Minimality, Non-Determinism, and Absent Information in Multi-Context Systems

Floris Roelofsen
Institute for Logic, Language, and Computation
Amsterdam, Netherlands

February 1, 2005

Abstract

Multi-context systems (MCS) can be used to represent contextual information flow. We show that the semantics of an MCS is completely determined by the information that is obtained when simulating the MCS, in such a way that a *minimal* amount of information is deduced at each step of the simulation.

The MCS framework implicitly presupposes that information flow is *deterministic*. In many natural situations, this is not a valid assumption. We propose an extension of the framework to account for non-determinism and provide an algorithm to efficiently compute the meaning of non-deterministic systems.

In MCS, the acquisition of new information is based on the *presence* of other information only. We give a generalized account to model situations in which information is obtained as a result of the *absence* of other information.

1 Introduction

The representation of contextual information and inter-contextual information flow has been formalized in several ways. Most notable are the propositional logic of context developed by McCarthy, Buvač and Mason [9, 10], and the multi-context systems devised by Giunchiglia and Serafini [7, 8], which later have been associated with the local model semantics introduced by Giunchiglia and Ghidini [6]. Serafini and Bouquet [13] have argued from a technical point of view that multi-context systems constitute the most general formal framework. This conclusion is supported by a more conceptual argument of Benerecetti et.al. [2].

A multi-context system describes the information available in a number of contexts (i.e., to a number of people / agents / databases, etc.) and specifies the information flow between those contexts. The local model semantics defines a system to entail a certain piece of information in a certain context, if and only if that piece of information is acquired in that context, independently of how the information flow described by the system is accomplished.

The first contribution of this paper is based on the observation that the local model semantics of a multi-context system is completely determined by the information that is obtained when simulating the information flow specified by the system, in such a way that a *minimal* amount of information is deduced at each step of the simulation. We define an operator which suitably implements such a simulation, and thus determines the information entailed by the system. This operator constitutes a first constructive account of the local model semantics.

The second contribution of this paper is based on the observation that, in its original formulation, the multi-context system framework implicitly rests on the assumption that information flow is *deterministic*. In many situations, this is not a suitable assumption. In a multi-agent scenario, for example, upon establishing a certain piece of information, an agent may decide to pass this information on to either one of a group of other agents. His choice as to which agent he will inform could be made non-deterministically. Another typical situation in which information flow is inherently non-deterministic is when the information channels between different contexts are subject to temporary failure or unavailability. Consider the case of online repositories. If information is obtained in one repository, the protocol may be to pass this information on to any one of a number of associated "mirror repositories": if the communication channel with one of these is defective or temporarily unavailable, another one is tried, until at least one successful communication is established.

The local model semantics can easily be adapted to account for non-deterministic systems. However, if a system describes a non-deterministic information flow, then the minimal information entailed by the system cannot be determined unequivocally. We provide a way to generate from a non-deterministic system a number of deterministic systems, the semantics of which can be determined constructively, and which, together, completely determine the semantics of the original non-deterministic system.

The third contribution of this paper is based on the observation that in multi-context systems, new information is derived based on the *presence* of other information only. However, in many natural situations (concrete

examples will be given below), new information is obtained due to a *lack* of other information. We propose a generalized framework so as to account for such situations. Non-monotonic reasoning techniques are applied to formulate a suitable semantics for this framework.

We proceed, in section 2, with a brief review of multi-context system syntax and local model semantics. Minimality, non-determinism, and absent information are discussed in section 3, 4, and 5, respectively. We conclude, in section 6, with a concise recapitulation of our main observations and results.

2 Preliminaries

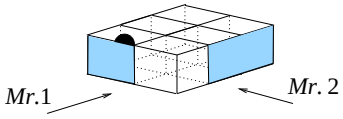

Figure 1: A magic box.

A simple illustration of the main intuitions underlying the multi-context system framework is provided by the situation depicted in figure 1. Two agents, Mr.1 and Mr.2, are looking at a box from different angles. The box is called magic, because neither Mr.1 nor Mr.2 can make out its depth. As some sections of the box are out of sight, both agents have partial information about the box. To express this information, Mr.1 only uses proposition letters l (there is a ball on the left) and r (there is a ball on the right), while Mr.2 also uses a third proposition letter c (there is a ball in the center).

In general, we consider a set of contexts I, and a language L_i for each context $i \in I$. Henceforward, we assume I and $\{L_i\}_{i \in I}$ to be fixed, unless specified otherwise. Moreover, for the purpose of this paper we assume each L_i to be built over a finite set of proposition letters, using standard propositional connectives.

To state that the information expressed by a formula $\varphi \in L_i$ is established in context i we use so-called *labeled formulas* of the form $i : \varphi$ (if no ambiguity arises, we simply refer to labeled formulas as formulas, and we even use capital letters F, G, and H to denote labeled formulas, if the context label is irrelevant). A *rule* r is an expression of the form:

$$F \leftarrow G_1 \wedge \ldots \wedge G_n \tag{1}$$

where F and all G's are labeled formulas; F is called the consequence of r and is denoted by $cons(r)$; all G's are called premises of r and together make up the set $prem(r)$. Rules without premises are called *facts*. Rules with at least one premiss are called *bridge rules*. A *multi-context system* (system hereafter) is a finite set of rules. A fact describes information that is established in a certain context, independent of which information is obtained in other contexts. A bridge rule specifies which information is established in one context, if other pieces of information are obtained in different contexts. So a system can be seen as a specification of contextual information available a priori plus an inter-contextual information flow.

Example 1 *The situation in figure 1 is modeled by the following system S:*

$$\begin{array}{lll} 1 : \neg r & \leftarrow & \\ 2 : l & \leftarrow & \\ 1 : l \vee r & \leftarrow & 2 : l \vee c \vee r \\ 2 : l \vee c \vee r & \leftarrow & 1 : l \vee r \end{array}$$

Mr.1 knows that there is no ball on the right, Mr.2 knows that there is a ball on the left, and if any agent gets to knows that there is a ball in the box, then he will inform the other agent about it.

A classical interpretations m of language L_i is called a *local model* of context i. A set of local models is called a *local information state*. Intuitively, every local model in a local information state represents a "possible state of affairs". If a local information state contains exactly one local model, then it represents complete information. If it contains more than one local model, then it represents partial information: more than one state of affairs is considered possible. A *distributed information state* is a set of local information states, one for each context. In conformity with the literature, we will refer to distributed information states as *chains*.

Example 2 *The situation in figure 1, in which Mr.1 knows that there is no ball on the right but does not know whether there is a ball on the left, is represented by a chain whose first component $\{\{l, \neg r\}, \{\neg l, \neg r\}\}$ contains two local models. As such, the chain reflects Mr.1's uncertainty about the left section of the box.*

A chain c *satisfies* a labeled formula $i : \varphi$ (denoted $c \models i : \varphi$) if and only if all local models in its i^{th} component classically satisfy φ. A rule r is *applicable* with respect to a chain c if and only if c satisfies every premiss of r. Notice that facts are applicable with respect to any chain. A chain c

complies with a rule r, if and only if, whenever r is applicable with respect to c, then c satisfies r's consequence. We call c a *solution chain* of a system S if and only if it complies with every rule in S. A formula F is *true* in S (denoted $S \models F$) if and only if every solution chain of S satisfies F.

For convenience, we introduce some auxiliary terminology and notation. Let $\mathbf{C}$ denote the set of all chains. Notice that, as each L_i is assumed to be built over a finite set of proposition letters, $\mathbf{C}$ is assumed to be finite. Let $c^{\perp}$ denote the chain containing every local model of every context ($c^{\perp}$ does not satisfy any non-tautological expression); let $c^{\top}$ denote the chain containing no local models at all ($c^{\top}$ satisfies all expressions). If C is a set of chains, then the component-wise union (intersection) of C is the chain, whose i^{th} component consists of all local models that are in the i^{th} component of some (every) chain in C. If c and c' are chains, then $c \setminus c'$ denotes the chain, whose i^{th} component consists of all local models that are in c_i but not in c'_i. Finally, let us sometimes say that a local model m is (not) in c, when we actually mean that m is (not) in some (any) component c_i of c.

3 Minimality

We order chains according to the amount of information they convey. Intuitively, the more local models a chain component contains, the more possibilities it permits, so the less informative it is. Formally, we say that c is *less informative* than c' ($c \preceq c'$), if for every i we have $c_i \supseteq c'_i$. If, moreover, for at least one i we have $c_i \supset c'_i$, then we say that c is *strictly less informative* than c' ($c \prec c'$).

Lemma 1 *Let C be a set of chains. Let c^u (c^i) denote the component-wise union (intersection) of all chains in C. Then c^u (c^i) is less (more) informative than any chain in C.*

Proof. We proof the *union* part. Let c' be a chain in C. Then for every i, every local model m in c'_i is also in c^u_i. So $c^u_i \supseteq c'_i$, and thus $c^u \preceq c'$. $\square$

Lemma 2 *$(\boldsymbol{C}, \preceq)$ forms a complete lattice.*

Proof. We should prove that every finite subset of $\mathbf{C}$ has both a greatest lower bound and a least upper bound in $\mathbf{C}$. Let C be a subset of $\mathbf{C}$ (note that

C is finite, so C must be finite as well). Let c^u (c^i) denote the component-wise union (intersection) of all chains in C. Then, by lemma 1, c^u is a lower bound of C. Now consider a chain c', such that $c^u \prec c'$. For this to be the case, there must be a local model m, which is in c^u but not in c'. But then m must also be in some chain c^m in C, which makes $c' \preceq c^m$ impossible. So c' cannot be a lower bound of C, which implies that c^u is the greatest lower bound of C. Analogously, it is shown that c^i is least upper bound of C. □

Note that $c^\perp$ is strictly less informative than any other chain, whereas $c^\top$ is strictly more informative than any other chain. If $c \preceq c'$ we say that c' is an *extension* of c. So, intuitively, extending c corresponds to *adding* information to it. More technically, to extend c is to *remove* local models from it. We say that c is *minimal* among a set of chains C, if c is in C and no other chain c' in C is strictly less informative than c. In particular, we say that c is a *minimal solution chain* of a system S, if it is minimal among the set of all solution chains of S.

Lemma 3 *Let c and c' be two chains, such that $c \preceq c'$. Then any formula that is satisfied by c is also satisfied by c'.*

Proof. Suppose $c \models i : \phi$. Then, per definition, $m \models \phi$ for every $m \in c_i$. As c'_i is contained in c_i, we also have $m' \models \phi$ for every $m' \in c'_i$. So $c' \models i : \phi$. □

Lemma 4 *Let C be a set of chains and let c^u denote the component-wise union of all chains in C. Then a formula is satisfied by c^u if and only if it is satisfied by every chain in C.*

Proof.
($\Rightarrow$) Follows directly from lemma 1 and lemma 3.
($\Leftarrow$) Suppose all chains in C satisfy $i : \phi$. Then all local models in the i^{th} component of every chain in C must satisfy ϕ. These are exactly the local models that make up the i^{th} component of c^u. So c^u also satisfies $i : \phi$. □

Lemma 5 *Let S be a system. Then the set of all solution chains of S is closed under component-wise union. That is, if C is a set of solution chains of S, then the component-wise union c^u of all chains in C is again a solution chain of S.*

Proof. Let C be a set of solution chains of S. Let c^u be the component-wise union of C. Let r be an arbitrary rule in S. Then all c' in C comply with r. Suppose, towards a contradiction, that c^u does not comply with r, i.e., c^u satisfies all of r's premises, but does not satisfy r's consequence. By lemma 4 all c' in C satisfy all of r's premises, and therefore, by assumption, they all satisfy r's consequence as well. But then, again by lemma 4, c^u must also satisfy r's consequence, which contradicts the assumption that c^u does not comply with r. So c^u must comply with r, and as r was arbitrary, c^u must be a solution chain of S. □

Theorem 1 *Every system S has a unique minimal solution chain c_S.*

Proof. Every system has at least one solution chain, namely $c^\top$. Now, let S be a system and let C_S be the set of all its solution chains. Then, by lemma 5, the component-wise union c_S of C_S is itself in C_S. Moreover, by lemma 1, c_S is less informative than any other chain in C_S. So c_S is minimal among C_S and, moreover, any chain c' in C_S which is minimal among C_S, must be equal to c_S. In other words, c_S is the unique minimal solution chain of S. □

Theorem 2 *The meaning of a system S is completely determined by its unique minimal solution chain c_S. For any formula F we have:*

$$S \models F \quad \Leftrightarrow \quad c_S \models F$$

Proof. Let S be a system and let F be a formula. Then F is true in S if and only if F is satisfied by all solution chains of S. By lemma 4, this is the case if and only if F is satisfied by the component-wise union of all solution chains of S. By the proof of theorem (1) this union constitutes the minimal solution chain c_S of S. □

Theorem (1) and (2) are extremely useful, because they establish that, to answer queries about a system S, it is no longer necessary to compute all solution chains of S; we only need to consider the system's minimal solution chain c_S.

3.1 Computing the Minimal Solution Chain

Recall that a system S can be thought of as a specification of inter-contextual information flow. It turns out that the minimal solution chain of S can be characterized as the $\preceq$-least fixpoint of an operator $\mathbf{T}_S$, which, intuitively, simulates the information flow specified by S.

Let $S^*(c)$ denote the set of rules in S that are applicable w.r.t. c. Then:

$$\mathbf{T}_S(c) = c \setminus \{m \mid \exists r \in S^*(c) : m \not\models \mathit{cons}(r)\} \tag{2}$$

For every rule r in S that is applicable w.r.t. c, $\mathbf{T}_S$ removes from c all local models that do not satisfy $\mathit{cons}(r)$. Intuitively, this corresponds to augmenting c with the information expressed by $\mathit{cons}(r)$. In this sense, $\mathbf{T}_S$ simulates the information flow described by S. Clearly, $\mathbf{T}_S(c)$ is obtained from c only by *removing* local models from it. As a result, $\mathbf{T}_S(c)$ is always more informative than c.

Lemma 6 *For every chain c and every system S: $c \preceq \boldsymbol{T}_S(c)$.* □

We now prove that, starting with the least informative chain $c^\perp$, $\mathbf{T}_S$ will reach its $\preceq$-least fixpoint after finitely many iterations, and that this $\preceq$-least fixpoint coincides with the minimal solution chain of S. The first result is typically established using Tarski's fixpoint theorem [14]. In order to apply this theorem, we first need to show that $\mathbf{T}_S$ is monotone and continuous with respect to $\preceq$.

Lemma 7 *$\boldsymbol{T}_S$ is monotone with respect to $\preceq$.*

Proof. Let c and c' be any two chains such that $c \preceq c'$. We need to prove that $\mathbf{T}_S(c) \preceq \mathbf{T}_S(c')$. Suppose, towards a contradiction that this is not the case. Then there is a local model m that belongs to $\mathbf{T}_S(c')$ but not to $\mathbf{T}_S(c)$. Clearly, m must already be present in c', and therefore also in c. From the fact that m has been removed from c by $\mathbf{T}_S$ it follows that there must be a rule r in S such that c satisfies $\mathit{prem}(r)$, whereas m does not satisfy $\mathit{cons}(r)$. But then, by lemma 3, c' must also satisfy $\mathit{prem}(r)$, so $\mathbf{T}_S$ should have removed m from c' as well. We conclude that $\mathbf{T}_S(c) \preceq \mathbf{T}_S(c')$. So $\mathbf{T}_S$ is monotone with respect to $\preceq$. □

Lemma 8 *$\boldsymbol{T}_S$ is continuous with respect to $\preceq$.*

Proof. Let $c^0 \preceq c^1 \preceq c^2 \preceq \ldots$ be an infinite sequence of chains, each of which contains more information than all preceding ones. We need to prove that $\mathbf{T}_S(\bigcup_{n=0}^{\infty} c^n) = \bigcup_{n=0}^{\infty} \mathbf{T}_S(c^n)$. As $\mathbf{C}$ is finite, $\{c^0, c^1, c^2, \ldots\}$ must have a maximum c^m in $\mathbf{C}$. So $\mathbf{T}_S(\bigcup_{n=0}^{\infty} c^n) = \mathbf{T}_S(c^m) = \bigcup_{n=0}^{\infty} \mathbf{T}_S(c^n)$. □

Theorem 3 *$\boldsymbol{T}_S$ has a $\preceq$-least fixpoint, which is obtained after a finite number of consecutive applications of $\boldsymbol{T}_S$ to $c^\perp$.*

Proof. Follows from lemmas 2, 7, and 8 by Tarski's fixpoint theorem [14]. □

Lemma 9 *Let c be a chain and let S be a system. Then c is a fixpoint of $\boldsymbol{T}_S$ if and only if c is a solution chain of S.*

Proof. A chain c is a fixpoint of $\mathbf{T}_S$ if and only if for every rule r in S, c satisfies $cons(r)$ whenever c satisfies $prem(r)$. This is the case if and only if c is a solution chain of S. □

Theorem 4 *Let S be a system. Then the minimal solution chain c_S of S coincides with the $\preceq$-least fixpoint of $\boldsymbol{T}_S$.*

Proof. Follows directly from lemma 9. □

From theorems 3 and 4 we conclude that the minimal solution chain c_S of a system S is obtained by a finite number of applications of $\mathbf{T}_S$ to the least informative chain $c^\perp$. But we can even prove a slightly stronger result:

Theorem 5 *Let S be a system and let $|S|$ denote the number of* bridge rules *in S. Then the minimal solution chain c_S of S is obtained by at most $|S| + 1$ consecutive applications of $\boldsymbol{T}_S$ to $c^\perp$.*

Proof. Let c be a chain and let S be a system. Notice that $\mathbf{T}_S(c)$ is a fixpoint of $\mathbf{T}_S$ if and only if $S^*(\mathbf{T}_S(c))$ coincides with $S^*(c)$. Lemmas 3 and 6 imply that, in any case, $S^*(\mathbf{T}_S(c)) \supseteq S^*(c)$. In other words, during each iteration of $\mathbf{T}_S$ some (possibly zero) rules are added to S^*. In the case that S^* remains unaltered, $\mathbf{T}_S$ must have reached a fixpoint. Now we observe that during the first application of $\mathbf{T}_S$ (to $c^\perp$) all facts in S are added to S^*.

Clearly, after that, $\mathbf{T}_S$ can be applied at most $|S|$ times before a fixpoint is reached. □

In fact, a slightly more involved, but essentially equivalent procedure was introduced for rather different reasons in [11]. This procedure was shown to have worst-case time complexity $O(|S|^2 \times 2^M)$, where M is the maximum number of propositional variables in either one of the contexts involved in S. The greater part of a typical computation is taken up by propositional reasoning within individual contexts, which itself requires exponential time in the worst case.

Example 3 *Consider the system S given in example 1. Applying $\boldsymbol{T}_S$ to $c^\perp$ establishes the facts given by the first two rules of the system. But then Mr.2 knows that there is a ball in the box, so the next application of $\boldsymbol{T}_S$ simulates the information flow specified by the third rule of the system: Mr.2 informs Mr.1 of the presence of the ball. The resulting chain is left unaltered by any further application of $\boldsymbol{T}_S$, and therefore constitutes the minimal solution chain of S. The fact that this chain satisfies the formula $1 : l$ reflects, as desired, that Mr.1 has come to know that there is a ball in the left section of the box.*

4 Non-Determinism

The original formulation of multi-context systems implicitly rests on the assumption that information flow is *deterministic*. However, there are many natural situations in which information flow is inherently non-deterministic.

Example 4 *Adriano is on holiday after having submitted his final school exams. He has promised to call his father or his mother in case his teacher lets him know that he has passed his exams. This situation can be modeled by a system S consisting of the following rule:*

$$m : p \ \textbf{\textit{or}} \ f : p \leftarrow a : p$$

Notice that Adriano may be conceived of as an agent in a multi-agent system, who non-deterministically decides which other agents to inform when acquiring novel information. Alternatively, Adriano's parents may be conceived of as mirror repositories of information about Adriano's well-being (assuming that they tell each other everything they come to know about Adriano). Typical telephonic connections may be broken or temporarily

unavailable. Analogous to the situation sketched in the introduction, Adriano will try to reach his parents, until at least one of them is informed. In general, we would like to consider systems in which rules r are of the form:

$$F_1 \textbf{ or } \ldots \textbf{ or } F_m \leftarrow G_1 \wedge \ldots \wedge G_n \tag{3}$$

where all F's and G's are labeled formulas; all F's are called consequences of r and together form the set $cons(r)$; and as before, all G's are called premises of r and together constitute the set $prem(r)$. A rule doesn't necessarily have any premises ($n \geq 0$), but always has at least one consequence ($m \geq 1$). We call a rule deterministic if it has only one consequence, and non-deterministic otherwise. We call finite sets of possibly non-deterministic rules *non-deterministic multi-context systems* (non-deterministic systems for short). Systems which consist of deterministic rules only, are from now on referred to as deterministic systems.

A chain c complies with a non-deterministic rule r if and only if, whenever r is applicable w.r.t. c, *at least one of* its consequences is satisfied by c. A chain is a solution chain of S if and only if it complies with all rules in S. A formula F is true in S, $S \models F$, if and only if F is satisfied by all solution chains of S.

Observation 1 *Let S be a non-deterministic system, let c' and c'' be two solution chains of S, and let c be the component-wise union of c' and c''. Then it is* not *generally the case that c is again a solution chain of S. Therefore, S does not generally have a unique minimal solution chain.*

Example 5 *Suppose Adriano's teacher lets him know that he passed his exams. The resulting system S is given by the following rules:*

$$\begin{aligned} a:p &\leftarrow \\ m:p \textbf{ or } f:p &\leftarrow a:p \end{aligned}$$

This system has two minimal solution chains:

$$\begin{aligned} c^m &= \{\{p\}_m, \quad \{p, \neg p\}_f, \quad \{p\}_a\} \\ c^f &= \{\{p, \neg p\}_m, \quad \{p\}_f, \quad \{p\}_a\} \end{aligned}$$

whose component-wise union $\{\{p, \neg p\}_m, \quad \{p, \neg p\}_f, \quad \{p\}_a\}$ is not a solution chain of S.

Theorem 6 *The meaning of a non-deterministic system S is completely determined by the set C_S of all its minimal solution chains. For any formula F we have:*

$$S \models F \quad \Leftrightarrow \quad \forall c \in C_S : c \models F$$

Proof. Let S be a non-deterministic system and let F be a formula. Then F is true in S if and only if F is satisfied by every solution chain of S. Clearly, if F is satisfied by every solution chain of S, then it must in particular be satisfied by every minimal solution chain of S. Moreover, every solution chain of S is an extension of some minimal solution chain of S, which implies, by lemma 3, that F is satisfied by all minimal solution chains of S only if F is satisfied by all solution chains of S. □

Theorem 6 establishes that the meaning of a non-deterministic system S is completely determined by the set C_S of all its minimal solution chains. We will now provide a way to compute C_S, re-using the method outlined in section 3.

4.1 Computing Minimal Solution Chains

Inspired by an idea originally developed for disjunctive databases [12], we generate from a non-deterministic system S a number of deterministic systems S_1, S_2, ..., S_n, in such a way that the minimal solution chains of S are among the minimal solution chains of S_1, S_2, ..., S_n (note that each S_i has a unique minimal solution chain which can be computed as outlined in section 3). Hereto, we introduce the notion of a *generated system*. Let S be a non-deterministic system and let r be a rule in S. Then we say that a deterministic rule r' is *generated by* r if and only if $cons(r') \in cons(r)$ and $prem(r') = prem(r)$. We say that a system S' is *generated by* S if and only if it is obtained from S by replacing each rule r in S by some rule r' generated by r. Notice that, indeed, a generated system is always deterministic, and that any non-deterministic system S generates at most $\prod_{r \in S} |cons(r)|$ different deterministic systems.

Example 6 *The non-deterministic system from example 5 generates two deterministic systems:* $\{a : p \leftarrow, m : p \leftarrow a : p\}$ *and* $\{a : p \leftarrow, f : p \leftarrow a : p\}$. *The only system generated by a deterministic system is that system itself.*

Lemma 10 *A chain c is a solution chain of a non-deterministic system S if and only if it is a solution chain of some system S' generated by S.*

Proof.
($\Rightarrow$) Suppose c is a solution chain of S. Then c complies with every rule in S. For every rule r in S, if c complies with r, then there must be a rule r' generated by r such that c complies with r' as well. Let S' be the system $\{r' \mid r \in S\}$. Then c is a solution chain of S'.

($\Leftarrow$) Suppose c is a solution chain of a system S' generated by S. Then c complies with every rule in S'. Every rule r in S has generated some rule r' in S', and clearly, if c complies with r' then it must also comply with r. So c is a solution chain of S. □

We call a chain c a *potential solution chain* of S if and only if c is a minimal solution chain of some system S' generated by S.

Lemma 11 *Every minimal solution chain of a system S is also a potential solution chain of S.*

Proof. Suppose c is a minimal solution chain of S. Then, by lemma 10, c is a solution chain of some system S' generated by S. Let c' be the minimal solution chain of S'. Then c must be an extension of c'. By lemma 10 c' must be a solution chain of S. But then, as c is a minimal solution chain of S, c' must be equal to c. So c is a minimal solution chain of S', and therefore a potential solution chain of S. □

Observation 2 *It is* not *generally the case that a potential solution chain of S is also a minimal solution chain of S.*

Example 7 *Suppose Adriano's teacher also called Adriano's mother to tell her the good news. The resulting system S is given by the following rules:*

$$\begin{aligned} a : p \quad &\leftarrow \\ m : p \quad &\leftarrow \\ m : p \ \textbf{\textit{or}}\ f : p \quad &\leftarrow \quad a : p \end{aligned}$$

This system has two potential solution chains:

$$\begin{aligned} c^m &= \{\{p\}_m, \quad \{p, \neg p\}_f, \quad \{p\}_a\} \\ c^{mf} &= \{\{p\}_m, \quad \{p\}_f, \quad \{p\}_a\} \end{aligned}$$

But as c^{mf} extends c^m only the latter is a minimal solution chain of S.

We call c an *essential solution chain* of S if and only if c is minimal among all potential solution chains of S.

Theorem 7 *A chain is a minimal solution chain of S if and only if it is an essential solution chain of S.*

Proof.
($\Rightarrow$) Suppose c is a minimal solution chain of S. Then, by lemma 11, c is a potential solution chain of S. If c is minimal among all potential solution chains of S, then, per definition, it is essential. Now, towards a contradiction, suppose that c is *not* minimal among all potential solution chains of S. Then there must be another potential solution chain c' of S, such that $c' \prec c$. But, by lemma 10, c' must also be a solution chain of S, which contradicts the assumption that c is a minimal solution chain of S.
($\Leftarrow$) Suppose c is an essential solution chain of S. Furthermore, towards a contradiction, suppose that c is *not* a minimal solution chain of S. Then there must be a minimal solution chain c' of S, such that $c' \prec c$. By lemma 11, c' is a potential solution chain of S. But this contradicts the assumption that c is minimal among all potential solution chains of S. $\square$

Theorem 7 establishes that, in order to compute the meaning of a non-deterministic system S it suffices to compute the meaning of all deterministic systems generated by S. This can be done re-using the method developed in section 3. Given that S generates at most $\prod_{r \in S} |cons(r)|$ different systems, and that computing the meaning of each of these systems takes at most time $O(|S|^2 \times 2^M)$, we conclude that, in the worst case, computing the meaning of S takes time $O(\prod_{r \in S} |cons(r)| \times |S|^2 \times 2^M)$.

5 Absent Information

Rules of the form (1) only allow us to model a rather restricted kind of information flow, namely one in which new information is established based on the *presence* of other information only. There are many natural situations in which information is obtained as a result of the *absence* of other information. Such situations cannot be modeled by the present formalism.

Example 8 (Coordination) *Let d_1, d_2 be two meteorological databases, which collect their respective data from sensors located in different parts of the country. At the end of the day each database produces a weather forecast based on its own data but also on the information obtained by the other database. For example, d_1 predicts rain, if that follows from its own data and if, moreover, d_2 does not maintain that it won't rain:*

$$1 : r \leftarrow 1 : r \wedge \textbf{not } 2 : \neg r$$

Example 9 (Integration) *Let d_1 and d_2 be as in example 8 and let d_3 be a third database, which integrates the information obtained in d_1 and*

d_2, respectively. Any piece of information that is established by d_1 and not refuted by d_2 (or vice versa) is included in d_3:

$$\begin{aligned} 3:\varphi &\leftarrow 1:\varphi \wedge \textbf{not } 2:\neg\varphi \\ 3:\varphi &\leftarrow 2:\varphi \wedge \textbf{not } 1:\neg\varphi \end{aligned}$$

Example 10 (Trust) *Let d_1, d_2, and d_3 be as in example 9. It would be natural for d_3 to regards d_1 as more trustworthy than d_2 (or vice versa). In this case any piece of information that is established in d_1 is automatically included in d_3, but information obtained in d_2 is only included in d_3 if it is not refuted by d_2:*

$$\begin{aligned} 3:\varphi &\leftarrow 1:\varphi \\ 3:\varphi &\leftarrow 2:\varphi \wedge \textbf{not } 1:\neg\varphi \end{aligned}$$

In general, to model situations in which new information is obtained based on the absence of other information we need rules r of the form[1]:

$$F \leftarrow G_1 \wedge \ldots \wedge G_m \wedge \textbf{not } H_1 \wedge \ldots \wedge \textbf{not } H_n \tag{4}$$

where F, all G's, and all H's are labeled formulas. As before, F is called the consequence of r ($cons(r)$). $G_1, \ldots, G_m$ are called *positive premises* of r and together constitute the set $prem^+(r)$. $H_1, \ldots, H_n$ are called *negative premises* of r and make up the set $prem^-(r)$. A rule does not necessarily have any premises ($m, n \geq 0$). In analogy with commonplace terminology in deductive database and logic programming theory, we call such rules *normal* rules, and finite sets of them *normal multi-context systems* (normal systems for short). If a rule only has positive premises, we call it a *positive* rule. Note that a system, which consists of positive rules only conforms with the original definition of multi-context systems. From now on we call such systems *positive* systems.

Our aim is to generalize the result obtained section 3, i.e. to define the semantics of a normal system S in terms of a single *canonical* chain c_S of S, such that, whenever S is a positive system, c_S coincides with the minimal solution chain of S.

A first naive attempt would be to say that a chain c complies with a normal rule r if and only if it satisfies r's consequence, whenever it satisfies every positive premise of r and does not satisfy any negative premise of r.

[1]For now, we take deterministic systems as a starting point. The results in this section are *not* straightforwardly generalized to the case of non-deterministic systems.

The (minimal) solution chains of a normal system S can then be defined as for positive systems. However, as the following example shows, a normal system does not generally have a unique minimal solution chain, and worse, minimal solution chains of a normal system do not generally correspond with the intended meaning of that system.

Example 11 *Let a system S be given by the following rule:*

$$1 : p \quad \leftarrow \quad \textbf{not } 2 : q$$

Then S has two minimal solution chains:

$$\begin{aligned} c^p &= \{\{p\}, \{q, \neg q\}\} \\ c^q &= \{\{p, \neg p\}, \{q\}\} \end{aligned}$$

Intuitively, S provides no ground for deriving q in context 2. Thus, p should be derived in context 1, and every "proper" canonical chain of S should satisfy $1 : p$. As c^q fails to do so, it should be rejected as such.

But how, then, should the canonical chain of a normal system be characterized?

Extensive research efforts have been involved with an analogous question in the setting of logic programming, when, in the late 80's / early 90's, a proper semantics for normal logic programs was sought. In motivating our characterization of canonical chains for normal multi-context systems, we will recall some important intuitions and adapt some crucial definitions that have resulted from these efforts.

A first desired property of canonical chains, first introduced in the setting of logic programming by Apt, Blair, and Walker [1] and Bidoit and Froidevaux [3], is termed *supportedness*. Intuitively, a chain c is a supported solution chain of a normal system S if and only if, whenever c satisfies a formula F, then S *provides an explanation for why this is so.*

Definition 1 *We call a chain c a supported solution chain of a normal system S if and only if, whenever c satisfies a formula F, then S contains a set R of rules, such that:*

- $\forall r \in R : \begin{cases} \forall G \in prem^+(r) : c \models G \\ \forall H \in prem^-(r) : c \not\models H \end{cases}$
- $\bigcup_{r \in R} cons(r) \models F$

Example 12 *In example 11, as desired, c^p is a supported solution chain of S, while c^q is not. But both c^p and c^q are supported solution chains of the following extension S' of S:*

$$\begin{array}{lcl} 1:p & \leftarrow & \textbf{not } 2:q \\ 2:q & \leftarrow & 2:q \end{array}$$

Intuitively, c^p should be accepted as a canonical chain of S', but c^q should be rejected as such, because the explanation provided by S' for the fact that c^q satisfies $2:q$ is *circular*, i.e., it relies on the very fact that c^q satisfies $2:q$. So, in general, the concept of supportedness does not satisfactorily characterize the canonical chain of a normal system.

The notion of *well-supportedness*, first introduced for logic programs by Fages [4], refines the notion of supportedness to avoid the counter-intuitive result obtained in example 12. Intuitively, a chain c is a well-supported solution chain of a normal system S if and only if, whenever c satisfies a formula F, then S provides a *non-circular* explanation for why this is so.

Fages also proved this notion to be equivalent to the notion of *stability*, which had been defined somewhat earlier by Gelfond and Lifschitz [5]. The results obtained in section 3 pave the way for a straightforward adaptation of the notion of stability to our present setting.

Definition 2 *Let c be a chain and S a normal system. Define:*

$$\begin{array}{lcl} S'(c) & = & \{r \in S \mid \forall H \in prem^-(r) : c \not\models H\} \\ S''(c) & = & pos(S'(c)) \end{array}$$

where $pos(S'(c))$ is obtained from $S'(c)$ by removing all negative premises from its rules. Then, c is a stable solution chain of S, iff it is the unique minimal solution chain of $S''(c)$.

Intuitively, a solution chain c of a system S is stable if, whenever the information represented by c is assumed, then the information flow specified by S reproduces exactly c. Namely, if c is assumed to contain valid information, then any rule in S, one of whose negative premises is satisfied by c, is certainly not applicable. Negative premises which are *not* satisfied by c can be removed from the remaining rules, because they do not have any influence on whether those rules are applicable or not. Thus, S can be reduced to $S''(c)$, and c is stable if and only if it corresponds exactly to the meaning of $S''(c)$, i.e., by Theorem 2, to its minimal solution chain.

Example 13 *In example 12, as desired, c^p is a stable solution chain of S', while c^q is not.*

For many systems, stability suitably characterizes a unique canonical chain. There are still some special cases, however, in which it fails to do so. We give some typical examples.

Example 14 *Both c^p and c^q from example 11 are stable solution chains of the system given by the following rules:*

$$\begin{aligned} 1:p &\leftarrow \textbf{not } 2:q \\ 2:q &\leftarrow \textbf{not } 1:p \end{aligned}$$

Example 15 *The following system does not have any stable solution chains.*

$$1:p \leftarrow \textbf{not } 1:p$$

In both cases we think it is most reasonable to conclude that no information is derived at all, i.e. to regard $c^\perp$ as the proper canonical chain.

Example 16 *The following system does not have any stable solution chains either.*

$$\begin{aligned} 1:p &\leftarrow \textbf{not } 1:p \\ 1:t &\leftarrow \textbf{not } 2:q \\ 2:r &\leftarrow 1:t \end{aligned}$$

Example 16 illustrates that, even if the rest of the system is unproblematic, one single rule (in this case the first one) can cause the system not to have any stable solution chain at all. In this case, t and r should be derived in context 1 and 2, resp.

The *well-founded semantics*, first proposed for logic programs by van Gelder, Ross, and Schlipf [15] avoids the problems encountered in the above examples. The well-founded model of a program is defined as the least fixpoint of an operator, which, given an interpretation, determines the atoms that are necessarily true and those that are necessarily not true with respect to the program and the interpretation. It assigns *true* to the former set of atoms, and $false$ to the latter. As a result, more atoms may become necessarily true or necessarily not true. Corresponding truth values are assigned until a fixpoint is reached. All atoms that have not been assigned a definite truth value, are interpreted as *unknown*.

Our approach shares an important intuition with the well-founded semantics for logic programs, namely, that while constructing the canonical chain of a system, it is not only important to accumulate the information that *can certainly be derived* from the system, but also to keep track of information that *can certainly not be derived* from the system.

But the two approaches are also fundamentally different. The well-founded semantics constructs a 3-valued interpretation I, which is minimal with respect to a *truth order* $\sqsubseteq$ (i.e. $I \sqsubseteq I'$ iff I makes less atoms true and more atoms false than I'), whereas we seek a chain which is minimal with respect to an *information order* $\preceq$ (i.e. $c \preceq c'$ iff c makes less expressions either true or false than c'). This particularly results in a different treatment of expressions that are found *not* to be true. To regard these expressions as false, as the well-founded semantics does, would be to introduce redundant information. Instead, in our setting, such expressions should simply be recorded as not being derivable.

5.1 Constructing the Canonical Chain

The canonical chain of a normal system S, henceforward denoted by c_S, is constructed by an iterative transformation of a datastructure $\langle c, a \rangle$, where:

- c is the "canonical chain under construction". Initially, $c = c^{\perp}$. Every transformation of c removes from it those local models that are found not to be in c_S. So at any phase of the construction of c_S, c contains those local models that are *possibly* in c_S, and as such represents the information that is *necessarily* conveyed by c_S.
- a is the "anti-chain". Initially, $c = c^{\top}$. Every transformation of a adds to it those local models that are found to be in c_S. So at any phase of the construction of c_S, a contains those local models that are *necessarily* in c_S, and as such represents the information that is *possibly* conveyed by c_S.

Observation 3 *By construction, we have $c \preceq c_S \preceq a$. Therefore, by lemma 3, for any formula F:*

$$\begin{aligned} c \models F \quad &\Rightarrow \quad c_S \models F \\ a \not\models F \quad &\Rightarrow \quad c_S \not\models F \end{aligned}$$

□

We call a chain-anti-chain pair $\langle c, a\rangle$ *less evolved* than another such pair $\langle c', a'\rangle$ (denoted as $\langle c, a\rangle \leq \langle c', a'\rangle$) if and only if c is less informative than c' and a is more informative than a'. If, moreover, c is strictly less informative than c' or a is strictly more informative than a', then we say that $\langle c, a\rangle$ is strictly less evolved than $\langle c', a'\rangle$.

Lemma 12 $(\boldsymbol{C} \times \boldsymbol{C}, \leq)$ *forms a complete lattice.*

Proof. Let $\mathcal{CA}$ be a set of chain-anti-chain pairs. Let c^u and a^u (c^i and a^i) denote the component-wise union (intersection) of all chains c and a, respectively, such that $\langle c, a\rangle$ is in $\mathcal{CA}$. Then $\langle c^u, a^i\rangle$ is the greatest lower bound of $\mathcal{CA}$ and $\langle c^i, a^u\rangle$ is the least upper bound of $\mathcal{CA}$. The proof of this statement is completely analogous to that of lemma 2. □

We say that $\langle c, a\rangle$ is *minimal* among a set $\mathcal{CA}$ of chain-anti-chain pairs, if and only if $\langle c, a\rangle \in \mathcal{CA}$ and no other chain-anti-chain pair $\langle c', a'\rangle$ in $\mathcal{CA}$ is strictly less evolved than $\langle c, a\rangle$. Notice that, if $\langle c, a\rangle$ is minimal among $\mathcal{CA}$, then c is minimal among $\{c \mid \langle c, a\rangle \in \mathcal{CA}\}$.

Given a certain chain-anti-chain pair $\langle c, a\rangle$, the intended transformation $\boldsymbol{\Psi}_S$ first determines which rules in S will (not) be applicable w.r.t. c_S, and then refines $\langle c, a\rangle$ accordingly. The canonical chain c_S of S will be characterized as the first component of the $\leq$-least fixpoint of $\boldsymbol{\Psi}_S$.

We first specify how $\boldsymbol{\Psi}_S$ determines which rules will (not) be applicable w.r.t. c_S. Let $\langle c, a\rangle$ and a rule r in S be given. If r has a positive premise G, which is satisfied by c, then G will also be satisfied by c_S. On the other hand, if r has a negative premiss H, which is *not* satisfied by a, then H will not be satisfied by c_S either. So if all positive premises of r are satisfied by c and all negative premises of r are not satisfied by a, then r will be applicable with respect to c_S:

$$S^+(c, a) \quad = \quad \left\{ r \in S \;\middle|\; \begin{array}{c} \forall G \in prem^+(r) : c \models G \\ \text{and} \\ \forall H \in prem^-(r) : a \not\models H \end{array} \right\}$$

If r has a positive premise G, which is not satisfied by a, then G will not be satisfied by c_S either. If r has a negative premise H, which is satisfied by c, then H will be satisfied by c_S as well. In both cases r will certainly not be applicable with respect to c_S:

$$S^-(c, a) \quad = \quad \left\{ r \in S \;\middle|\; \begin{array}{c} \exists G \in prem^+(r) : a \not\models G \\ \text{or} \\ \exists H \in prem^-(r) : c \models H \end{array} \right\}$$

For convenience, we write:

$$S^{\sim}(c,a) = S \setminus S^{-}(c,a)$$

Think of $S^{\sim}(c,a)$ as the set of rules that is *possibly* applicable with respect to c_S, and notice that $S^{+}(c,a) \subseteq S^{\sim}(c,a)$, whenever $c \preceq a$, and that $S^{+}(c,a) = S^{\sim}(c,a)$, if $c = a$.

Lemma 13 *If S is a normal system and $\langle c,a\rangle$ and $\langle c',a'\rangle$ are two chain-anti-chain pairs s.t. $\langle c,a\rangle \leq \langle c',a'\rangle$, then we have:*

1. $S^{+}(c,a) \subseteq S^{+}(c',a')$
2. $S^{-}(c,a) \subseteq S^{-}(c',a')$
3. $S^{\sim}(c,a) \supseteq S^{\sim}(c',a')$

Proof. Suppose that $\langle c,a\rangle \leq \langle c',a'\rangle$. Then, by definition, $c \preceq c'$ and $a' \preceq a$. Let r be a rule in S. For the first statement, suppose that $r \in S^{+}(c,a)$. Then c satisfies all of r's positive premises, and a does not satisfy any of r's negative premises. By lemma 3, the same goes for c' and a', respectively, which implies that $r \in S^{+}(c',a')$. The second statement is proven analogously; the third follows directly from the second. □

Next, we specify how $\mathbf{\Psi}_S$ refines $\langle c,a\rangle$, based on $S^{+}(c,a)$ and $S^{\sim}(c,a)$. Every local model $m \in c_i$ that does not satisfy the consequence of a rule in $S^{+}(c,a)$ should certainly not be in c_S and is therefore removed from c. On the other hand, every local model $m \in c_i$ that satisfies the consequences of every rule in $S^{\sim}(c,a)$ should certainly be in c_S (S provides no ground for removing it) and is therefore added to a.

$$\mathbf{\Psi}_S(\langle c,a\rangle) \;=\; \langle \mathbf{\Psi}^{c}_S(\langle c,a\rangle), \mathbf{\Psi}^{a}_S(\langle c,a\rangle)\rangle$$

where:

$$\begin{aligned}
\mathbf{\Psi}^{c}_S(\langle c,a\rangle) &= c \setminus \{m \mid \exists r \in S^{+}(c,a) : m \not\models \mathit{cons}(r)\} \\
\mathbf{\Psi}^{a}_S(\langle c,a\rangle) &= a \cup \{m \mid \forall r \in S^{\sim}(c,a) : m \models \mathit{cons}(r)\}
\end{aligned}$$

Notice that $\mathbf{\Psi}^{c}_S$ only *removes* local models from c, whereas $\mathbf{\Psi}^{a}_S$ only *adds* local models to a.

We now prove that, starting with $\langle c^{\bot}, c^{\top}\rangle$, $\mathbf{\Psi}_S$ reaches its $\leq$-least fixpoint after finitely many iterations. To apply Tarski's fixpoint theorem, we first need to show that $\mathbf{\Psi}_S$ is monotone and continuous with respect to $\leq$.

Lemma 14 *$\mathbf{\Psi}_S$ is monotone with respect to $\leq$, that is, for every normal system S, $\langle c,a\rangle \leq \langle c',a'\rangle$ implies $\mathbf{\Psi}_S(\langle c,a\rangle) \leq \mathbf{\Psi}_S(\langle c',a'\rangle)$.*

Proof. Let S be a normal system and let $\langle c, a\rangle$ and $\langle c', a'\rangle$ be any two chain-anti-chain pairs such that $\langle c, a\rangle \leq \langle c', a'\rangle$. Then, by definition, $c \preceq c'$ and $a' \preceq a$. We need to prove that $\mathbf{\Psi}_S(\langle c, a\rangle) \leq \mathbf{\Psi}_S(\langle c', a'\rangle)$. Suppose, towards a contradiction, that this is not the case. Then $\mathbf{\Psi}^c_S(\langle c, a\rangle) \not\preceq \mathbf{\Psi}^c_S(\langle c', a'\rangle)$ or $\mathbf{\Psi}^a_S(\langle c', a'\rangle) \not\preceq \mathbf{\Psi}^a_S(\langle c, a\rangle)$. We consider both possibilities.

$\mathbf{\Psi}^c_S(\langle c, a\rangle) \not\preceq \mathbf{\Psi}^c_S(\langle c', a'\rangle)$
: In this case there must be a local model m which is contained in $\mathbf{\Psi}^c_S(\langle c', a'\rangle)$ but not in $\mathbf{\Psi}^c_S(\langle c, a\rangle)$. In the process of applying $\mathbf{\Psi}^c_S$ to $\langle c', a'\rangle$ local models may be removed from c', but no local models are added to it. So m must already be present in c'. As $c \preceq c'$, m must also be in c, thus it must have been removed from c in the process of applying $\mathbf{\Psi}^c_S$ to $\langle c, a\rangle$. It follows that there must be a rule r in $S^+(c, a)$, such that $m \not\models \mathit{cons}(r)$. From the fact that $\langle c, a\rangle \leq \langle c', a'\rangle$, by lemma 13, it follows that $S^+(c, a) \subseteq S^+(c', a')$. So r is also in $S^+(c', a')$ and m should be removed from c' in the process of applying $\mathbf{\Psi}^c_S$ to $\langle c', a'\rangle$ as well. This contradicts our earlier conclusion that $m \in \mathbf{\Psi}^c_S(\langle c', a'\rangle)$.

$\mathbf{\Psi}^a_S(\langle c', a'\rangle) \not\preceq \mathbf{\Psi}^a_S(\langle c, a\rangle)$
: In this case there must be a local model m which is contained in $\mathbf{\Psi}^a_S(\langle c, a\rangle)$ but not in $\mathbf{\Psi}^a_S(\langle c', a'\rangle)$. For $m \notin \mathbf{\Psi}^a_S(\langle c', a'\rangle)$ to hold, there must be a rule r in $S^\sim(c', a')$ such that $m \not\models \mathit{cons}(r)$. As $\langle c, a\rangle \leq \langle c', a'\rangle$, by lemma 13, we have $S^\sim(c', a') \subseteq S^\sim(c, a)$. So r must be in $S^\sim(c, a)$ as well, and therefore m cannot be in $\mathbf{\Psi}^a_S(\langle c, a\rangle)$. This contradicts our earlier conclusion that $m \in \mathbf{\Psi}^a_S(\langle c, a\rangle)$.

It follows that $\mathbf{\Psi}_S(\langle c, a\rangle) \leq \mathbf{\Psi}_S(\langle c', a'\rangle)$, as desired. □

Lemma 15 *$\mathbf{\Psi}_S$ is continuous with respect to $\leq$.*

Proof. Let $\langle c_0, a_0\rangle \leq \langle c_1, a_1\rangle \leq \langle c_2, a_2\rangle \leq \ldots$ be an infinite sequence of chain-anti-chain pairs, each of which is more evolved than all preceding ones. We need to prove that $\mathbf{\Psi}_S(\bigcup_{n=0}^{\infty}\langle c_n, a_n\rangle) = \bigcup_{n=0}^{\infty} \mathbf{\Psi}_S(\langle c_n, a_n\rangle)$. As $\mathbf{C} \times \mathbf{C}$ is finite, $\{\langle c_0, a_0\rangle, \langle c_1, a_1\rangle, \langle c_2, a_2\rangle, \ldots\}$ must have a maximum $\langle c_m, a_m\rangle$ in $\mathbf{C} \times \mathbf{C}$. So $\mathbf{\Psi}_S(\bigcup_{n=0}^{\infty}\langle c_n, a_n\rangle) = \mathbf{\Psi}_S(\langle c_m, a_m\rangle) = \bigcup_{n=0}^{\infty} \mathbf{\Psi}_S(\langle c_n, a_n\rangle)$. □

Theorem 8 *$\mathbf{\Psi}_S$ has a $\leq$-least fixpoint, which is obtained after finitely many iterations of $\mathbf{\Psi}_S$, starting with $\langle c^\bot, c^\top\rangle$.*

Proof. Follows from lemmas 12, 14, 15, by Tarski's fixpoint theorem [14]. □

Definition 3 *Let S be a normal system, and let $\langle c_S, a_S\rangle$ be the $\leq$-least fixpoint of $\mathbf{\Psi}_S$. We define c_S to be the canonical chain of S, and we define the semantics of S to be completely determined by c_S. That is, for every formula F:*

$$S \models F \quad \equiv \quad c_S \models F$$

□

A bound on the number of iterations needed by $\mathbf{\Psi}_S$ to reach its $\leq$-least fixpoint can be formulated in terms of the number of bridge rules in S.

Theorem 9 *Let S be a normal system and let $|S|$ denote the number of* bridge rules *in S. Then, starting with $\langle c^\perp, c^\top\rangle$, $\mathbf{\Psi}_S$ will reach its $\leq$-least fixpoint after at most $|S|+1$ iterations.*

Proof. A chain-anti-chain pair $\langle c, a\rangle$ is a fixpoint of $\mathbf{\Psi}_S$ if and only if $S^+(c,a) = S^+(\mathbf{\Psi}_S(\langle c,a\rangle))$ and $S^-(c,a) = S^-(\mathbf{\Psi}_S(\langle c,a\rangle)))$. During the first application of $\mathbf{\Psi}_S$ (to $\langle c^\perp, c^\top\rangle$), all facts in S are added to S^+ and all bridge rules in S are added to $S^\sim$. Each further application of $\mathbf{\Psi}_S$ either leads to a fixpoint or to the addition of at least one bridge rule to S^+ or S^-. Once a bridge rule is added to S^+ or S^-, it will not be removed again in any further iteration of $\mathbf{\Psi}_S$. It follows that $\mathbf{\Psi}_S$ must reach its $\leq$-least fixpoint after at most $|S|+1$ iterations. □

The next theorem shows that definition 3 is a proper generalization of the local model semantics for positive systems.

Theorem 10 *Let S be a positive system. Then its canonical chain coincides with its minimal solution chain.*

Proof. If S is a positive system, then for every pair $\langle c, a\rangle$, $S^+(\langle c,a\rangle)$ coincides with $S^*(c)$ and therefore $\mathbf{\Psi}^c_S(\langle c,a\rangle)$ is independent of a. As a consequence, $\langle c_S, a_S\rangle$ is the $\leq$-least fixpoint of $\mathbf{\Psi}_S$, for some anti-chain a_S, if and only if c_S is the $\preceq$-least fixpoint of $\mathbf{T}_S$. □

The canonical chain of a system S, and other fixpoints of $\mathbf{\Psi}_S$, are intimately related to the stable solution chains of S.

Lemma 16 *If c is a stable solution chain of a normal system S, then $\langle c, c\rangle$ is a fixpoint of $\mathbf{\Psi}_S$.*

Proof. Recall that:

$$\begin{aligned} S'(c) &= \{r \in S \mid \forall H \in prem^-(r) : c \not\models H\} \\ &= \left\{ r \in S \;\middle|\; \begin{array}{l} \exists G \in prem^+(r) : c \not\models G \\ \forall H \in prem^-(r) : c \not\models H \end{array} \right\} \\ &\cup \underbrace{\left\{ r \in S \;\middle|\; \begin{array}{l} \forall G \in prem^+(r) : c \models G \\ \forall H \in prem^-(r) : c \not\models H \end{array} \right\}}_{S^+(c,c)} \end{aligned} \tag{5}$$

and that c is a stable solution chain of S, only if it is the minimal solution chain of $S''(c) = pos(S'(c))$. Furthermore, observe that for every c, $S^+(c,c) = S^\sim(c,c)$, which implies that $\langle c, c\rangle$ is a fixpoint of $\mathbf{\Psi}_S$ if and only if:

$$c = \left\{ m \mid \forall r \in S^+(c,c) : m \models cons(r) \right\} \tag{6}$$

Suppose that $\langle c, c\rangle$ is *not* a fixpoint of $\mathbf{\Psi}_S$. There are two possibilities to consider:

$\exists m \in c : \exists r \in S^+(c,c) : m \not\models cons(r)$
: In this case, c is not a solution chain of $S^+(c,c)$, and therefore not a solution chain of $S''(c)$.

$\exists m \notin c : \forall r \in S^+(c,c) : m \models cons(r)$
: Suppose c is a solution chain of $S''(c)$. Then every rule in $S''(c)$ is such that c either satisfies its consequence, or does not satisfy at least one of its premises. Let c' be the chain obtained from c by adding m to it, and let r be a rule in $S''(c)$. If c satisfies r's consequence, then, by definition of m, c' does so as well. If c does not satisfy a premiss G of r, then, by lemma 3 and the fact that $c' \preceq c$, c' does not satisfy G either. It follows that c' is a solution chain of $S''(c)$ as well. So c is not the *minimal* solution chain of $S''(c)$.

In both cases, as desired, c is not a stable solution chain of S.

□

Theorem 11 *Let S be a normal system, let $\langle c_S, a_S\rangle$ be the $\leq$-least fixpoint of $\mathbf{\Psi}_S$, and let c_{stable} be a stable solution chain of S. Then $c_S \preceq c_{stable} \preceq a_S$.*

Proof. Suppose that $c_S \not\preceq c_{stable}$ or that $c_{stable} \not\preceq a_S$. Then $\langle c_S, a_S\rangle \not\leq \langle c_{stable}, c_{stable}\rangle$, while $\langle c_{stable}, c_{stable}\rangle$, by lemma 16, is a fixpoint of $\mathbf{\Psi}_S$. This contradicts the assumption that $\langle c_S, a_S\rangle$ is the $\leq$-least fixpoint of $\mathbf{\Psi}_S$. $\square$

Lemma 17 *Let S be a normal system. If $\langle c, c\rangle$ is the $\leq$-least fixpoint of $\mathbf{\Psi}_S$, then c is a stable solution chain of S.*

Proof. Recall, as in the proof of lemma 16, that c is a stable solution chain of S only if it is a minimal solution chain of $S''(c)$, that $S''(c)$ can be expressed in terms of $S^+(c, c)$ as in equation (5), and that $\langle c, c\rangle$ is a fixpoint of $\mathbf{\Psi}_S$ if and only if c satisfies condition (6).

Suppose that $\langle c, c\rangle$ is the $\leq$-least fixpoint of $\mathbf{\Psi}_S$. We first show that c is a solution chain of $S''(c)$. Observe that every rule in $S''(c) \setminus pos(S^+(c, c))$ is such that at least one of its premises is not satisfied by c, and that every rule in $pos(S^+(c, c))$ is such that c satisfies its consequence. It follows that c is a solution chain of $S''(c)$.

Next, we show that, if c is not a *minimal* solution chain of $S''(c)$, then $\langle c, c\rangle$ cannot be the $\leq$-least fixpoint of $\mathbf{\Psi}_S$.

Claim 1 *If c' is a solution chain of $S''(c)$ such that $c' \prec c$, then there is an a' such that $\langle c', a'\rangle$ is a fixpoint of $\mathbf{\Psi}_S$.*

By construction, $\langle c, c\rangle \not\leq \langle c', a'\rangle$, so from claim 1 it would follow directly that $\langle c, c\rangle$ is not the $\leq$-least fixpoint of $\mathbf{\Psi}_S$.

To prove claim 1 we show that, if a is such that $c \preceq a$, then $\mathbf{\Psi}^c_S(\langle c', a\rangle) = c'$. To see this, first notice that, as $\langle c', a\rangle \leq \langle c, c\rangle$, by lemma 13, we have $S^+(c', a) \subseteq S^+(c, c)$. Now, let r be a rule in $S^+(c, c)$. As c' complies with all rules in $pos(S^+(c, c))$, c' must satisfy $cons(r)$, whenever it satisfies all positive premises of r. Every rule r' in $S^+(c', a)$ is also in $S^+(c, c)$ and is such that c' satisfies all its positive premises. Therefore, c' must satisfy $cons(r')$ as well. This means that no local models are removed from c' in the process of applying $\mathbf{\Psi}^c_S$ to $\langle c', a\rangle$. In other words: $\mathbf{\Psi}^c_S(\langle c', a\rangle) = c'$.

For any a we have $\mathbf{\Psi}^a_S(\langle c', a\rangle) \preceq a$ (no local models will be removed from a in the process of applying $\mathbf{\Psi}^a_S$ to $\langle c', a\rangle$). So in particular, the sequence:

$$c, \mathbf{\Psi}^a_S(\langle c', c\rangle), \mathbf{\Psi}^a_S(\mathbf{\Psi}^a_S(\langle c', c\rangle)), \ldots$$

is a descending sequence with respect to $\preceq$, bounded by $c^\perp$. This means that after applying $\mathbf{\Psi}_S$ finitely many times to $\langle c', c\rangle$ a fixpoint $\langle c', a'\rangle$ of $\mathbf{\Psi}_S$ will be reached, which proves claim 1.

We conclude that c is a minimal solution chain of $S''(c)$, and therefore a stable solution chain of S. □

Theorem 12 *Let S be a normal system and let $\langle c_S, a_S \rangle$ be the $\leq$-least fixpoint of $\mathbf{\Psi}_S$. If c_S and a_S coincide, then c_S is the unique stable solution chain of S.*

Proof. Stability of c_S is established by lemma 17; uniqueness follows from theorem 11. □

Finally, we remark that, in our view, all the examples presented above are suitably dealt with by the present analysis.

6 Conclusions

We investigated the multi-context system formalism as a framework for representing contextual information and inter-contextual information flow.

We observed that the semantics of a multi-context system is completely determined by the information that is obtained when simulating the information flow specified by the system, in such a way that a *minimal* amount of information is deduced at each step of the simulation. Based on this observation, we defined an operator which determines the information entailed by the system by implementing a suitable simulation of the prescribed information flow. This operator provides a first constructive account of the local model semantics.

Next we observed that the multi-context system framework implicitly rests on the assumption that information flow is *deterministic*. We sketched a number of situations, in which this is not a valid assumption. We extended the framework in order to account for non-deterministic information flow, and provided a way to express the semantics of a non-deterministic system in terms of the semantics of a number of associated, deterministic systems. This allowed us to give a constructive account of the semantics of non-deterministic systems as well.

Finally, we observed that in the multi-context framework, new information is deduced based on the *presence* of other information. We presented a generalized framework that accounts for situations in which new information can be derived based on the *absence* of other information as well. Non-monotonic reasoning techniques were applied to establish a suitable semantics for this framework.

References

[1] K. R. Apt, H. A. Blair, and A. Walker. Towards a theory of declarative knowledge. 1988.

[2] M. Benerecetti, P. Bouquet, and C. Ghidini. Contextual reasoning distilled. *Journal of Experimental and Theoretical Artificial Intelligence*, 12(3):279–305, 2000.

[3] N. Bidoit and C. Froidevaux. General logical databases and programs: Default logic semantics and stratification. *Information and Computation*, 91:15–54, 1991.

[4] F. Fages. A new fixpoint semantics for general logic programs compared with the wellfounded and the stable model semantics. *New Generation Computing*, 9(4), 1991.

[5] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In *International Conference on Logic Programming (ICLP 88)*, pages 1070–1080, 1988.

[6] C. Ghidini and F. Giunchiglia. Local models semantics, or contextual reasoning = locality + compatibility. *Artificial Intelligence*, 127(2):221–259, 2001.

[7] F. Giunchiglia. Contextual reasoning. *Epistemologia*, XVI:345–364, 1993.

[8] F. Giunchiglia and L. Serafini. Multilanguage hierarchical logics, or: how we can do without modal logics. *Artificial Intelligence*, 65(1):29–70, 1994.

[9] J. McCarthy. Notes on formalizing context. In *International Joint Conference on Artificial Intelligence (IJCAI 93)*, pages 555–560, 1993.

[10] J. McCarthy and S. Buvač. Formalizing context (expanded notes). In *Computing Natural Language*, volume 81 of *CSLI Lecture Notes*, pages 13–50. 1998.

[11] F. Roelofsen, L. Serafini, and A. Cimatti. Many hands make light work: Localized satisfiability for multi-context systems. In *European Conference on Artificial Intelligence (ECAI 04)*, pages 58–62, 2004.

[12] C. Sakama and K. Inoue. An alternative approach to the semantics of disjunctive programs and deductive databases. *Journal of Automated Reasoning*, 13:145–172, 1994.

[13] L. Serafini and P. Bouquet. Comparing formal theories of context in AI. *Artificial Intelligence*, 155:41–67, 2004.

[14] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.

[15] A. van Gelder, K. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, 1991.