(c_i)$, we have $\mathcal{F}^O(\pi_{|i}), \mathcal{G}^C(\pi_{|i}) \vdash f : F$, which in turn implies $\mathcal{F}^O(\pi_{|i}) \vdash F$.

Note that consistent with our intuitive interpretation of local state, nothing is said about $\mathcal{G}(c_i)$ here, which can (and typically will) include unsatisfiable and contradictory goals.

With additonal assumptions we can also show logical completeness relative to the underlying logic. These assumptions take the form of consistency conditions between the theory and the subsumption and replacement orderings. First, we say that *subsumption is preserved* iff $K_i \leq K'_i$ and $K_1, \ldots, K_n \vdash K$ implies that there exists K' such that $K'_1, \ldots, K'_n \vdash K'$ and $K \leq K'$. We say that *replacement is restricted* iff the following conditions hold: **(1)** if $K'_1 \rhd^+ K'_2$ then $K'_1 \not\succ K'_2$; **(2)** if $K'_1 \rhd^+ K'_2$ does not hold then $K'_1 \prec K'_2$ and $K'_1 \not< K'_2$ implies $K_1 \prec K_2$ for some atomic K_1, K_2 with $K_1 \rhd^* K'_1$ and $K_2 \rhd^* K'_2$; **(3)** if $K \rhd^+ K'_1, K'_2$ and $K \prec K'_2$ and neither $K'_1 \rhd^+ K'_2$ nor $K'_2 \rhd^+ K'_1$ then $K'_1 \leq K'_2$; and **(4)** if $K'_1 < K'_2$ then $K_1 \not\succ K_2$ for all atomic K_1, K_2 with $K_1 \rhd^* K'_1$ and $K_2 \rhd^* K'_2$.

To formulate that a computation has generated a derived fact, we say that $f : F$ is *eventually covered in* π iff there is an i such that there exists $f' : F' \in \mathcal{F}(c_i)$ such that $f : F \leq f' : F'$. Note that we do not exclude the possibility that the derived fact that is eventually covered is replaced by new facts or goals in the future. For instance, this may be the case when a derived fact is not needed any more (according to the replacement ordering) after it has been used in the proof of a goal.

Completeness: Assume that subsumption is preserved, upwards well-founded, and replacement is restricted. Let π be a logically fair and communication fair execution, and let $\mathcal{F} \subseteq \mathcal{F}^O(\pi)$ and $\mathcal{G} \subseteq \mathcal{G}^C(\pi)$ such that each element in $\mathcal{F} \cup \mathcal{G}$ is maximal in $\mathcal{F}^O(\pi) \cup \mathcal{G}^C(\pi)$ w.r.t. the replacement ordering. If $\mathcal{F} \vdash_{\mathrm{f}} F$ then there exists a derived fact $f : F$ such that $\mathcal{F} \vdash f : F$, which in turn implies that $f : F$ is eventually covered in π. If $G \in \mathcal{G}$ and $\mathcal{F} \vdash_{\mathrm{b}} \sigma(G)$ then there exists a derived fact $f : \sigma(G)$ such that $\mathcal{F}, \mathcal{G} \vdash f : \sigma(G)$, which in turn implies that $f : \sigma(G)$ is eventually covered in π.

If the network conditions are not sufficient to achieve communication fairness, we can still achieve a local version of this property (see appendix), which could also be formulated more realistically for subsets of cyber-physical nodes.

It is noteworthy that the completeness result is the only place where additional restrictions on the orderings are needed. In particular, soundness (and also the following termination property) does not depend on any such conditions.

Since the reasoning rules are a possible source of nontermination, Ω must be suitably restricted in practice. We define an execution π as *terminating* iff it is finite or has a finite prefix after which only the sleep rule is applied. A practically useful and sufficient condition for termination will be formulated below. Different from facts, a goal can have variables and hence can potentially represent an infinite set of solutions. Hence, we define goal G as *finitary* w.r.t. a set $\mathcal{F}$ of (derived) facts iff if has a finite set of solutions w.r.t. $\mathcal{F}$, where a *solution* w.r.t. $\mathcal{F}$ is any fact $\sigma(G)$ such that $\mathcal{F} \vdash \sigma(G)$. We say that a set $\mathcal{F} \cup \mathcal{G}$ of derived facts and goals has the *finite closure property* iff there exists a set $\mathcal{K}$ of derived atoms such that $\mathcal{F} \cup \mathcal{G} \subseteq \mathcal{K}$ and $(\mathcal{K}, \leq)$, defined subsequently, is a well-founded quasi-order, and for each induced equivalence class $\mathcal{K}'$ then projection on atoms $\mathrm{at}(\mathcal{K}')$ is finite, where $\mathrm{at}(d\!:\!P)$ is defined as P.

Identifying goals that are equivalent modulo consistent renaming of variables, we inductively define the partial order $(\mathcal{K}, \leq)$ such that for all (not necessarily ground) substitutions σ the following conditions hold:

(0) If $g\!:\!G \in \mathcal{K}$ is a built-in goal then $\mathtt{B}_\sigma(g)\!:\!\sigma(G) \in \mathcal{K}$ and $\mathtt{B}_\sigma(g)\!:\!\sigma(G) \leq g\!:\!G$.

(1) If $l\!:\!P_1, \ldots, P_n \Rightarrow Q$ in Ω_f and $\mathcal{K} \vdash f_1\!:\!\sigma(P_1), \ldots, f_n\!:\!\sigma(P_n)$, then $l_\sigma(f_1, \ldots, f_n)\!:\!\sigma(Q) \in \mathcal{K}$, and if $n \geq 1$ we require that $f_i\!:\!\sigma(P_i) \in \mathcal{K}$ implies $l_\sigma(f_1, \ldots, f_n)\!:\!\sigma(Q) \leq f_i\!:\!\sigma(P_i)$.

(2) If $l\!:\!P_1, \ldots, P_n \Rightarrow Q$ in Ω_f with a goal $\sigma(P_j)$ and $\mathcal{K} \vdash f_1\!:\!\sigma(P_1), \ldots, f_{j-1}\!:\!\sigma(P_{j-1})$, then $l_\sigma^{-1}(f_1, \ldots, f_{j-1})\!:\!\sigma(P_j) \in \mathcal{K}$, and if $j \geq 2$ we require that $f_i\!:\!\sigma(P_i) \in \mathcal{K}$ with $i < j$ implies $l_\sigma^{-1}(f_1, \ldots, f_{j-1})\!:\!\sigma(P_j) \leq f_i\!:\!\sigma(P_i)$.

(3) If $l\!:\!P_1, \ldots, P_n \Rightarrow Q$ in Ω_b and $\mathcal{K} \vdash f_1\!:\!\sigma(P_1), \ldots, f_n\!:\!\sigma(P_n)$, and $g'\!:\!G' \in \mathcal{K}$ with $\sigma(Q) = \sigma(G')$, then $l_\sigma(f_1, \ldots, f_n; g')\!:\!\sigma(Q) \in \mathcal{K}$, and we require that $f_i\!:\!\sigma(P_i) \in \mathcal{K}$ implies $l_\sigma(f_1, \ldots, f_n; g')\!:\!\sigma(Q) \leq f_i\!:\!\sigma(P_i)$.

(4) If $l\!:\!P_1, \ldots, P_n \Rightarrow Q$ in Ω_b with a goal $\sigma(P_j)$ and $\mathcal{K} \vdash f_1\!:\!\sigma(P_1), \ldots, f_{j-1}\!:\!\sigma(P_{j-1})$, and $g'\!:\!G' \in \mathcal{K}$ with $\sigma(Q) = \sigma(G')$, then $l_\sigma^{-1}(f_1, \ldots, f_{j-1}; g')\!:\!\sigma(P_j) \in \mathcal{K}$, and we require either that $l_\sigma^{-1}(f_1, \ldots, f_{j-1}; g')\!:\!\sigma(P_j) \leq g'\!:\!G'$, or that $f_i\!:\!\sigma(P_i) \in \mathcal{K}$ with $i < j$ implies $l_\sigma^{-1}(f_1, \ldots, f_{j-1}; g')\!:\!\sigma(P_j) \leq f_i\!:\!\sigma(P_i)$.

Intuitively, the set $\mathcal{K}$ overapproximates the set of all derived facts and goals that could be generated in response to an element from this set, where condition (0) corresponds to the built-in rule, conditions (1) and (2) correspond to the forward rules (which can be applied to solutions of goals), and (3) and (4) correspond to the backward rules. It is noteworthy that none of these conditions requires a strict ordering to ensure termination, because the proof rules cannot generate facts or goals that are already present. We should also note that $\mathcal{K}$ may be infinite, but due to the use of most general substitutions σ in the proof rules, only a finite subset of $\mathcal{K}$ will be generated in any actual execution.

We call an execution π *closed* iff no control or observation rules are applied in π. We say that π is *eventually closed* iff a suffix of π is a closed execution. This property formalizes a test that will enable us to observe the result of the reasoning process, which normally could be interrupted by new facts and goals from the environment or from the user.

Termination: If π is an eventually closed and $\mathcal{F}^O(\pi) \cup \mathcal{G}^C(\pi)$ has the finite closure property then π is terminating.

It is noteworthy that global termination does not need any restictions on the subsumption and replacement orderings, because the replacement rules can never be a cause of nontermination. In fact, in a terminating system only finitely many derived facts/goals can be generated, and hence replacement can only be applied finitely often. Furthermore, once a derived fact/goal is replaced it can never be regenerated due to the freshness conditions in the proof rules.

In a globally fair and terminating system, the replacement and communication rules will eventually ensure that all cyber-nodes will reach the same logical state (disregarding time and name). The same property can be formulated modulo subsumption equivalence, if we do not identify subsumption equivalent units of knowledge.

Confluence: If π is a globally fair and terminating execution then π is *confluent*, i.e., there exists a suffix π' such that $\mathcal{F}_x(c) = \mathcal{F}_y(c)$ and $\mathcal{G}_x(c) = \mathcal{G}_y(c)$ for all cyber-nodes x, y and $c \in \pi'$.

Global termination expresses an important logical termination condition that excludes nontermination due to an illformed declarative specification for a given set of potential facts and goals. Typical cyber-physical systems are however nonterminating, because they are part of a dynamic environment with potentially changing requirements.

The proof system is specifically designed to enable a distributed implementation using randomization techniques that can be used to satisfy the local fairness conditions with probability one. Among other sources, our approach is inspired by the use of randomized backtracking [13] that has been proposed for implementation of Prolog on a parallel architecture. At several places in the proof system nondeterminism at a given node x can be implemented by a randomized choice. First, there is the selection of clauses in the four reasoning rules. Second, there is the selection of facts and/or goals to which the clause is applied. Third, there is the choice of the substitution and hence solution in the rule for built-ins. And finally, there is the sleep rule, which can be implemented by random waiting using a suitable distribution to make the reasoning process adaptive to resource constraints and network conditions.

The idea of applying declarative techniques in communication and networking is not new. They have been, for instance, used for networking policies and protocols in the context of security, routing, or dynamic spectrum access. Specifically, [5] develops a very interesting approach to declarative sensor networks based on Datalog that can transmit generated facts to specific neighbors and can also utilize knowledge about neighbors to specify e.g., routing algorithms. We have presented first steps toward combining forward and backward reasoning in a fully distributed fashion with knowledge that is transparently shared. A fixed or known neighborhood is not assumed in our more abstract approach, and the use and dissemination of both facts and goals aims at general cyber-physcial systems with distributed actuation, and hence leads us beyond sensor networks.

The proof system that we have presented in this paper attempts to focus on a few core ideas, but the work can be generalized in many directions. One step is the generalization of the underlying logic, the incorporation of equational features like in Maude [6] being one example. Furthermore, we have presented proof rules with simplified forward and backward reasoning rules that proceed according to the ordering of atoms in the conditions of a Horn clause, but more general proof strategies are possible that can potentially lead to a higher degree of parallelism. There are also various optimizations that are conceivable. For instance, solved or unsolved goals that cannot generate further solutions are not removed in this paper. The use of an expiration time is a possible approach. Furthermore, since several conflicting goals can be active at the same time, strategies guided by priorization and more generally distributed optimization techniques need to be developed. In fact, the logical framework may be thought of a means of expressing the space of logically sound behaviours, which can be further constrainted by more quantitative techniques. Finally, the semantics in this paper is an interleaving semantics, whereas a true concurrency semantics, where the concurrent application of proofs rules is represented explicitly, is more appropriate.

4 Sample Application

To test our ideas we will focus on a specific application that we call *self-organizing mobile robots* as a special case of controllable networks that captures many interesting aspects of NCPS [7]. Consider a self-organizing network of mobile robots deployed in a building, e.g., to achieve situation awareness during an emergency. This is a challenging test case for various reasons. The network is highly dynamic, and temporary disconnections or failures are part of the normal operation and need to be compensated for by real-world actions. Parameters such as a robot's position can be controlled only indirectly via actions, and costs of changes (e.g., energy consumption) cannot be neglected.

As a concrete sample mission, we chose a primary goal such as delivery of the collected information (e.g., images) from a particular area to a specific node with some time constraints. We assume that each room in the building is equipped with acoustic sensors or motion sensors and the goal is to collect information in areas where noise or motion is detected. The mobile robots have camera devices that can capture a fullsight (i.e., 360-degree view) snapshot of a target area. The raw image may be directly sent to other nodes if the network supports it, or it can be preprocessed, e.g., by applying some form of compression or abstraction, and feature extraction (possibly at a different more powerful node), and then communicated to other nodes.

For the specific example, we assume that the primary goal is injected into the network by the user at a fixed root node around which initially the robots are randomly clustered. Goals and facts are opportunistically shared whenever connectivity exists. Each robot can compute its local solution based on its local knowledge. The solution assigns an approximate target region as a subgoal to each robot, which then starts to move in order to locally realize the goal. The distributed reasoning continues so that the local solutions are continuously recomputed and movements are adjusted correspondingly. In addition to the randomness due to network and environment, randomization techniques, e.g., random selection of clauses and goals, and random waiting, are used to desynchronize the robots. The movements are constrained by the floorplan, which could be opportunistically updated and shared, but is simply assumed to be given as part of the logical theory, which is available at all nodes, in our simple example. The network connectivity model can also be exploited to suppress position changes that would lead to disconnections. Still, temporary disruptions are possible due to system perturbations, failures, and uncertainty caused by delayed/incomplete knowledge.

4.1 Simulation Setup

Figure 2 shows knowledge sharing between three cyber-nodes and their devices. Each node is equipped with a knowledge manager, i.e., an implementation of the distributed knowledge-sharing model, a reasoner, i.e., an implementation of our logical framework, and attached devices that can be regarded as subnodes exhibiting a declarative knowledge-based interface. The devices are using the distributed knowledge-sharing model but are implemented using conventional code (simulation code in our case), i.e., outside the logical framework. Figure 2 contains two mobile robots, each with a positioning device and a camera device. It also contains a fixed sensor node, which can have attached noise or motion sensors or both. In our simulation, we experimented with one to five robots and used a root node that is similar to a robot but assumed to be at a fixed location and serves as a user access point to the cyber-physical system.

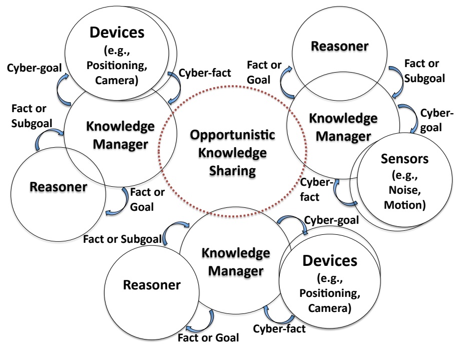

Fig. 2. Distributed Knowledge Sharing between two Robots and a Sensor in an Instrumented Cyber-Space

By definition, cyber-facts/goals have a form $p_c(t, e_1, ...e_n)$. However, in our example we leave the creation time t implicit in the case of cyber-goals, because it is not used in the theory.

To illustrate a simple operation, assume now that a user injects a goal such as $TakeSnapshot(t_T, t_T + \Delta t_{sd}, a, I)$ with $t_T = 0.0$ and $\Delta t_{sd} = 20.0$ into the root node. t_T and Δt_{sd} indicate the earliest time and the deadline to take the snapshot. The reasoner cannot solve this goal by means of the logical theory, but the local camera device may find that it can handle $TakeSnapshot(0.0, 20.0, a, I)$ by taking an image of the area a in the time interval $0.0, ..., 20.0$. As a result it generates a fact $Snapshot(10.0, a, i)$, indicating that image i was taken at time 10.0 in area a, which is added to the local knowledge base and can in turn lead to further reasoning in the logical theory.

Knowledge (i.e., facts and goals) is opportunistically disseminated whenever connectivity exists among robots. In our experiments, we use an abstract topological *mobility model* instead of a model with actual coordinates. We assume that rooms in our scenario correspond to regions exhibiting similar connectivity, and the *network model* is defined so that links are up between robots when they reside in the same or adjacent rooms. For our experiments, we use a disruption-tolerant networking simulator [20] that abstracts from the underlying networking stack. It uses a simple graph-based dynamic network model, where each link has a state, e.g., up or down, and is characterized by its abstract features such as bandwidth, latency, and error rate. The floor plan restricts the mobility and will be reflected by a collection of adjacency facts as part of the theory.

4.2 Declarative Problem Formulation

Figure 3 shows the logical theory that is used to declaratively represent our sample application. Recall that a user wants the system to take an image whenever some trigger condition occurs, process it, and deliver it at the root node. The trigger conditions are specified as forward clauses $F1$ and $F2$, which can be applied at any time when the conditions are met. Like all knowledge, facts are disseminated in the network, hence compensating for heterogeneity in the node capabilities and other limitations, e.g., due to resources and sensor failures.

Different from forward clauses, backward clauses are evaluated only when a user or the reasoner injects a goal (or a new subgoal) that unifies with the conclusion of the clause. For example, the backward clause $B1$ is triggered by an $Interest$ goal and generates a corresponding fact when successfully applied. In the case of $B2$ and $B3$, the reasoner attempts to check if the required image is delivered to the root node. If a corresponding *Delivered* fact is available, then the *Deliver* goal is satisfied by applying $B2$. Otherwise, $B3$ needs to be used to check the current position of the root node (via the *Position* fact) and to guide the robot toward the area where the root node is positioned (via the $MoveTo$ subgoal). $B4$-$B6$ specify how we construct an image I. When one of trigger conditions, $F1$ or $F2$, is met, a robot needs to be located at the specific area (the $MoveTo$ predicate should be satisfied for this purpose) to take a snapshot. Only after these two subgoals, $Trigger$ and $MoveTo$ in $B6$, are satisfied, the $TakeSnapshot$ goal can be realized by the device, which generates *Snapshot* as a new fact. Now, a fact $RawImage$ can be generated as specified by $B6$ and $B7$.

The clauses $B4$ and $B5$ show the processing of a captured image, namely, compression and feature extraction. For example, in the condition of $B5$ the $Compress$ function takes an image I and assigns the compressed image to I'. Raw images are implemented as built-in objects with attributes such as time and area information, and image processing functions such as $Compress$ and $Extract$ do not alter that information.

The *Adjacent* predicate is used to represent the topological floor plan, which in turn determines the network connectivity among robots. $Adjacent(a, b)$ means areas a and b are adjacent rooms. In our simulation setup, communication is possible when robots are located in the same or in two adjacent areas. The forward clause $F3$ captures the assumption that adjacency, and hence connectivity, is symmetric. The clause $B9$ also uses the *Adjacent* predicate to plan the robot movement, since a $Move$ goal can be realized by the robot's positioning device only if the room is adjacent.

The $MoveTo$ goal will generate a new $Position$ fact when it is realized. For example, $MoveTo(T_I, T_T, N_D, 0, T_T + \Delta t_{sd}, R, A)$ is used in the condition as a

Forward Clauses:
$F1 : Noise(T, A) \Rightarrow Trigger(T, A).$
$F2 : Motion(T, A) \Rightarrow Trigger(T, A).$
$F3 : Adjacent(A, B) \Rightarrow Adjacent(B, A).$

Backward Clauses:
$B1 : Interest(T_I, I, R) \Leftarrow Result(T_I, T_T, 0, I), Deliver(T_I, T_T, 1, I, R).$
$B2 : Deliver(T_I, T_T, N_D, I, R) \Leftarrow Delivered(T_I, T_T, N_D, I, R).$
$B3 : Deliver(T_I, T_T, N_D, I, R) \Leftarrow$
$\quad Position(T_P, R, A), Position(T'_P, R', A'), R' \neq R,$
$\quad MoveTo(T_I, T_T, N_D, 0, \infty, R', A), Deliver(T_I, T_T, N_D, I, R).$
$B4 : Result(T_I, T_T, N_D, I') \Leftarrow CompImage(T_I, T_T, N_D, I), I' = Extract(I).$
$B5 : CompImage(T_I, T_T, N_D, I') \Leftarrow RawImage(T_I, T_T, N_D, I), I' = Compress(I).$
$B6 : RawImage(T_I, T_T, N_D, I) \Leftarrow Trigger(T_T, A), T_I \leq T_T,$
$\quad MoveTo(T_I, T_T, N_D, 0, T_T + \Delta t_{sd}, R, A),$
$\quad TakeSnapshot(T_I, T_T, N_D, T_T + \Delta t_{sd}, A, I).$
$B7 : TakeSnapshot(T_I, T_T, N_D, D, A, I) \Leftarrow$
$\quad Snapshot(T_I, T_T, N_D, T_S, A, I), T_T \leq T_S, T_S \leq D.$
$B8 : MoveTo(T_I, T_T, N_D, W', D, R, B) \Leftarrow Position(T_P, R, B), T_P \leq D.$
$B9 : MoveTo(T_I, T_T, N_D, W', D, R, B) \Leftarrow Adjacent(A, B), W' > -b_w, W = W' - 1,$
$\quad MoveTo(T_I, T_T, N_D, W, D, R, A), Move(T_I, T_T, N_D, W', D, R, A, B).$

Replacement Ordering: (f denotes a fact and g a goal and x denotes either)
$O1 : f : Position(t_P, r, \ldots) \prec f : Position(t'_P, r, \ldots)$ if $t_P < t'_P$.
$O2 : x : X(t_I, \ldots) \prec g : Interest(t'_I, \ldots)$ if $t_I < t'_I$.
$O3 : x : X(t_I, t_T, n_D, \ldots) \prec f : Result(t_I, t_T, n_D, \ldots)$ if $x : X \neq f : Result$.
$O4 : x : X(t_I, t_D, n_D, \ldots) \prec f : Deliver(t_I, t_D, n_D, \ldots)$ if $x : X \neq f : Deliver$.

Variables: T: time, D: snapshot deadline, A and B: area, R: robot,
I: image or derived information, N: identifier, W: weight
Constants: Δt_{sd}: relative snapshot deadline (max. delay from trigger event),
b_w: bound for weight (diameter of the floor plan)

Fig. 3. Logical Theory for Distributed Surveilance by a Team of Mobile Robots

subgoal of $B6$, and $Position(0.0, r, a)$ is a fact that the local device of robot r provides as its initial position. The clause $B9$ is of particular interest, since it initiates the position change of a robot. As we see from $B8$, if the current position of a robot is equal to its destination, then the $MoveTo$ predicate is already satisfied. Otherwise, a robot plans to move toward the destination as specified in $B9$. We use W and the bound b_w to limit the depth of the $MoveTo$ subgoal generation. The value W is also used to represent weighted goals to enable locally weight-adaptive goal selection, a first step toward combining distributed reasoning and optimization. When a positioning device has a local choice among several possible $MoveTo$ goals, it favors a $MoveTo$ goal with a higher weight W since higher weight indicates a goal closer to the final destination, 0 being the maximum. Generally, other factors can influence the local goal selection of cyber-physical devices, e.g., in this case, since R is unbound, robots closer to an area with sufficient resources may be more likely to select a corresponding goal if it is easier for them to realize.

Atom	Type	Realization	Meaning
$Adjacent(a, b)$	Fact	Theory	areas a and b are adjacent (represents floorplan)
$Noise(t, a)$	Cyber-Fact	Sensor	noise is detected in the area a at time t
$Motion(t, a)$	Cyber-Fact	Sensor	motion is detected in the area a at time t
$Trigger(t, a)$	Fact	Theory	triggering condition is met in the area a at time t
$Position(t, r, a)$	Cyber-Fact	Positioning	robot r is positioned in the area a at time t
$Interest(t_I, I, r)$	Cyber-Goal	Theory	user at root node r is interested in information I
$Result(t_I, t_T, n_D, I)$	Goal	Theory	an feature extraction I needs to be computed
$CompImage(t_I, t_T, n_D, I)$	Goal	Theory	an abstract image I needs to be computed
$RawImage(t_I, t_T, n_D, I)$	Goal	Theory	an image I needs to be generated
$MoveTo(t_I, t_T, n_D, w, t, R, b)$	Goal	Theory	robot R needs to move to the area b until time t
$Move(t_I, t_T, n_D, w, t, R, a, b)$	Cyber-Goal	Positioning	robot R needs to move from area a to area b until time t
$TakeSnapshot(t_I, t_T, n_D, t, a, I)$	Goal	Theory	a snapshot I needs to be taken in area a between time $t_T, ..., t$
$TakeSnapshot(t_I, t_T, n_D, t, a, I)$	Cyber-Goal	Camera	a snapshot I needs to be taken in area a between time $t_T, ..., t$
$Snapshot(t_I, t_T, n_D, t_s, a, i)$	Cyber-Fact	Camera	a snapshot i is taken in the area a at time t_s
$Deliver(t_I, t_T, n_D, i, r)$	Cyber-Goal	Root Node	request information i to be delivered to user at root node r
$Delivered(t_I, t_T, n_D, i, r)$	Cyber-Fact	Root Node	information i has been delivered to user at root node r

Table 1. Interpretation of Cyber-Predicates and Theory Predicates

In addition to forward and backward clauses, the replacement ordering is specified in Figure 3. Under $O1$ we specify that an old *Position* fact of a robot is replaced by a new *Position* fact based on their timestamps. In $O2$, an old *Interest* goal becomes obsolete in view of a new *Interest* goal based on the time t_I associated with the *Interest* goal. Under $O3$ and $O4$, we specify orderings so that a fact can cancel goals and facts (except itself) that have led to the generation of such a fact. To avoid canceling goals and facts that are not related to a specific fact, we assume that t_I, t_T, n_D can serve as unique identifiers.

Specifically, t_I and t_T are intended to distinguish different interest goals and trigger conditions, respectively. We define t_I and t_T as the time when an interest goal is injected ($B1$) and when a trigger condition occurs ($B6$), respectively. The number n_D, which can be either 0 or 1, distinguishes between $Result$ and $Deliver$ stages in $B1$, and their corresponding subgoals and related facts.

4.3 Sketch of a Distributed Execution

Figure 4 shows duality of two kinds of knowledge: facts and goals. Facts can be derived from observations or by applying forward clauses. Goals can be injected by a user or refined by applying backward clauses. Forward reasoning derives facts from known facts. Backward reasoning refines a goal to a number of subgoals. Facts and goals are both disseminated in the network as shown in Figure 2. A goal can be matched with a fact anywhere in the network as illustrated with dotted lines in Figure 4. For example, the goal $Trigger(T, A)$ can be matched with the fact $Trigger(0.0, a)$ as illustrated in Figure 5.

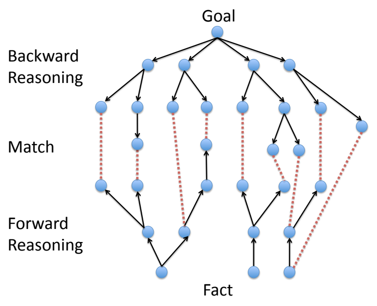

Fig. 4. Matching between Goals and Facts

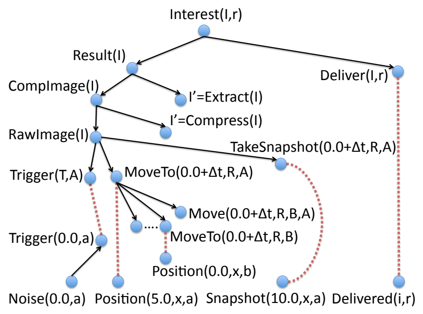

Fig. 5. Example of Distributed Execution

Figure 5 shows a more detailed view of a possible execution of the theory from Figure 3. For brevity, we have omitted some identifier arguments (e.g., T_I,T_T,N_D) in the figure and in the following description unless they are needed. At the top of Figure 5, the user injects a cyber-goal $Interest(I, r)$ at the root node r. At the bottom, a $Noise$ cyber-fact (representing an observation) leads to a fact $Trigger(0.0, a)$ by forward reasoning. In our framework, conditions are processed from left to right as we explained in Section 3. For example, the clause $B1$ in Figure 3 applied to the goal $Interest(I, r)$ attempts to solve $Result(I)$ before $Deliver(I, r)$, and $Result(I)$ fails at first, since a fact $Result(i)$ does not exist yet. The local reasoner feeds $Result(I)$ as a new subgoal into the local knowledge base. The knowledge base contains two goals at this point, $Interest(I, r)$ and $Result(I)$. Assume that $Result(I)$ is randomly selected from the local knowledge base. Subsequently, $CompImage(I)$ and then $RawImage(I)$ are fed into the knowledge base as new subgoals.

Clause $B6$ for $RawImage(I)$ has three subgoals involving $Trigger$, $MoveTo$, and $TakeSnapshot$. The leftmost subgoal can be finally matched with a fact $Trigger(0.0, a)$ that is derived from forward reasoning as depicted at the bottom of Figure 5. Next, the $MoveTo$ goal is further refined by applying $B8$ or $B9$. In case the root node r is positioned in the area a, it is able to detect its position and generate $Position(0.0, r, a)$ as a fact. Otherwise, backward clause $B9$ is used to plan for movement toward the desired area. The cyber-goal $Move(w, t + \Delta t, R, a, b)$ is realized by the local positioning device of a robot x when its current position is a and current time is before the deadline t. As a result, a new cyber-fact such as $Position(5.0, x, a)$ will be generated if the robot x indeed manages to move to a at the time 5.0 assuming a deadline $t + \Delta t = 20.0$. In a similar manner, the local camera device of robot x could take a snapshot of the area and have $Snapshot(10.0, a, i)$ generated to realize $TakeSnapshot(t, a, I)$, which eventually leads to the satisfaction of the $RawImage(I)$ goal.

The goals $CompImage(I)$ and $Result(I)$ can in turn be solved by the robot x, since it has a $RawImage(i)$ available as a fact in its local knowledge base. Alternatively, thanks to the fuly distributed nature of the reasoning process, *Compress* or *Extract* can be solved at other nodes (e.g., depending on resource availability) including the root node r, because $RawImage(i), CompImage(i)$, and $Result(i)$ are facts and disseminated through the network. The backward clause $B3$ is used to steer a robot toward the root node r unless backward clause $B2$ can be applied because the delivery has already been accomplished by means of other nodes or the robot can directly satisfy the delivery to the user (or to a higher-layer application). In the end, $Interest(I, r)$ is selected and satisfied by generating a fact $Delivered(i, r)$ at the root node r.

4.4 Discussion and Variations

The distributed nature of the framework improves the robustness of the system under intended or unintended perturbations. For instance, the system continues to operate correctly, i.e., within the scope of the logical theory, even after a human disables or replaces a robot or moves it to a different location. Complete failure of robots or some of their devices is covered as well. If the system is not already heterogeneous from the beginning, partial failures or resource limitations (e.g., battery charge of a robot) will eventually lead to a heterogeneous system which is why any assumption of homogeneity is avoided.

We have also experimented with several variations of the running example. For instance, the robot movement can be further constrained. To this end, we can add one of the subgoals $MovableFrom(T, R, A)$, $MovableTo(T, R, B)$, or $Movable(T, R)$ in the backward clause $B9$ before the *Move* predicate to check whether a robot R can move away from an area A at time T ($MovableFrom$), to an area A at time T ($MovableTo$), or simply move at time T without refering to an area ($Movable$).

Now, a coverage constraint can be specified as

$$\begin{gathered} Position(T, R, A), Position(T', R', A), R \neq R', T'' - T \leq b_p, T'' - T' \leq b_p \\ \Rightarrow MovableFrom(T'', R, A). \end{gathered}$$

where b_p is a sufficiently small delay bound for the position update from other robots to ensure that the knowledge is not obsolete. With this clause, an additional robot needs to exist within the same area to formulate that the area should not be left uncovered.

In a similar manner, a connectivity constraint can be specified as

$$Position(T', R', B), R \neq R', T'' - T' \leq b_p \\ \Rightarrow MovableTo(T'', R, B).$$

$$Position(T', R', A), Adjacent(A, B), R \neq R', T'' - T' \leq b_p \\ \Rightarrow MovableTo(T'', R, B).$$

With this clause, a robot only can move to certain area if the move is not expected not break the robot's connectivity to other robots. The robot R' at the area C will be connected with robot R at the area B after the move.

Another interesting example can be an energy constraint such as

$$Energy(T, R, E), E \geq e \Rightarrow Movable(T, R)$$

that can check the residual an energy level of a robot to make sure it has a sufficient amount of energy to change its position.

Our simple example assumes that the floor plan is given in advance. However, generalizing techniques such as SLAM (simultaneous localization and mapping) [4], which are well established in robotics, models could continuously adapt to the facts derived from observations, specifically the floor plan and the connectivity model. In the case of the floor plan, observations could be simply added as facts and SLAM could benefit from consequences generated based on a background theory. In the case of connectivity, the model parameters could be adapted to match the facts generated by neighbor discovery or even (distributed) signal strength measurements if available. SLAM could be generalized to a distributed mapping of signal strength or more generally the wireless spectrum for the purpose of network optimization [12]. An interesting direction would be to explore how such algorithms can be developed in a declarative framework.

5 Related Work

The idea of applying declarative techniques in communication and networking is not new. A large body of work exists in the areas of specification, analysis, and synthesis of networking policies and protocols, e.g., in the context of security, routing, or dynamic spectrum access.

Declarative querying of sensor networks has been studied through several approaches, for instance in [22], which composes services on the fly and in a goal-driven fashion using a concept of semantic streams. Declarative techniques to specify destinations have been used in disruption-tolerant networking [2]. A variant of Datalog has been applied to the declarative specification of peer-to-peer protocols in the P2 system [15]. Based on this work, [5] develops a very interesting approach to declarative sensor networks that can transmit generated facts to specific neighbors and can also utilize knowledge about neighbors to specify, e.g., routing algorithms. The promising idea of providing an abstraction that views a system as a single asset (an ensemble) rather then programming its individual components has been explored in several papers. Most interesting, the approach adapted in Meld [1] extends the ideas from declarative sensor networks to modular robots, i.e., ensembles of robots with inter-robot communication limited to immediate neighbors. As an example, the movement of a composite robot emerges as a result of the coordinated interaction between its homogeneous robot modules. An earlier functional approach to programming sensor networks is the Regiment macro-programming system [16], which uses a stream-based data-flow language. Most of the work focuses not on the theoretical foundations, but on the efficient compilation into a conventional programming language, which is one possible approach for practical deployment. Another approach, which we sometimes refer to as *embedded formal methods*, is the use of an effient reasoning engine in embedded systems such as software-defined radios [8] or routers [20] as explored in the context of disruption-tolerant networking.

In this paper, we have presented first steps toward combining forward and backward reasoning in a fully distributed fashion with knowledge that is transparently shared. A fixed or known neighborhood is not assumed in our more abstract approach, and the use and dissemination of both facts and goals aims at general cyber-physical systems with distributed actuation, and hence leads us beyond sensor networks, in particular to dynamic sensor/actuator networks that are, unlike ensembles, inherently heterogeneous.

6 Conclusions and Future Work

We have presented a distributed computational and logical foundation for declarative control of NCPS. Our approach is based on distributed knowledge sharing and supports a broad spectrum of operations between autonomy and cooperation with minimum assumptions on network connectivity. Specifically, we developed a distributed inference system based on Horn clause logic with built-ins and cyber-predicates that can capture the interaction with the physical world. In the underlying distributed computing model, facts and goals are represented as knowledge that can be shared opportunistically and guide the distributed reasoning process. The duality between facts and goals extends to the proof system, which treats forward and backward reasoning on an equal footing. In our logical framework, explicit derivations are maintained for both, facts and goals, generalizing the concept of proof objects, which traditionally only serve as witnesses for theorems. Essential properties of our logical framework, such as soundness, completeness, and termination, have been established under very general conditions.

A key feature of our distributed logical framework is its dynamic and interactive nature, meaning that facts represent observations, and goals can lead to changes in the environment that will manifest themselves as new facts flowing into the system. Whether an original local goal can be solved is often secondary, because the combined effect of a set of local goals on the cyber-physical system and its nondeterministic dynamics can lead to solutions of higher-level goals even without requiring solutions of all lower levels. To cope with system perturbations and unexpected failures, it is essential that local goals are not eliminated as soon as they are satisfied, because their main purpose, namely steering the system toward the satisfaction of a high-level property, may not have been achieved yet. Only after certain high-level goals are reached, auxiliary goals can be eliminated so that they do not have to comptete with new goals that steer the system toward the next high-level objective. Real-world systems are more complex, because such processes take place concurrently at multiple levels of abstraction and at different time scales. Our combination of a logic with an underlying partial order facilitates a new declarative specification style, which is particularly suitable for open distributed systems that can interact with a cooperative or uncooperative environment.

For experimentation, we have implemented a prototype of our logical framework that heavily relies on randomization techniques to achieve decoupling and implement locally fair nondeterminism that is used in our proof system to cover a broad range of possible implementations with different reasoning strategies. Our experiments with the simulation of an abstract networked multirobot system are a snapshot of ongoing work, but illustrate the application and the features of our framework in a simplified but nontrivial setting, and indicate possible directions for future work.

For realistic experiments, we plan to make use of existing robotic abstraction layers, such as the open-source Player/Stage/Gazebo Project [19]. Experiments in a real-world multirobot testbed similar to that of SRI's Centibots [17] and Commbots [10] projects are another possible direction. Beyond multirobot teams, we see opportunities to apply (extensions of) our framework to other unmanned autonomous vehicles, networks of pico-satellites [21], instrumented smart spaces [14], ad hoc social networking in a cyber-physical (instrumented) world, and next-generation adaptive networks and cognitive radios [11], where the devices exhibit a high degree of flexibility and the network can be morphed to adapt to user objectives and policies. Clearly, a lot of work remains ahead, because many of these applications require the combination of distributed reasoning with distributed optimization, and the use of weighted goals in our example is only a very first step in this direction.

Acknowledgments Support from National Science Foundation Grant 0932397 (A Logical Framework for Self-Optimizing Networked Cyber-Physical Systems) and Office of Naval Research Grant N00014-10-1-0365 (Principles and Foundations for Fractionated Networked Cyber-Physcial Systems) is gratefully acknowledged. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NSF or ONR.

References

1. Michael P. Ashley-Rollman, Seth Copen Goldstein, Peter Lee, Todd C. Mowry, and Padmanabhan Pillai. Meld: A declarative approach to programming ensembles. In *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS '07)*, October 2007.

2. P. Basu, R. Krishnan, and D. W. Brown. Persistent delivery with deferred binding to descriptively named destinations. In *Proc. of IEEE Military Communications Conference*, 2008.
3. Paola Bruscoli and Alessio Guglielmi. A tutorial on proof theoretic foundations of logic programming, 2003.
4. Howie Choset and K. Nagatani. Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization. *IEEE Trans. on Robotics and Automation*, 17(1):125–137, April 2001.
5. David Chu, Lucian Popa, Arsalan Tavakoli, Joseph M. Hellerstein, Philip Levis, Scott Shenker, and Ion Stoica. The design and implementation of a declarative sensor network system. In *SenSys '07: Proc. of the 5th International Conference on Embedded Networked Sensor Systems*, pages 175–188, 2007.
6. Manuel Clavel, Francisco Durán, Steven Eker, Patrick Lincoln, Narciso Martí-Oliet, José Meseguer, and Carolyn L. Talcott, editors. *All About Maude - A High-Performance Logical Framework, How to Specify, Program and Verify Systems in Rewriting Logic*, volume 4350 of *Lecture Notes in Computer Science*. Springer, 2007.
7. Falko Dressler. *Self-Organization in Sensor and Actor Networks*. Wiley, 2008.
8. Daniel Elenius, Grit Denker, and Mark-Oliver Stehr. A semantic web reasoner for rules, equations and constraints. In *RR '08: Proc. of the 2nd International Conference on Web Reasoning and Rule Systems*, pages 135–149, Berlin, Heidelberg, 2008. Springer-Verlag.
9. Stephen Farrell and V. Cahill. *Delay- and Disruption-Tolerant Networking*. Artech House, Inc., Norwood, MA, USA, 2006.
10. Brian P. Gerkey, Roger Mailler, and Benoit Morisset. Commbots: Distributed control of mobile communication relays. In *Proc. of the AAAI Workshop on Auction Mechanisms for Robot Coordination (AuctionBots)*, pages 51–57, 2006.
11. Gregory D. Troxel et al. Enabling open-source cognitively-controlled collaboration among software-defined radio nodes. *Comput. Netw.*, 52(4):898–911, 2008.
12. Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Nonlinear constraint network optimization for efficient map learning. *Trans. Intell. Transport. Sys.*, 10(3):428–439, 2009.
13. V. K. Janakiram, D. P. Agrawal, and R. Mehrotra. A randomized parallel backtracking algorithm. *IEEE Trans. Comput.*, 37(12):1665–1676, 1988.
14. M. Kim, D. Massaguer, N. Dutt, S. Mehrotra, S. Ren, M.-O. Stehr, C. Talcott, and N. Venkatasubramanian. A semantic framework for reconfiguration of instrumented cyber physical spaces. In *Workshop on Event-based Semantics, CPS Week*, 2008.
15. Boon Thau Loo, Tyson Condie, Minos Garofalakis, David E. Gay, Joseph M. Hellerstein, Petros Maniatis, Raghu Ramakrishnan, Timothy Roscoe, and Ion Stoica. Declarative networking. *Commun. ACM*, 52(11):87–95, 2009.
16. Ryan Newton, Greg Morrisett, and Matt Welsh. The regiment macroprogramming system. In *IPSN '07: Proc. of the 6th International Conference on Information Processing in Sensor Networks*, pages 489–498, New York, NY, USA, 2007. ACM.
17. Charles L. Ortiz, Régis Vincent, and Benoit Morisset. Task inference and distributed task management in the Centibots robotic system. In *AAMAS '05: Proc. of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 860–867, 2005.
18. Peter Padawitz. *Computing in Horn clause theories*. Springer-Verlag, 1988.

19. Radu Bogdan Rusu, Alexis Maldonado, Michael Beetz, and Brian Gerkey. Extending Player/Stage/Gazebo towards cognitive robots acting in ubiquitous sensor-equipped environments. In *ICRA '07: Proc. of the IEEE International Conference on Robotics and Automation Workshop for Network Robot Systems*, 2007.
20. Mark-Oliver Stehr and Carolyn Talcott. Planning and learning algorithms for routing in disruption-tolerant networks. In *Proc. of IEEE Military Communications Conference*, 2008.
21. S. Toorian, K. Diaz, and S. Lee. The CubeSet approach to space access. In *Aerospace Conference, IEEE*, 2008.
22. Kamin Whitehouse, Feng Zhao, and Jie Liu. Semantic streams: A framework for composable semantic interpretation of sensor data. In *Proc. of the European Workshop on Wireless Sensor Networks*, pages 5–20. EWSN, 2006.

A Proofs

Definition 1. *Given a derived fact or goal, K, we define $Base(K)$ as follows:*
$Base(K) = \{K\}$ if K is $\mathtt{O}(F):F$ or $\mathtt{C}(G):G$.
$Base(\mathtt{B}_\sigma(g):G)$ is $Base(g:G')$ where G' is uniquely determined by g.
$Base(L(d_1,...,d_k):Q) = Base(d_1:P_1) \cup \ldots \cup Base(d_k:P_k)$,
where L is any one of the derivation constructors and $P_1,\ldots,P_k$ are uniquely determined by $d_1,\ldots,d_k$.

Lemma 1. *The following basic statements hold:*
(1) $Base(K) \vdash K$ (by induction on construction of K).
(2) $\mathcal{K} \vdash K$ implies $\mathcal{K}, \mathcal{K}' \vdash K$.
(3) $\mathcal{K} \vdash K$ implies $Base(K) \subseteq Base(\mathcal{K})$.

Lemma 2 (Subderivations). *For any execution π, if $K \in \mathcal{F}(\pi) \cup \mathcal{G}(\pi)$ then all subderivations of K are in $\mathcal{F}(\pi) \cup \mathcal{G}(\pi)$.*

Proof. **(1)** Case $K = f:F$. **(a)** Case $f = \mathtt{O}(F) \in \mathcal{F}$. K has no subderivations **(b)** Case $f = l_\sigma(f_1,...,f_n[,g])$ (forward rule without g, backward rule with g). Then $f:F$ was introduced by Forward1 or Backward1 with $l:P_1,\ldots,P_n \Rightarrow Q$ in Ω, $F = \sigma(Q)$ and $\sigma(Q) = \sigma(G)$. Thus for $1 \leq k \leq n$, $f_k:F_k \in \mathcal{F}(\pi)$ at the point of rule application and if l is a backward clause, then $g:G \in \mathcal{G}(\pi)$ at that point. **(c)** Case $f = \mathtt{B}_\sigma(g)$ with $F = \sigma(G)$. Then K was introduced by the Built-in rule and $g:G \in \mathcal{G}(\pi)$ at the point of rule application.
(2) Case $K = g:G$. **(a)** Case $g = \mathtt{C}(G)$. Then K has not subderivations. **(b)** Case $g = l_\sigma^{-1}(f_1,\ldots,f_{j-1}[,g'])$. Then K was introduced by Forward2 or Backward2 and thus for $1 \leq k < j$, $f_k:F_k \in \mathcal{F}(\pi)$ at the point of rule application, and if l is a backward clause then $g':G' \in \mathcal{G}(\pi)$ at that point (and $\sigma(Q) = \sigma(G)$).

Definition 2 (Horn clause proof and provability). *Given a theory Ω, we inductively define Horn clause proof objects p and the relation p proves $\mathcal{F} \vdash Q$ as follows:*
***(1)** $*$ proves $\mathcal{F} \vdash G$ if G is a solution to a builtin goal.*
***(2)** $\mathtt{O}(F)$ proves $\mathcal{F} \vdash F$ if $F \in \mathcal{F}$.*
***(3)** $l_\sigma(p_1,\ldots,p_n)$ proves $\mathcal{F} \vdash \sigma(Q)$ if $l:P_1,\ldots,P_n \Rightarrow Q$ in Ω and p_i proves $\mathcal{F} \vdash \sigma(P_i)$ for $i \in 1,\ldots,n$.*

For a Horn clause proof p, $\sigma(p)$ is defined by composing σ with the substitution used in each rule application. Clearly, if p proves $\mathcal{F} \vdash P$, where P is any atom, not necessarily ground, then $\sigma(p) proves \mathcal{F} \vdash \sigma(P)$,

Note that Horn clause proofs are independent of the partition into forward and backward clauses.

Lemma 3 (Derivability implies provability).
If $f:F$ is a derived fact then $\Phi(Base(f:F)) \vdash F$.

Proof. Let $\mathcal{F} = \Phi(Base(f:F))$. We construct the proof p.
(1) Case $B_\sigma(g):\sigma(G)$. Then p is $*$ by definition since σ must be a solution for

G. **(2)** Case $\mathtt{O}(F) : F$. Then p is $\mathtt{O}(F)$. **(3)** Case $l_\sigma(f_1, \dots, f_n, [g']) : \sigma(Q)$, with $l : P_1, \dots, P_n \Rightarrow Q$ in Ω By IH we have p_i proves $\mathcal{F} \vdash f_i : \sigma(P_i), 1 \leq i \leq n$ thus p is $l_\sigma(p_1, \dots, p_n)$, applying clause l. □

Definition 3 (Grounded proof). *A Horn clause proof p of $\mathcal{F} \vdash F$ is grounded if F is a fact (ground atom) and all clauses used in p are fully instantiated (ground), i.e., if $p = l_\sigma(p_1, \dots, p_n)$, $l : P_1, \dots, P_n \Rightarrow Q$ in Ω, then $\sigma(P_i)$ is a ground atom for $1 \leq i \leq n$ and each p_i proving $\mathcal{F} \vdash \sigma(P_i)$ is similarly grounded.*

Lemma 4 (Horn clause proofs are groundable). *If p is a Horn clause proof of $\mathcal{F} \vdash F$, as above then there is some grounded $p!$ that is a Horn clause proof of $\mathcal{F} \vdash F$.*

Proof. **(1)** If p is $*$ or $\mathtt{O}(F)$ we are done. **(2)** If p is $l_\sigma(p_1, \dots, p_n)$ with $l : P_1, \dots, P_n \Rightarrow Q$ in Ω, (and $\sigma(Q) = F$), let σ' map the remaining variables of $\sigma(P_i)$ to ground terms (since variables are unptyped, 0 will do). By induction let $p_i!$ be grounded proofs of $\mathcal{F} \vdash \sigma'(\sigma(P_i))$. It follows that $l_{\sigma' \circ \sigma}(p_1!, \dots, p_n!) \vdash F$, where $\circ$ is composition of substitutions. □

Lemma 5 (Provability implies derivability). *Let $\mathcal{F}$,$\mathcal{G}$ be atomic derived facts and goals, respectively. **(1)** Let p be any grounded Horn clause proof of $\Phi(\mathcal{F}) \vdash_{\mathrm{f}} F$. Then there exists a derived fact $f : F$ such that $\mathcal{F} \vdash f : F$.*
***(2)** Let p be any grounded Horn clause proof of $\Phi(\mathcal{F}) \vdash_{\mathrm{b}} F$ where $F = \sigma(G)$ for some g such that $g : G \in \mathcal{G}$. Then there exists a derived fact $f : \sigma(G)$ such that $\mathcal{F}, \mathcal{G} \vdash f : \sigma(G)$*

Proof. Note that, generally, if $\mathcal{F}, \mathcal{G} \vdash f : F$ and $Base(\mathcal{G})$ is empty, then $\mathcal{F} \vdash f : F$. We now prove the lemma by induction on the construction of p. **(1)** Assume p is $l_\sigma(p_1, \dots, p_n)$ where p_k proves $\sigma(P_k), 1 \leq k \leq n, F = \sigma(Q)$, and $l : P_1, \dots, P_n \Rightarrow Q$ in Ω_f. If p_k is an atomic proof or the last rule is forward then by induction there is f_k such that $\mathcal{F} \vdash f_k : \sigma(P_k)$. If p_j ends with $l'_{\sigma'}(p'_1, \dots, p'_k)$ where $l' : P'_1, \dots, P'_k \Rightarrow Q'$ in Ω_b then, by inner induction we have $\mathcal{F}, \mathcal{G} \vdash g' : G' = l^-1_\sigma(f_1, \dots, f_{j-1}) : \sigma(P_j)$ and so there is $f : k$ such that $\mathcal{F}, \mathcal{G} \vdash p_k$.

(2) Assume p is $l_{\sigma'}(p_1, \dots, p_n)$ where p_k proves $\sigma'(P_k), 1 \leq k \leq n, F = \sigma(G)$, $g : G \in \mathcal{G}$, and $l : P_1, \dots, P_n \Rightarrow Q$ in Ω_b, and $F = \sigma'(Q)$. Then by induction let f_k be such that $\mathcal{G}$, $\mathcal{F} \vdash f_k : \sigma'(P_k)$, $1 \leq k \leq n$. It follows that f is $l_\sigma(f_1, \dots, f_n, g)$. Note that by the fact that renaming variables in derived atoms does not change the atom, if $\sigma'(Q) = \sigma''(G)$ then there is some σ such that $\sigma(Q) = \sigma'(Q) = \sigma''(G) = \sigma(G)$. □

Theorem 1 (Monotonicity). *If π does not apply any replacement rules, then for all steps i, j of π with $i \leq j$ we have $\mathcal{F}_x(c_i) \subseteq \mathcal{F}_x(c_j)$ and $\mathcal{G}_x(c_i) \subseteq \mathcal{G}_x(c_j)$ for every cyber-node x.*

Proof. By inspection of the inference rules, the only rules that remove a derived fact or goal are the replacement rules. □

Note that if replacement rules are allowed we still have monotonicity modulo the replacement relation. For example if $K \in \mathcal{F}_x(c_j) \cup \mathcal{F}_x(c_j)$, $j < i$ and K is not it $\mathcal{F}_x(c_j) \cup \mathcal{F}_x(c_j)$ then there is K' such that $K \prec K'$ and $K' \cap \mathcal{F}_x(c_j) \cup \mathcal{F}_x(c_j)$.

Theorem 2 (Soundness). *For every step i of π, and for each $f : F \in \mathcal{F}(c_i)$, we have $\mathcal{F}^O(\pi_{|i}), \mathcal{G}^C(\pi_{|i}) \vdash f : F$, which in turn implies $\mathcal{F}^O(\pi_{|i}) \vdash f : F$*

Proof. Let π be an execution $c_0, r_1, \ldots, r_i, c_i, \ldots$, and $f : F \in \mathcal{F}(c_i)$. By Lemma 3, we only need to show $\mathcal{F}^O(\pi|i)$, $\mathcal{G}^C(\pi_{|i}) \vdash f : F$ We also show that $\mathcal{F}^O(\pi_{|i})$, $\mathcal{G}^C(\pi_{|i}) \vdash g : G$ for $g : G \in \mathcal{G}(c_i)$.

We proceed by induction on i. We note that if $\mathcal{F}^O(\pi_{|i-1})$, $\mathcal{G}^C(\pi_{|i-1}) \vdash f : F$ then $\mathcal{F}^O(\pi_{|i})$, $\mathcal{G}^C(\pi_{|i}) \vdash f : F$ (monotonicity of $\vdash$).

We only need to consider rules r_i that introduce a new derived fact $f : F$ at some cyber-node x. There are five cases for facts and four for goals:

(Observation) The new derived fact is $\mathtt{O}(F) : F$, which is in $\mathcal{F}^O(\pi_{|i})$.

(Control) The new derived goal is $\mathtt{C}(G) : G$, which is in $\mathcal{G}^C(\pi_{|i})$.

(Communication1) The new derived fact $f : F$ is in $\mathcal{F}(c_{i-1})$. By induction, $\mathcal{F}^O(\pi_{|i-1})$, $\mathcal{G}^C(\pi_{|i-1}) \vdash f : F$ and we are done since $\mathcal{F}^O(\pi_{|i-1}), \mathcal{G}^C(\pi_{|i-1}) = \mathcal{F}^O(\pi_{|i}), \mathcal{G}^C(\pi_{|i})$.

(Communication2) Analogous to (Communication1).

(Builtin) The new derived fact is $\mathtt{B}_\sigma(g) : \sigma(G)$, where $g : G \in \mathcal{G}(c_{i-1})$. By induction, $\mathcal{F}^O(\pi_{|i-1})$, $\mathcal{G}^C(\pi_{|i-1}) \vdash g : G$ and thus $\mathcal{F}^O(\pi_{|i})$, $\mathcal{G}^C(\pi_{|i}) \vdash f : F$.

(Forward1) The new derived fact is $f : F$ where $F = \sigma(Q)$ and $f = l_\sigma(f_1, \ldots, f_n)$, $l : P_1, \ldots, P_n \Rightarrow Q$ in Ω_f. $f_j : \sigma(P_j) \in \mathcal{F}(c_{i-1})$, $1 \leq j \leq n$. By induction $\mathcal{F}^O(\pi_{|i-1})$, $\mathcal{G}^C(\pi_{|i-1}) \vdash f_j : F_j$ for $1 \leq j \leq n$ and so $\mathcal{F}^O(\pi_{|i})$, $\mathcal{G}^C(\pi_{|i}) \vdash f : F$.

(Backward1) The new derived fact is $f : F$ where $F = \sigma(Q)$ and $f = l_\sigma(f_1, \ldots, f_n, g')$, $l : P_1, \ldots, ..., P_n \Rightarrow Q$ in Ω_b, $f_j : \sigma(P_j) \in \mathcal{F}(c_{i-1})$ for $1 \leq j \leq n$ and $g' : G' \in \mathcal{G}(c_{i-1})$, $\sigma(Q) = \sigma(G')$, so by induction $\mathcal{F}^O(\pi_{|i-1})$, $\mathcal{G}^C(\pi_{|i-1}) \vdash f_j : F_j$ for $1 \leq j \leq n$ and thus $\mathcal{F}^O(\pi_{|i})$, $\mathcal{G}^C(\pi_{|i}) \vdash f : F$.

(Forward2) The new derived goal is $g : G$ where $G = \sigma(P_j)$ and $g = l_\sigma^{-1}(f_1, \ldots, f_{j-1})$ where $l : P_1, \ldots, P_n \Rightarrow Q$ in Ω_f. $f_k : \sigma(P_k) \in \mathcal{F}(c_{i-1})$, $1 \leq k < j$. By induction $\mathcal{F}^O(\pi_{|i-1})$, $\mathcal{G}^C(\pi_{|i-1}) \vdash f_k : F_k$ for $1 \leq k \leq j$ and thus $\mathcal{F}^O(\pi_{|i})$, $\mathcal{G}^C(\pi_{|i}) \vdash g : G$.

(Backward2) The new derived goal is $g : G$ where $G = \sigma(P_j)$ and $g = l_\sigma^{-1}(f_1, \ldots, f_{j-1}, g')$ where $l : P_1, \ldots, P_n \Rightarrow Q$ in Ω_b, $f_k : \sigma(P_k) \in \mathcal{F}(c_{i-1})$, $1 \leq k < j$, $g' : G' \in \mathcal{G}(c_{i-1})$ and $\sigma(G') = \sigma(Q)$ so by induction $\mathcal{F}^O(\pi_{|i-1})$, $\mathcal{G}^C(\pi_{|i-1}) \vdash f_k : F_k$ for $1 \leq k < j$ $\mathcal{F}^O(\pi_{|i-1})$, $\mathcal{G}^C(\pi_{|i-1}) \vdash g' : G'$ and thus $\mathcal{F}^O(\pi_{|i})$, $\mathcal{G}^C(\pi_{|i}) \vdash g : G$. □

For the proof of completeness we restate the restrictions on replacement in a slightly different form:

Definition 4 (Replacement Restrictions). *Replacement is restricted iff the following conditions hold: (1) If $K_1 \prec K_2$, then $K_2 \not\triangleright^+ K_1$. (2) $K_1 \prec K_2$ and $K_1 \not\triangleright^+ K_2$ and $K_1 \not\prec K_2$, then there exists atomic K_1', K_2' such that $K_1' \triangleright^*$*

*$K_1 \wedge K_2' \rhd^* K_2 \wedge K_1' \prec K_2'$. **(3)** If $K_1 \prec K_2$, $K_1 \rhd^+ K_2$, $K_1 \rhd^+ K_3$, $K_3 \not\rhd^+ K_2$, and $K_2 \not\rhd^+ K_3$, then $K_3 \leq K_2$. **(4)** If $K_1 < K_2$ then for all atomic K_1', K_2', if $K_1' \rhd^* K_1$ and $K_2' \rhd^* K_2$, then $K_2' \not\prec K_1'$*

Theorem 3 (Completeness). *Assume that subsumption is preserved and is upward well-founded and that replacement is restricted. Assume, furthermore, that π is a logically fair and communication-fair execution. $\mathcal{F} \subseteq \mathcal{F}^O(\pi)$ and $\mathcal{G} \subseteq \mathcal{G}^C(\pi)$ such that each element of $\mathcal{F}, \mathcal{G}$ is replacement-maximal in $\mathcal{F}^O(\pi) \cup \mathcal{G}^C(\pi)$. The the following two statements hold: **(1)** If $\mathcal{F} \vdash_{\mathrm{f}} F$ then there exists a derived fact $f : F$ such that $\mathcal{F} \vdash f : F$, which in turn implies that $f : F$ is eventually covered in π. **(2)** If $G \in \mathcal{G}$ and $\mathcal{F} \vdash_{\mathrm{b}} \sigma(G)$ then there exists a derived fact $f : \sigma(G)$ such that $\mathcal{F}, \mathcal{G} \vdash f : \sigma(G)$, which in turn implies that $f : \sigma(G)$ is eventually covered in π.*

Proof. We have lemmas that show: **(i)** $\mathcal{F} \vdash_{\mathrm{f}} F$ implies $\mathcal{F} \vdash f : F$. **(ii)** $\mathcal{F} \vdash_{\mathrm{b}} \sigma(G)$ for $G \in \mathcal{G}$ implies $\mathcal{F}, \mathcal{G} \vdash f : \sigma(G)$ for some f, i.e., proofs yield derivations if there are sufficient goals.

Hence we only have to show the "in turn" part of (1) and (2). Let $\mathcal{K}(\pi)$ denote $\mathcal{F}(\pi) \cup \mathcal{G}(\pi)$, all the knowledge (i.e. derived atoms) of π.

Instead of (1), we prove something slightly stronger: **(a)** If $\mathcal{F} \vdash f : F$ (not atomic) there is some $f' : F' \in \mathcal{F}(\pi)$ such that $f : F \leq f' : F'$ and $\mathcal{F} \vdash f' : F'$. **(b)** If $f : F \rhd^+ K_3$ with $\mathcal{F} \vdash K_3$ then either there exists K_3 such that $K_3 \leq K_3' \in \mathcal{K}(\pi)$ or we can pick $f' : F'$ replacement-maximal.

Abstracting to general derivations, covering (1) and (2), we will prove: **(a)** If $\mathcal{F}, \mathcal{G} \vdash K_1$ (not atomic) there is some $K_2 \in \mathcal{K}(\pi)$ such that $K_1 \leq K_2$ and $\mathcal{F}, \mathcal{G} \vdash K_2$. **(b)** Furthermore, if $K_1 \rhd^+ K_3$ with $\mathcal{F}, \mathcal{G} \vdash K_3$ then either there exists K_3 such that $K_3 \leq K_3' \in \mathcal{K}(\pi)$ or we can pick K_2 replacement-maximal in π.

With slight shift of notation, assume if $\mathcal{F}, \mathcal{G} \vdash K_3$ (not atomic) and we want to show there is some $K_3' \in \mathcal{K}(\pi)$ such that $K_3 \leq K_3'$ and $\mathcal{F}, \mathcal{G} \vdash K_3'$. Suppose this is not the case.

Suppose that for each immediate subderivation K_1 of K_3 we find $K_1' \in \mathcal{K}(\pi)$ such that $K_1 \leq K_1'$, $\mathcal{F}, \mathcal{G} \vdash K_1'$, and K_1' is replacement-maximal in π. Then by persistance we have K_3' with $K_3 \leq K_3'$ such that $K_1', \ldots, K_m' \vdash K_3'$ (and hence $\mathcal{F}, \mathcal{G} \vdash K_3'$). If the Ki' are immediate subderivations, the relevant reasoning rule is enabled til it fires and we are done. It is possible that some subderivation K'' of K_3' is not one of the K_i'. Then we have to repeat the argument for $\mathcal{F}, \mathcal{G} \vdash K_3'$ for K''. Note we have to restart the induction as K_3' may be larger than K_3. However, we obtain K_3' with $K_3 \leq K_3'$ and by upward wellfoundedness of subsumption, we eventually find the desired subsumption of K_3.

Now we are reduced to finding $K_1' \in \mathcal{K}(\pi)$ such that $K_1 \leq K_1'$, $\mathcal{F}, \mathcal{G} \vdash K_1'$, and K_1' is replacement-maximal in π for each immediate subderivation of K_3.

Case analysis on K_1 yields the following cases:
(Case 1) The subderivation K_1 is atomic. **(Case 1a)** K_1 is of the form $\mathtt{O}(F1) : F1$ or $\mathtt{C}(G) : G$. This is in $\mathcal{F}, \mathcal{G}$ and we are done with (a) and (b) by maximality of $\mathcal{F}, \mathcal{G}$. **(Case 1b)** K_1 is of the form $\mathtt{B}_\sigma(g) : \sigma(G)$. So $\mathcal{F}, \mathcal{G} \vdash g : G$ by IH is persis-

tant by assumption that builtins are replacement-maximal thus the builtin rule is enabled until it fires and we are done with (a) and (b).

(Case 2) The subderivation K_1 is not atomic. By IH (a) we have $K_1' \in \mathcal{K}(\pi)$ with $K_1 \leq K_1'$ and $\mathcal{F}, \mathcal{G} \vdash K_1'$. Now we need to show that we can find a replacement-maximal such K_1'.

So we have $\mathcal{F}, \mathcal{G} \vdash K_3$, $\mathcal{F}, \mathcal{G} \vdash K_1'$, $K_1' \in \mathcal{K}(\pi)$, $K_1 \leq K_1'$, $K_1 \rhd K_3$, and we have assumed that K_3 is not eventually covered, i.e. $K_3 \not\leq K_3'$ for any $K_3' \in \mathcal{K}(\pi)$.

Suppose K_1' is not replacement-maximal so there is a K_2 such that $K_1' \prec K_2 \in \mathcal{K}(\pi)$. By compatibility, $K_1 \prec K_2$.

Without loss of generality we can assume that K_1' is subsumption-maximal in $\mathcal{K}(\pi)$ (justification, see last paragraph below). Hence, $K_1' \not< K_2$. Furthermore, $K_1 \leq K_1' \prec K_2$ yields $K_1 \not< K_2$ by ordering consistency (between subsumption and replacement).

Using $K_1 \prec K_2$ and $K_1 \not< K_2$, Condition 2 (replacement restrictions) tells us that $K_1 \not\rhd^+ K_2$ implies that there exists atomic K_1', K_2' such that $K_1' \rhd^* K_1$, $K_2' \rhd^* K_2$, $K_1' \prec K_2'$ which contradicts maximality of $\mathcal{F}, \mathcal{G}$.

Thus we only need to deal with the case that $K_1 \rhd^+ K_2$. Since K_1 is an *immediate* subderivation of K_3 we cannot have $K_2 \rhd^+ K_3$. Also we cannot have $K_3 \rhd^+ K_2$, because this would imply $K_3 \in \mathcal{K}(\pi)$.

Condition 3 (replacement restrictions) tells us that $K_3 \leq K_2$. But $K_3 \leq K_2$ contradicts our assumption that K_3 is not eventually covered.

It remains to justify our assumption that K_1' can be choosen subsumption-maximal such that $\mathcal{F}, \mathcal{G} \vdash K_1'$. By upward well-foundedness of subsumption we can certainly find a K_1' that is subsumption maximal in $\mathcal{K}(\pi)$. Now assume that $\mathcal{F}, \mathcal{G} \vdash K_1'$ does not hold. This clearly implies that $K_1' \neq K_1$, hence $K_1 < K_1'$. Then there is an atomic K_1'' in $\mathcal{K}(\pi)$ with $K_1'' \rhd^* K_1'$. By replacement-maximality of $\mathcal{F}, \mathcal{G}$ there must be a $K'' \in \mathcal{F}, \mathcal{G}$ such that $K_1'' \prec K''$. Using $K_1 < K_1'$ and Condition 4 (replacement restrictions) we obtain a contrdiction. □

Theorem 4 (Termination). *Let $\pi = c_0, r_1, c_1, \ldots, r_k, c_k, \ldots$ be an eventually closed execution. Let $\mathcal{F} = \mathcal{F}^O(\pi)$ and $\mathcal{G} = \mathcal{G}^C(\pi)$, be the observation facts and control goals of π, respectively. If $\mathcal{F}, \mathcal{G}$ has the finite closure property. then π is terminating.*

In the following we assume the premisses of the Termination theorem and let $[\mathcal{K}, \lessdot]$ be the derived atoms and transitive reduction of the partial ordering defined inductively from $\mathcal{F}, \mathcal{G}$ by clauses (0)–(4) of the finite closure property requirement. In fact, $\lessdot$ is defined by omitting the transitive closure part of the definition, and $\leq$ is the transitive reflexive closure of $\lessdot$. Then the finite closure property implies there are no unbounded descending chains $\ldots, K_{n+1} \lessdot K_n \ldots \lessdot K_0$.

We begin the Termination proof with some basic properties.

Lemma 6. $\mathcal{F}(\pi) \cup \mathcal{G}(\pi) \subseteq \mathcal{K}$

Proof. Show $\mathcal{F}(\pi_{|n}) \cup \mathcal{G}(\pi_{|n}) \subseteq \mathcal{K}$ by induction on n. For $n = 0$ there are no facts or goal in the initial configuration. For $n > 0$ consider cases on rule r that introduces a new fact or goal. Use the closure conditions on $\mathcal{K}$ and the fact that $K \in \mathcal{K}$ implies $\mathcal{K} \vdash K$. If r is Observation or Control, the new fact or goal is in $\mathcal{F}, \mathcal{G} \subset \mathcal{K}$ by definition. If r is Builtin use clause (0), if r is Forward1 use clause (1), if r is Forward2 use clause (2), if r is Backward1 use clause (3), if r is Backward2 use clause (4).

Lemma 7. *If $K \lessdot K'$ then $K' \rhd^{+} K$.*

Proof. By inspection of clauses defining $\lessdot$

Lemma 8. $(\forall i)(\exists j > i)(\mathcal{F}(\pi_{|i}) \subset \mathcal{F}(\pi_{|j}))$

Proof. Since π is non-terminating, by freshness requirements eventually a new fact must be introduced.

Proof (Termination). We define the depth of a derived atom $d(K)$ such that

(1) if π has a derived atom of depth d, there is a $\lessdot$ chain in π of length d (from the derived atom to a root atom of depth 0), and
(2) the set of derived atoms of depth d in π is finite.

Now by lemma 8, for any depth d, there is some i such that $\mathcal{F}(\pi_{|i}) \cup \mathcal{G}(\pi_{|i})$ has an element of depth greater than d. Thus a non-terminating execution must have an unbounded decreasing $\lessdot$ chain. □

[1]

Definition 5 (Depth K). *The depth $d(K)$ of $K \in \mathcal{F}(\pi) \cup \mathcal{G}(\pi)$ is defined according to the rule that introduces K. Since a derivation f or g uniquely determines the derived atom we will often use the derivation to stand for the derived atom.*
Goals.

(gc) $d(C(G) : G) = 0$ *where* $c(G) : G \in \mathcal{G}^{C}(\pi)$
(gf) $d(l^{-}() : P_1) = 0$ *where* $l : P_1 \dots P_n \Rightarrow Q \in \Omega_f$
(gf+) $K = l^{-}_{\sigma}(f_1 ... f_{j-1}) : \sigma(P_j)$, $d(K) = 1 + max_{1 \leq i < j} d(f_i)$ *where* $l : P_1 \dots P_n \Rightarrow Q \in \Omega_f$, $k \geq j > 1$.
- *Since $K \lessdot f_i$ for $1 \leq i < j$ there is a $\lessdot$ chain from K through some f_i to a root of length $d(K)$.*

(gb+) $K = l^{-}_{\sigma}(f_1 ... f_{j-1}; g) : \sigma(P_j)$ $d(K) = 1 + max(max_{1 \leq i < j} d(f_i), d(g))$ *where* $l : P_1 \dots P_n \Rightarrow Q \in \Omega_b$,
- *Since $K \lessdot f_i$ for $1 \leq i < j$ and $K \lessdot g$ there is a $\lessdot$ chain from K to a root of length $d(K)$.*

Facts.

(fo) $d(O(F) : F) = 0$ *where* $O(F) : F \in \mathcal{F}^{O}(\pi)$.

[1] CAVEAT – to get some proof done, I am going to use 'and' rather than 'or' in clause (4) of the definition of $[\mathcal{K}, \lessdot]$. Later I will see if this can be relaxed.

(fbi) $K = B_\sigma(g) : \sigma(G)$, $d(K) = 1 + d(g)$ *where G is a builtin, $g : G$ a derived goal and* $\vdash \sigma(G)$.

- *Since $K \lessdot g$ there is a $\lessdot$ chain of length $d(K)$ from K to a root.*

(ff0) $d(l() : Q) = 0$ *where* $l : \Rightarrow Q \in \Omega_f$

(ff+) $K = l_\sigma(f_1, \ldots, f_k) : \sigma(Q)$, $d(K) = 1 + max_{1 \leq i < k} d(f_i)$ *where* $l : P_1 ... P_k \Rightarrow Q \in \Omega_f$, $k > 0$,

- *Since $K \lessdot f_i$ for $1 \leq i \leq k$ there is a $\lessdot$ chain from K through f_i to a root of length $d(K)$.*

(fb0) $d(l(g) : Q) = d(g : G)$ *where* $l : \Rightarrow Q \in \Omega_g$ $g : G$ *a derived goal,* $\sigma(G) = \sigma(Q)$.

(fb+) $d(K) = l_\sigma(f_1, \ldots, f_k; g) : \sigma(Q)$, $d(K) = 1 + max_{1 \leq i < k} d(f_i)$ *where* $l : P_1 ... P_k \Rightarrow Q \in \Omega_b$, $k > 0$, $g : G$ *a derived goal,* $\sigma(G) = \sigma(Q)$.

- *Since $K \lessdot f_i$ for $1 \leq i \leq k$ there is a $\lessdot$ chain from K through f_i to a root of length $d(K)$.*

Now we define depth n sets of facts and goals and show they are finite by induction on n.

Lemma 9 (Depth n knowledge is finite).
The argument for finiteness is interleaved with the definition of the depth n knowledge sets.
Goals GD(n).

$GD(0) = \quad \{G \mid (\exists g) g : GinCG\}$

control– finite by eventually closed

$\cup$

$\{P1 \mid (\exists l)(l : P_1 \ldots P_n \Rightarrow : Q \in \Omega_f)\}$

fwd2,j $= 1$*– finite since Ω has finitely many clauses*

$GD(n > 0) = \{\sigma(P_j) \mid (\exists l, f_1 \ldots f_{j-1} \in \mathcal{F}(\pi))(l : P_1 \ldots P_k \Rightarrow Q \in \Omega_f,$
$k \geq j > 1, d(f_i : \sigma(P_i))) < n\}$

fwd2, $j > 1$–finite because σ is determined by $f_1..f_{j-1}$

by max σ and finiteness of $FD(< n)$

$\cup$

$\{\sigma(P_j) \mid (\exists l, f_1 \ldots f_{j-1} \in \mathcal{F}(\pi), g \in \mathcal{G}(\pi))$
$(l : P_1 \ldots P_k \Rightarrow Q \in \Omega_b,$
$d(f_i : \sigma(P_i)) < n, d(g : G) < n,$
σ mgu extending $\sigma / vars(P_1, \ldots, P_{j-1})$ *s.t.* $\sigma(Q) = \sigma(G))\}$

bwd2– $\sigma = \sigma_1 * \sigma_0$, $\sigma_0 = \sigma / vars(P_1 ... P_{j-1})$ *and* $\sigma_1 = mgu(\sigma_0(Q), G)$

[taking vars G fresh], finite by finiteness of $FD(< n)$ and $GD(< n)$

and σ determined by the choice of f_i and g.

Facts FD(n).

$$FD(0) = \{F \mid O(F) : F \in \mathcal{F}^O\}$$

observation– finite by eventually closed

$$\cup$$

$$\{Q \mid (\exists l)(l : \Rightarrow Q \in \Omega_f)\}$$

fwd1, $k = 0$– finite since Ω has finitely many clauses and Q ground by variable restriction

$$\cup$$

$$\{\sigma(Q) \mid (\exists l, g \in \mathcal{G}(\pi))(l : \Rightarrow Q \in \Omega_b, d(g : G) = 0, \sigma(Q) = \sigma(G))\}$$

bwd1, $k = 0$– finite Ω, finite $GD(< n)$ means finite l, g pairs.

$\sigma = mgu(Q, G)$ since by variable constraint, unifier is ground

$$FD(n > 0) = \{\sigma(G) \mid g : G \in \mathcal{G}(\pi), G \text{ builtin}, \vdash \sigma(G), d(g : G) < n\}$$

builtin– finite solutions

$$\cup$$

$$\{\sigma(Q) \mid (\exists l, f_1 \dots f_k \in \mathcal{F}(\pi))(l : P_1 \dots P_k \Rightarrow Q \in \Omega_f, k > 0, dom(\sigma) = vars(P_1, \dots, P_k), d(f_i : \sigma(P_i)) < n,\ 1 \leq i \leq k)\}$$

fwd1, $k > 0$– finite Ω, finite $FD(< n)$ means finite $l, f_1, ..., f_k$ tuples

$$\cup$$

$$\{\sigma(Q) \mid (\exists l, g \in \mathcal{G}(\pi))(l : \Rightarrow Q \in \Omega_b, \sigma = mgu(Q, G), d(g : G) \leq n)\}$$

bwd1, $k = 0$– finite Ω, finite $GD(\leq n)$ means finite l, g pairs,

$\sigma ismgu(Q, G)$ since by variable constraint, unifier is ground

$$\cup$$

$$\{\sigma(Q) \mid (\exists l, f_1 \dots f_k \in \mathcal{F}(\pi), g \in \mathcal{G}(\pi))(l : P_1 \dots P_k \Rightarrow Q \in \Omega_b, k > 0, d(g : G) < n, d(f_i : P_i) < n\ 1 \leq i \leq k, \sigma = mgu(Q, G))\}$$

bwd1, $k > 0$– σ a mgu by variable constraint, finite Ω, finite $FD(< n), GD(< n)$ means finite $l, f_1, ..., f_k, g$ tuples