Disjunction and Modular Goal-directed Proof Search

MATTHEW STONE
Rutgers, the State University of New Jersey

This paper explores goal-directed proof search in first-order multi-modal logic. I focus on a family of modal logics which offer the expressive power to specify modular goals and local assumptions. A modular goal must be proved from designated assumptions; conversely, a local assumption can only be used to prove a designated goal. Indefinite modal specifications can avoid combinatorial interactions among independent ambiguities by making separate goals modular and corresponding disjunctive alternatives local. Such specifications can effectively guarantee that provable goals have short proofs. The key result of this paper is to establish a sound and complete goal-directed proof system that actively uses the modularity and locality of modal logic to constrain proof search. In particular, logically independent, local ambiguities will not interact in proof search. The challenge is that in goal-directed proof, a modal prover cannot simply reason locally, in a module, because modularity is a property of formulas rather than proof problems. The result therefore requires prior proof-theoretic justifications of logic programming to be extended, strengthened, and combined with new proof-theoretic analyses of modal deduction.

Categories and Subject Descriptors: F.4.1 [**Mathematical Logic**]: Modal logic; Proof theory; I.2.3 [**Deduction and Theorem Proving**]: Deduction; Logic Programming; I.2.4 [**Knowledge Representation Formalisms and Methods**]: Modal logic

General Terms: Modal logic, logic programming

Additional Key Words and Phrases: Modal logic, indefinite information, goal-directed proof, logic programming, modularity, locality

1. INTRODUCTION

Logics give rise to characteristic patterns of *information-flow*—that is, possible connections between assumptions and conclusions in proofs. For example, intuitionistic information-flow manifests itself in the correspondence between intuitionistic proofs and typed λ-terms; see, e.g., Howard [1980]. In particular, the use of assumptions in intuitionistic proofs mirrors the *modularity* and *locality* of variables bound within the scope of a λ-operator. Each assumption is restricted to a particular subtree of a proof, with a single conclusion; the assumption can only contribute to the proof of this one conclusion.

Modularity and locality characterize information-flow in a range of modal logics as well. These logics offer the expressive power to specify *modular conclusions*, which must be proved from designated assumptions, and *local assumptions*, which can only be used to

Author's address: M. Stone, Department of Computer Science and Center for Cognitive Science, Rutgers University, 110 Frelinghuysen Road, Piscataway NJ 08854-8019; email: `mdstone@cs.rutgers.edu`.

prove a designated conclusion. Modularity and locality offer general ways of structuring logical specifications [Miller 1989; Baldoni et al. 1993; 1998], as well as expressive resources for practical knowledge representation, as in e.g. [McCarthy 1993; 1997]. For example, indefinite specifications can avoid combinatorial interactions among independent ambiguities by making separate goals modular and corresponding disjunctive alternatives local. Such specifications can effectively guarantee that provable goals have short proofs.

This paper explores how information-flow in these logics not only characterizes proofs but constrains proof search. Proof search necessarily deals with partial proofs containing incomplete inferences; to focus the search, inferences are typically added in an order that can be constructed quickly, not the order that best expresses the logical structure of the argument. Effective proof search thus exploits the well-known flexibility of deduction in nonclassical logics [Fitting 1972; 1983; Wallen 1990]. Such flexibility may obscure modularity and locality, but it must ultimately respect it. For example, intuitionistic sequent calculi can be formulated so as exhibit modularity and locality while nevertheless allowing logical inferences to be applied in any order whatsoever [Stone 1999]. I show in this paper that if a proof system is designed appropriately, it can even enforce modularity and locality *incrementally* during goal-directed proof search.

1.1 Problem Statement

I begin by delineating the focus of the paper more precisely. I will work with a family of first-order multi-modal logics in this paper. Qualitatively, what distinguishes the logics I consider (for which formal definitions are provided in Section 2) is that they permit rules of modal inference to be formulated in two equivalent ways [Fitting 1972; 1983; Wallen 1990]. The first formulation structures inferences so that each subproof belongs to one local, modular context; this formulation highlights the modularity and locality of these logics. The second formulation associates each formula with its own local, modular context and allows successive inferences to apply in different contexts; this formulation streamlines proof search by allowing inference to proceed in any order. I illustrate the alternatives for the simplest case, S4 modal logic. Following Giordano and Martelli [1994], we may perhaps regard S4 as the pure modal logic of local and global modular assumptions.

1.1.1 *Structural scope and modularity.* The first formulation of modal inference is illustrated by the sequent inference figure below:

$$\frac{\Gamma^* \rightarrow G, \Delta^*}{\Gamma \rightarrow \Box G, \Delta} \, (\rightarrow \Box)$$

Such inferences set up a discipline of structural scope in proofs. Read upward, as a description of proof search, the figure describes how to accomplish generic reasoning about a modal context, such as the conclusion $\Box G$ here. We have to transform the sequent we are considering, by restricting our attention just to the generic modal statements in the sequent. Specifically, Γ^* contains the formula occurrences of the form $\Box A$ in Γ, and Δ^* contains the formula occurrences of the form $\Diamond A$ in Δ. The effect of the transformation is that we move from our current scope into a new, nested scope in which just generic information is available. Figure 1 illustrates all the structurally-scoped S4 sequent inferences that I will draw on in this motivating discussion; I refer the reader to [Fitting 1983; Wallen 1990] for more details on structurally-scoped proof.

The ability to define structural scope is intimately connected with the ability to describe

$$\frac{\Gamma^* \to G, \Delta^*}{\Gamma \to \Box G, \Delta} (\to \Box)$$
$$\frac{\Gamma, G \to \Delta}{\Gamma, \Box G \to \Delta} (\Box \to)$$
$$\frac{\Gamma, P \to G, \Delta}{\Gamma \to P \supset G, \Delta} (\to\supset)$$
$$\frac{\Gamma \to G, \Delta \qquad \Gamma, P \to \Delta}{\Gamma, G \supset P \to \Delta} (\supset\to)$$
$$\frac{\Gamma, A \to \Delta \qquad \Gamma, B \to \Delta}{\Gamma, A \vee B \to \Delta} (\vee \to)$$
$$\frac{\Gamma \to A, \Delta \qquad \Gamma \to B, \Delta}{\Gamma \to A \wedge B, \Delta} (\to \wedge)$$
$$\Gamma, A \to A, \Delta \quad (\mathit{Axiom})$$

Fig. 1. Six inference figures and the axiom for structurally-scoped S4. After [Fitting 1983; Wallen 1990]. Sequents are multisets of modal formulas; this formulation (though not others that we will consider later) requires a structural rule of contraction. See Section 2.

modular and local reasoning. In specifying reasoning, we can think of antecedent formulas in sequents as program statements and succedent formulas in sequents as goals. In modal logics with structural scope, a necessary goal $\Box G$ can be seen as a *modular* goal because, as enforced by the structurally-scoped inference figure, only program statements of the form $\Box P$ can contribute to its proof. In other words, we cannot use the entire program to prove G; rather, we must use a designated *part* of the program: formulas of the form $\Box P$. This is the *module* we use to prove G. Multi-modal logic allows us to name modules in a general way [Baldoni et al. 1993; 1998]. To create a new module, we introduce a new modality $\textsc{m}$ with necessity operator $[\textsc{m}]$; the clauses for this module now take the form $[\textsc{m}]P$ and the goals for this module take the form $[\textsc{m}]G$.

In fragments of logic without the operator $\Diamond$, including S4 translations of intuitionistic formulas in particular, modularity brings *locality*. A goal $\Box(P \supset G)$ introduces a *local* assumption P. The assumption is local in the sense that it can only contribute to the proof of G, and cannot contribute to any other goal. We can motivate this locality in logical terms by examining the sequent inferences for $(\to \Box)$ and $(\to\supset)$ in combination:

$$\frac{\dfrac{\Gamma^*, P \longrightarrow G}{\Gamma^* \longrightarrow P \supset G} (\to\supset)}{\Gamma \longrightarrow \Box(P \supset G), \Delta} (\to \Box)$$

Observe that this logical fragment is constructed so that the succedent context Δ^* above $(\to \Box)$ is empty, and so we introduce P into a subproof where G is the only goal.

Logical modularity and locality underlie the use of the proof theory of modal logic as a declarative framework for structuring specifications [Miller 1989; Giordano and Martelli 1994; Baldoni et al. 1993; 1996; 1998].[1] Logical modularity and locality together allow a designer to view a fragment of a logical theory as a genuine component of a larger specification. For example, if all assumptions are local to the fragment, they will not interact with the other goals in a larger proof problem. Likewise, if all the goals in a fragment are modular, they must be proved directly from the fragment and so are insulated from changes in the other parts of the specification.

[1]The model theory of modal logic can also be used to structure specifications [Sakakibara 1987].

$$\dfrac{\dfrac{\dfrac{\dfrac{A,\ldots \longrightarrow A \qquad \dfrac{\dfrac{B,\ldots \longrightarrow B,A \qquad A,\ldots \longrightarrow A}{B,B\supset A,\ldots \longrightarrow A}(\supset\rightarrow)}{B,\Box(B\supset A),\ldots \longrightarrow A}(\Box\rightarrow)}{A\vee B,\Box(B\supset A),\ldots \longrightarrow A}(\vee\rightarrow)}{\Box(A\vee B),\Box(B\supset A),\ldots \longrightarrow A}(\Box\rightarrow)}{\Box(A\vee B),\Box(B\supset A),\ldots \longrightarrow \Box A}(\rightarrow\Box) \quad \dfrac{\dfrac{\dfrac{C,\ldots \longrightarrow C \qquad \dfrac{\dfrac{D,\ldots \longrightarrow D,C \qquad C,\ldots \longrightarrow C}{D,D\supset C,\ldots \longrightarrow C}(\supset\rightarrow)}{D,\Box(D\supset C),\ldots \longrightarrow C}(\Box\rightarrow)}{C\vee D,\Box(D\supset C),\ldots \longrightarrow C}(\vee\rightarrow)}{\Box(C\vee D),\Box(D\supset C),\ldots \longrightarrow C}(\Box\rightarrow)}{\Box(C\vee D),\Box(D\supset C),\ldots \longrightarrow \Box C}(\rightarrow\Box)}{\Box(A\vee B),\Box(C\vee D),\Box(B\supset A),\Box(D\supset C) \longrightarrow \Box A\wedge\Box C}(\rightarrow\wedge)$$

Fig. 2. This structurally-scoped S4 proof shows how the locality of modular assumptions limits the possible interactions in proof. Ellipses mark points in sequents where I have suppressed additional formula occurrences that no longer contribute to the inference.

In this paper, I further emphasize that logical modularity and locality provide declarative tools for constraining *the complexity of proof search itself*. My motivating example is the proof in Figure 2, which establishes that the conclusion

$$\Box A \wedge \Box C$$

follows from the assumptions

$$\Box(A\vee B),\Box(C\vee D),\Box(B\supset A),\Box(D\supset C)$$

The assumptions in this proof—the program statements—specify two ambiguities. Either A or B holds, and either C or D holds. As part of the specification, we use modal operators to say how to reason with these ambiguities: we have $\Box(A\vee B)$ and $\Box(C\vee D)$. This means that the ambiguities themselves are *generic*; we can use them to perform case analysis at any time. However, when we reason about any particular case, we make *local* assumptions—we will assume A rather than $\Box A$ for example.

This specification limits the way case analyses in the proof interact. Consider our goal here: $\Box A\wedge\Box C$. We must prove each conjunct separately, *using generic information*; that is, each conjunct is proved in its own new nested scope. Thus, in the proof of Figure 2, we perform case analysis from $\Box(A\vee B)$ within the nested scope for $\Box A$, and perform case analysis from $\Box(C\vee D)$ within the nested scope for $\Box C$. Observe that the logic dictates the choice for us. For instance, performing case analysis from $\Box(A\vee B)$ within the nested scope for $\Box C$ is useless—the assumption of A and B cannot help here. Importantly, performing case analysis from $\Box(A\vee B)$ at the initial, outermost scope is also useless. Whatever assumptions we make will have to be discarded when we try to prove each of the conjuncts $\Box A$ and $\Box C$, and consider only generic information. This specification therefore cordons off the two ambiguities from one another in this proof problem. We have to consider the ambiguities separately.

Effectively, it is part of the *meaning* of the specification of Figure 2 that *proofs must be short*. A proof in this specification must be a combined record of independent steps, not an interacting record with combined resolutions of ambiguities. To my knowledge, the possibility for this kind of declarative search control in disjunctive modal specifications has not received comment previously. But it seems to me to be one of the most exciting and unique uses for modal logic in representation and problem-solving.

$$\frac{\Gamma \to G^{\mu\alpha}, \Delta}{\Gamma \to \Box G^{\mu}, \Delta} (\to \Box)$$

$$\frac{\Gamma, G^{\mu\nu} \to \Delta}{\Gamma, \Box G^{\mu} \to \Delta} (\Box \to)$$

$$\frac{\Gamma, P^{\mu} \to G^{\mu}, \Delta}{\Gamma \to P \supset G^{\mu}, \Delta} (\to\supset)$$

$$\frac{\Gamma \to G^{\mu}, \Delta \qquad \Gamma, P^{\mu} \to \Delta}{\Gamma, G \supset P^{\mu} \to \Delta} (\supset\to)$$

$$\frac{\Gamma, A^{\mu} \to \Delta \qquad \Gamma, B^{\mu} \to \Delta}{\Gamma, A \vee B^{\mu} \to \Delta} (\vee \to)$$

$$\frac{\Gamma \to A^{\mu}, \Delta \qquad \Gamma \to B^{\mu}, \Delta}{\Gamma \to A \wedge B^{\mu}, \Delta} (\to \wedge)$$

$$\Gamma, A^{\mu} \to A^{\mu}, \Delta \quad (Axiom)$$

Fig. 3. Six inference figures and the axiom for explicitly-scoped S4. See [Fitting 1983; Wallen 1990; Stone 1999]. The $(\to \Box)$ inference is subject to an eigenvariable condition that α is new. In the $(\Box \to)$ inference, $\mu\nu$ refers to any sequence of terms that extends μ by a suffix ν.

1.1.2 *Explicit scope and goal-directed search.*

The second formulation of modal reasoning is illustrated by the sequent figure below:

$$\frac{\Gamma \longrightarrow G^{\mu\alpha}, \Delta}{\Gamma \longrightarrow \Box G^{\mu}, \Delta} (\to \Box)$$

Such inferences institute an explicitly-scoped sequent calculus; each formula is tagged with a label indicating the modal context which it describes. These labels are sequences of terms, each of which corresponds to an inference that changes scope. Superscripts are my notation for labels; above, μ labels the scope of the succedent formula $\Box G$. To reason about a generic modal formula, we again introduce a new, nested scope in which just generic information is available. We do this by introducing a symbol α to represent a generic possibility, subject to an eigenvariable condition—α cannot occur elsewhere in the sequent. We now label the new formula with its new scope: G is labeled $\mu\alpha$. At axioms, the scopes of premises and conclusions must match; therefore modal inference figures can propagate upward the complete inventories of premises and conclusions Γ and Δ.

Figure 3 illustrates the other explicitly-scoped S4 sequent inferences that I will draw on in this motivating discussion. Explicitly-scoped proof systems have a long history as *prefixed tableaus*; see [Fitting 1983; Wallen 1990] and references therein. Each label sequence can be viewed as representing a possible world in possible-worlds semantics, so for example the inference figure $(\to \Box)$ represents a transition from the world named by μ to a new world $\mu\alpha$ that represents a generic possibility accessible from μ. The more general study of such systems has put them in a new proof-theoretic perspective recently. They are closely related to semantics-based translation systems [Ohlbach 1991; Nonnengart 1993] and *labelled deductive systems* [Gabbay 1996; Basin et al. 1998]. I use the term *explicitly-scoped* from [Stone 1999] because I continue to emphasize the extent to which the two formulations of reasoning represent the same inferences, just in different ways.

The ability to define explicit scope is intimately connected with the ability to carry out goal-directed proof. I adopt the perspective due to [Miller et al. 1991] that goal-directed proof simply amounts to a specific strategy for constructing sequent calculus deductions. The strategy is first to apply inferences that decompose goals to atoms and then to apply inferences that use a specific program statement to match a specific goal. Proofs that respect

$$\boxed{3}\qquad \dfrac{\dfrac{\underline{B},\ldots \longrightarrow \ldots,\underline{B} \qquad \underline{D},\ldots \longrightarrow \ldots,\underline{D}}{B,D,\ldots \longrightarrow \underline{B\wedge D}}\,(\rightarrow\wedge) \qquad \underline{F},\ldots \longrightarrow \ldots,\underline{F}}{B,D,\underline{(B\wedge D)\supset F},\ldots \longrightarrow \ldots,\underline{F}}\,(\supset\rightarrow)$$

$$\boxed{2}\qquad \dfrac{\dfrac{\underline{C},\ldots \longrightarrow \ldots,\underline{C} \qquad \boxed{3}}{B,\underline{C\vee D},(B\wedge D)\supset F,\ldots \longrightarrow \ldots,\underline{C},F}\,(\vee\rightarrow) \qquad \underline{F},\ldots \longrightarrow \ldots,\underline{F}}{B,C\vee D,\underline{C\supset F},(B\wedge D)\supset F \longrightarrow \ldots,\underline{F}}\,(\supset\rightarrow)$$

$$\boxed{1}\qquad \dfrac{\dfrac{\underline{A},\ldots \longrightarrow \underline{A},F \qquad \boxed{2}}{\underline{A\vee B},C\vee D,C\supset F,(B\wedge D)\supset F \longrightarrow \underline{A},F}\,(\vee\rightarrow) \qquad \underline{F},\ldots \longrightarrow \underline{F}}{A\vee B,C\vee D,\underline{A\supset F},C\supset F,(B\wedge D)\supset F \longrightarrow \underline{F}}\,(\supset\rightarrow)$$

Fig. 4. A goal-directed proof in which multiple cases are considered. Each case is displayed in a separate block; numerical indices on blocks and in inferences, e.g., $\boxed{2}$, indicate how the blocks contribute to a single inference.

this strategy are called *uniform*. On this strategy, logical connectives amount to explicit instructions for search, and this is in fact what lets us view a logical formula concretely as a *program* [Miller et al. 1991].

Unlike other, more procedural characterizations of algorithmic proof, such as [Gabbay 1992], this view largely abstracts away from the exact state of computations during search. The key questions are purely proof-theoretic. In particular, goal-directed proof is possible in a logic if and only if any theorem has a uniform proof. In systems of structural scope, this is not possible, and we must instead restrict our to inference in specific logical fragments, as described for the intuitionistic case in, e.g., [Miller et al. 1991; Harland 1994; Harland et al. 2000].[2]

By contrast, systems of explicit scope can be lifted by a suitable analogue to the Herbrand-Skolem-Gödel theorem for classical logic so that *any* pair of unrelated inferences can be interchanged [Kleene 1951; Wallen 1990; Lincoln and Shankar 1994; Stone 1999]. Thus, unlike systems of structural scope, systems of explicit scope permit *general* goal-directed reasoning. If we adopt Miller's characterization of uniform proof for sequent calculi with multiple conclusions [Miller 1994; 1996], then any modal theorem has a uniform proof in a lifted, explicitly-scoped inference system. Put another way, explicitly-scoped inference assimilates modal proof to classical proof, and we know that uniformity is not really a restriction on classical proof [Harland 1997; Nadathur 1998]. This is why my investigation emphasizes questions of information-flow, such as modularity and locality, rather than questions of goal-directed proof *per se*.

I will refer to the proof of Figure 4 to illustrate some of the properties of information-flow in goal-directed search. The proof establishes the conclusion

$$F$$

[2]A further case of structural control of inference that has attracted particular interest is linear logic, where linear disjunction must be understood to specify synchronization between concurrent processes rather than proof by case analysis; see, e.g., [Andreoli 1992; Hodas and Miller 1994; Pym and Harland 1994; Miller 1996; Kobayashi et al. 1999]. The investigation of fragments of linear logic remains essential, as linear logic has no analogue of an explicitly-scoped proof system, and so—unlike intuitionistic logic and modal logic—must be understood as a refinement of classical logic rather than an extension to it [Girard 1993].

from assumptions

$$A \vee B, C \vee D, A \supset F, C \supset F, (B \wedge D) \supset F$$

First we must get clear on the reasoning Figure 4 represents. The assumptions in this proof again specify two ambiguities, $A \vee B$ and $C \vee D$. In modal terms, these are local ambiguities that introduce local assumptions; but actually, Figure 4 uses only classical connectives, and this classical reasoning suffices for my discussion here. In Figure 4, the two ambiguities interact to require inference for three separate cases: one case where A is true, one case where C is true, and a final case where B and D are true together. (These cases are laid out separately in Figure 4.) In goal-directed inference, we discover these cases by backward chaining from the main goal F through a series of implications: $A \supset F$, $C \supset F$, and $(B \wedge D) \supset F$.

Now we can describe the structure of the proof of Figure 4 more precisely. The inference is segmented out into three chunks, one for each case. The chunks are indexed to indicate how they should be assembled into a single proof-tree; the chunk indexed $\boxed{3}$ should appear as a subtree where the index $\boxed{3}$ is used in chunk $\boxed{2}$, and that chunk should in turn appear as a subtree where the index $\boxed{2}$ is used in chunk $\boxed{1}$. We could imagine writing out that tree in full—on an ample blackboard! However, the chunks are actually natural units of the proof of Figure 4; they are what Loveland calls *blocks* [Loveland 1991; Nadathur and Loveland 1995]. In general, a *block* of a derivation is a maximal tree of contiguous inferences in which the right subtree of any $(\vee \rightarrow)$ inference in the block is omitted. (Check this in Figure 4.) Each block presents reasoning that describes a single case from the specification.

Within blocks, we can trace the progress of goal-directed reasoning, as follows. At each step, our attention is directed to a distinguished goal formula—the current goal—and at most one distinguished program formula—the selected statement. For illustration, these distinguished formulas are underlined in each sequent in Figure 4. Logical operations apply only to distinguished formulas; we first decompose the goal down to atomic formulas, then select a program formula and reason from it to establish the current goal.

There are two things to notice about this derivation. First, we use a *restart* discipline when handling disjunctive case analysis across blocks [Loveland 1991; Nadathur and Loveland 1995]. In each new block, the current goal is reset to the original goal F to restart proof search. It is easy to see that it does not suffice, in general, to keep the current goal across blocks; in Figure 4, for example, keeping the current goal would mean continuing to try to prove A after we turn from the case of A to the case of B. The more general restart rule is however complete; in fact, the restart rule is a powerful way to extend a goal-directed proof system to logics where a single proof must sometimes analyze the same goal formula in qualitatively different ways [Gabbay and Reyle 1984; Gabbay 1985; 1992].

Second, note when and how newly-assumed disjuncts are used in new blocks. For example, B is assumed in block $\boxed{2}$ but it is not used until block $\boxed{3}$. By contrast, D is assumed in block $\boxed{3}$ and used immediately there. Following [Loveland 1991], I will refer to any use of a disjunctive premise in the first block of case analysis where it is assumed as a *cancellation*; I will also say that the inference that introduces the disjunct, and the new block it creates, are *canceled*. The proof of Figure 4 cannot be recast in terms of canceled inferences using the sequent rules of Figure 3. Whichever case of $A \vee B$ or $C \vee D$ is treated first cannot be canceled; the second disjunct of the one must wait to be used until the second disjunct of the other is introduced. This is a gap between Loveland's original

Near-Horn Prolog interpreter [Loveland 1991], which requires cancellations, and the generalized reformulation in terms of sequent calculi given in [Nadathur and Loveland 1995; Nadathur 1998] and suggested in Figure 4. Loveland suggests that cancellation is just an optimization, but we shall see that modal logic establishes an important proof-theoretic link between cancellation and modularity.

1.1.3 *On modular goal-directed proof search.* Structurally-scoped systems and explicitly-scoped systems highlight independent constraints on proof search. Structurally-scoped systems do not afford goal-directed search—they are incompatible with it. But explicitly-scoped systems do not enforce modularity and locality either—as proofs are constructed incrementally, inferences in any scope can be added. Nevertheless, as befits alternative proof methods for the same logic, structurally-scoped systems and explicitly-scoped systems are very close. In the case of intuitionistic logic, the two systems define not just the same theorems but the same proofs [Stone 1999]. Insights about information-flow in structurally-scoped proofs—including the modularity and locality exhibited by Figure 2—should therefore be applicable to restrict goal-directed proof search in explicitly-scoped systems. The present paper substantiates this conjecture.

In goal-directed proof search, any inference carries a commitment to potential links between assumptions and conclusions. Even in explicitly-scoped systems, these links are constrained by a discipline of scope. We can work backward to derive constraints on our current inferences from the requirements of these future links. For example, we know from [Stone 1999] that we can restrict reasoning from assumptions in explicitly-scoped inferences straightforwardly, as follows. Assume that we have an explicitly-scoped proof system for a logic with modularity and locality, with an eigenvariable condition on $(\rightarrow \Box)$, and we work in a fragment of logic without negation (this again includes the S4 translation of intuitionistic formulas). Then when we consider a sequent of this form in proof search:

$$\Gamma \longrightarrow \Delta$$

we apply inferences to a formula P in Γ only when P is labeled with a prefix of a label of a formula in Δ. That is, we can consider inferences on $P^{\mu} \in \Gamma$ only when there is some $G^{\mu\nu} \in \Delta$. The prefix relationship is required for P to eventually contribute to the proof of any Δ formula. For example here:

$$A, B^{\beta}, C^{\gamma}, D^{\delta} \longrightarrow E^{\beta}, F^{\gamma}$$

We can consider A, B^{β} or C^{γ}, but not D^{δ}.

However, this invariant is weaker than one might want or expect for certain kinds of goal-directed search. Specifically, we have seen that when we use goal-directed search as a model for logic programming, we understand the interpreter to be working on at most one goal and one program statement at a time. In this setting, we would like to require that the program statement must be able to contribute to the current goal. However, we must understand the result of [Stone 1999] to require only that the current program statement must contribute to *some* goal, not necessarily the *current* one. In addition, we must take into account other, inactive goals, such as the goals that we may potentially restart later in proof search. In the preceding example, even if E^{β} were the current goal, we might have to consider reasoning with C^{γ} because of the possibility of a restart with F^{γ}.

For most inferences, we can rule out their contribution to inactive goals, on independent grounds. Most inferences from assumptions in goal-directed proofs must contribute to the current goal or not at all. But there is one difficult case, which happens also the most meaningful one. This is the case of disjunction itself, where only one disjunct contributes to the current goal. The other disjunct may contribute to some other goal; we will set up a new proof problem by assuming this disjunct and making some inactive goal active. Because we are committed to use this newly-assumed disjunct in proof, modularity and locality suggest that we should be able now to select a goal that our newly-assumed disjunct could contribute to. In other words, if the new disjunct is P^{μ} the next goal should take the form $G^{\mu\nu}$ with μ a prefix of $\mu\nu$. Call this a modular restart. The alternative is that there is no relationship of scope between the new disjunct and the next goal.

Modular restarts are crucial to the use of logic to structure indefinite specifications. For example, modular restarts would allow us to capture the declarative search control illustrated in Figure 2; they seem required for it. In an explicitly-scoped goal-directed proof corresponding to Figure 2, the case analysis for $\Box(A \vee B)$ will look like this:

$$\frac{\dfrac{\underline{A}^{\alpha},\ldots \longrightarrow \underline{A}^{\alpha},\ldots \qquad B^{\alpha},\ldots \longrightarrow A^{\alpha},\ldots}{\underline{A \vee B}^{\alpha},\ldots \longrightarrow \underline{A}^{\alpha},\ldots}\vee\rightarrow}{\underline{\Box(A \vee B)},\ldots \longrightarrow \underline{A}^{\alpha},\ldots}\Box\rightarrow$$

With modular restarts, we know we must continue to take A^{α} as the current goal in the right top (B^{α}) subproof. In effect, we know to build a short proof in which ambiguities are considered independently. We can cut down the space for proof search accordingly—for example, there will be no question of introducing the other ambiguity from $\Box(C \vee D)$ in the new modular block. On the other hand, without modular restarts, we are free to reconsider the initial goal $\Box A \wedge \Box C$ at this stage; in subsequent search we will reconsider both $\Box(A \vee B)$ and $\Box(C \vee D)$. Thus, even though the logic guarantees that ambiguities do not interact in a proof, we still wind up considering interacting ambiguities in proof search.

The main result of this paper is to provide an explicitly-scoped goal-directed proof system in which modular restarts are complete. The proof system has modular restarts because, in the new proof system, any proof can be presented in such a way that all disjunctions are canceled. Each new disjunct P^{μ} therefore contributes to the proof of the restart goal in the current block, and so we know to choose a restart goal $G^{\mu\nu}$ that the new disjunct could contribute to.

It turns out that modular restarts are not automatic; you need to design the policy for disjunctive inference to respect it. Figure 4 already makes the problem clear. How can we enforce cancellations here? The sequent rules seem not to allow it. The new idea is simple actually—to allow a new inference figure for disjunction that considers disjuncts out of their textual order:

$$\frac{\Gamma, D^{\mu} \longrightarrow \Delta \qquad \Gamma, C^{\mu} \longrightarrow \Delta}{\Gamma, C \vee D^{\mu} \longrightarrow \Delta}\vee\rightarrow *$$

This is the direct analogue of the Near-Horn Prolog inference scheme, which can proceed by matching any of the heads of a disjunctive clause at any time [Loveland 1991]. The new sequent rule will allow us to reanalyze the constitution of higher blocks so that, wherever we use the new disjunct in the original proof, we can always reanalyze it as part of the current block. Figure 5 demonstrates this reanalysis for Figure 4. In fact, demonstrating the generality of such reanalysis will prove to be quite involved. Explicitly-scoped inferences with an eigenvariable condition give blocks in modal proofs an inherently hierarchical structure, because of the different modal scopes that are introduced and the local assump-

$$\boxed{2'}\qquad \frac{\underline{C},\ldots \longrightarrow \ldots,\underline{C} \qquad \underline{F},\ldots \longrightarrow \ldots,\underline{F}}{B,C,\underline{C\supset F},\ldots \longrightarrow \ldots,\underline{F}}\,(\supset\rightarrow)$$

$$\boxed{3'}\qquad \frac{\dfrac{\underline{B},\ldots \longrightarrow \ldots,\underline{B} \qquad \dfrac{\underline{D},\ldots \longrightarrow \ldots,\underline{D} \qquad \boxed{2'}}{B,\underline{C\vee D},C\supset F,\ldots \longrightarrow \ldots,\underline{D}}\,(\vee\rightarrow *)}{B,C\vee D,C\supset F,\ldots \longrightarrow \underline{B\wedge D}}\,(\rightarrow\wedge) \qquad \underline{F},\ldots \longrightarrow \ldots,\underline{F}}{B,C\vee D,C\supset F,\underline{(B\wedge D)\supset F},\ldots \longrightarrow \ldots,\underline{F}}\,(\supset\rightarrow)$$

$$\boxed{1}\qquad \frac{\dfrac{\underline{A},\ldots \longrightarrow \underline{A},F \qquad \boxed{3'}}{\underline{A\vee B},C\vee D,C\supset F,(B\wedge D)\supset F \longrightarrow \underline{A},F}\,(\vee\rightarrow) \qquad \underline{F},\ldots \longrightarrow \underline{F}}{A\vee B,C\vee D,\underline{A\supset F},C\supset F,(B\wedge D)\supset F \longrightarrow \underline{F}}\,(\supset\rightarrow)$$

Fig. 5. A reanalysis of the proof of Figure 4 to enforce cancellations. We rewrite block $\boxed{3'}$, in which B is canceled, to use the new disjunctive inference figure; block $\boxed{3'}$ thus becomes the second block after $\boxed{1}$. At the same time, we introduce a simplified block $\boxed{2'}$ which uses the assumption C, without disjunction at all.

tions that are made. Loveland's construction for cancellations, by contrast, assumes that the structure of blocks is flat. Instead, we must use the natural tools of the sequent calculus to develop suitable constructions for reanalyzed inferences.

1.2 The results and their context

The problem sketched in Section 1.1 is a pure problem of modal proof. Accordingly, all the proof systems I consider will describe sound and complete inference under the usual Kripke semantics for modal logic [Kripke 1963; Fitting 1983]. I will not consider interactions of disjunction with negation-by-failure and other operational features of of logic programming proof-search systems. For such issues in disjunctive logic programming, see for example [Lobo et al. 1992]. Nor will I attempt to describe a minimal model or fixed-point construction in which exactly the consequences of a modal program hold, as in [Orgun and Wadge 1992].

Moreover, my interest is in specific fragments of specific modal logics in particular. *Modularity* and *locality* allow consideration of the logics T, K and K4 in addition to S4, but are not compatible with such logics as S5, temporal logics with symmetric past and future operators [Gabbay 1987], the logic of context of [McCarthy and Buvač 1994] or the modal logic of named addresses of [Kobayashi et al. 1999]. For example, in S5, if $\Box A$ is true at any world, then $\Box A$ is true at all worlds; thus the logic prohibits making such an assumption locally. To see the problem, observe, for example, that $\Box(\Box A \supset B) \vee \Box A$ is a theorem of S5. Modal proofs in such cases require global restarts [Gabbay and Olivetti 1998]. *Locality* further rules out logical fragments with possibility or negation. Such fragments can be used to pose goals that access otherwise local assumptions, as in the theorem $\Box(A \supset B) \vee \Diamond A$ of all normal modal logics. (Goal-directed proof of this theorem also involves a global restart.) Moreover, such fragments make it more difficult to enforce modularity as well, since they do not permit an eigenvariable condition at $(\rightarrow \Box)$ inferences in goal-directed proofs. My investigation therefore sticks closely to the treatments of logical modularity and locality originally explored in [Miller 1989; Giordano and Martelli 1994]. Indeed, I continue to restrict implications and universal quantifiers in goals to strict statements of the form $\Box(P \supset G)$ and $\Box\forall x G$.

The basic strategy that I adopt is to start with a relatively straightforward proof system, and gradually to narrow the formulation of its inference rules—preserving soundness and completeness with respect to the underlying semantics—until we have a proof system, SCLP, with the desired characteristics, namely goal-directed search and modular restarts. I have been particularly influenced by Lincoln and Shankar's presentation of proof-theoretic results in terms of simple transformations among successive proof systems [Lincoln and Shankar 1994]; and by Andreoli's construction of focusing sequent calculi that embody the discipline of goal-directed proof directly in the form of inference figures [Andreoli 1992].

However, the correct design of the final proof system requires a variety of proof-theoretic ideas about logic programming to be extended, strengthened, and combined with new proof-theoretic results about modal logic. To describe logic programming, we start with the idea of uniform proof search described in [Miller et al. 1991] and extended to multiple-conclusion calculi in [Miller 1994]. To derive a uniform proof system in the presence of indefinite information in assumptions, however, we can no longer use the familiar quantifier rules used in previous logic programming research, which simply introduce fresh parameters; we must apply a generalization of Herbrand's Theorem [Lincoln and Shankar 1994] and work with quantifier rules that introduce structured terms. The calculus of Herbrand terms, SCL, lifts the explicitly-scoped proof systems considered in Section 1.1.2 and [Fitting 1983; Wallen 1990]. The key property of SCL is that inferences can be freely interchanged. This allows arbitrary proofs to be transformed easily into uniform proofs.

The *modular* behavior of this uniform system depends on the further proof-theoretic analyses of path-based sequent calculi adapted, in part, from [Stone 1999]. These analyses establish that path representations enforce modularity and locality in the uses of formulas in proofs, even with otherwise classical reasoning. Hence, although path-based calculi obscure the natural modularity of modal inference, they do not eliminate it. I finish with a streamlined uniform proof system that takes advantage of these results; as a consequence, proof search in this calculus can dynamically exploit the local use of modular assumptions.

The justification of this new proof system makes much of a strategy originally due to [Kleene 1951], in which the inferences in a proof are reordered so as to satisfy a global invariant. The strategy achieves termination despite generous copying and deepening of inferences by a judicious choice of transformations within a double induction. In our cases, these transformations are guided by the constraints of uniform proof, and by the cancellations of disjunctive assumptions that we know we must maintain in proofs, to achieve modularity. This provides an analogue of Loveland's transformations on restart proofs [Loveland 1991] in the sequent calculus setting.

Of course, modal logic is not just a modular logic. Modal logic provides a general, declarative formalism for specifying change over time, the knowledge of agents, and other special-purpose domains [Prior 1967; Hintikka 1971; Schild 1991]. Goal-directed systems for modal proof are often motivated by such specifications [Fariñas del Cerro 1986; Debart et al. 1992; Baldoni et al. 1993; 1996]. In generalizing goal-directed modal proof to indefinite specifications, SCLP can play an important role in applying modal formalisms to planning, information-gathering and communication [Stone 1998a; 2000]. Even when content, not modularity, is primary, the modular treatment of disjunction limits the size of proofs and the kinds of interactions that must be considered in proof search. Such constraints are crucial to the use of logical techniques in applications that require automatic assessment of incomplete information, such as planning and natural language generation. The interest of these more general applications helps explain why I pursue this investigation in the full first-order language.

1.3 Outline

The structure of the rest of this paper is as follows. I begin by presenting first-order multi-modal logic in Section 2. I consider syntax (Section 2.1), semantics (Section 2.2), and finally proof (Section 2.3); I describe the explicitly-scoped Herbrand proof system for modal logic that is my starting point. Section 2.4 shows that this calculus offers a suitable framework for goal-directed proof because uniform proof search in this calculus is complete.

Section 3 describes and justifies a modular goal-directed proof system, as advertised in Section 1.1. I introduce the calculus itself in Section 3.1, along with key definitions and examples. Then in Sections 3.2–3.4 I outline how this sequent calculus is derived in stages from the calculus of Section 2. Full details are provided in an electronic appendix.

Finally, Section 4 offers a broader assessment of these results. I consider some further optimizations that the new sequent calculus invites in Section 4.1, and briefly conclude in Section 4.2 with some applications of first-order multi-modal inference that the new sequent calculus suggests.

2. FIRST-ORDER MULTI-MODAL DEDUCTION

I begin by supporting the informal presentation of first-order multi-modal logic from Section 1 more explicitly. I will adopt a number of techniques that are individually quite familiar. I allow an arbitrary number of modal operators and a flexible regime for relating different modal operators to one another, following many applied investigations [Debart et al. 1992; Baldoni et al. 1993; 1996; Baldoni 2000]. I use prefix terms for worlds and sequent calculus inference, following the comprehensive treatment of the first-order modal logic using prefix terms and analytic tableaux (or, seen upside-down, in the cut-free sequent calculus) of [Fitting and Mendelsohn 1998]. I factor out reasoning about accessibility into side conditions on inference rules, similar to the proof-theoretic view of [Basin et al. 1998], in which reasoning about accessibility and boolean reasoning are clearly distinguished. And I use Herbrand terms to reason correctly about parameterized instances of formulas, avoiding the usual eigenvariable condition on quantifier (and modal) rules, as in [Lincoln and Shankar 1994].

Though the techniques are routine, the combination is still somewhat unusual. Research in modal logic—whether the investigation is more mathematical [Goré 1992; Massacci 1998b; 1998a; Goré 1999] or primarily concerns algorithms for proof search [Otten and Kreitz 1996; Beckert and Goré 1997; Schmidt 1998]—is dominated by the study of the propositional logic of a single modal operator (or accessibility relation). Moreover, researchers who have investigated modal logic in a first-order setting have tended to jump directly into a discussion of particular theorem-proving strategies, particularly resolution [Jackson and Reichgelt 1987; Wallen 1990; Catach 1991; Frisch and Scherl 1991; Auffray and Enjalbert 1992; Nonnengart 1993; Ohlbach 1993].

2.1 Syntax

Our language depends on a *signature* including a suitable set of atomic constants C (and suitable predicate symbols and modalities). We then consider program statements of the syntactic category $D(C)$ and goals of the category $G(C)$ defined recursively as in (1); we refer to the union of these two languages as $L(C)$. In (1), we make the conditions observed in Section 1.2 explicit: there is no possibility or negation, and universal and hypothetical goals must be modal.

$$\begin{aligned} G &::= A \mid [\mathrm{M}]G \mid G \wedge G \mid G \vee G \mid [\mathrm{M}](\forall x G) \mid \exists x G \mid [\mathrm{M}](D \supset G) \\ D &::= A \mid [\mathrm{M}]D \mid D \wedge D \mid D \vee D \mid \forall x D \mid \exists x D \mid G \supset D \end{aligned} \quad (1)$$

In (1), A schematizes an atomic formula; atomic formulas take the form $p_i(a_1, \ldots, a_k)$ where p_i is a predicate symbol of arity k and each a_i is either a variable or an atomic constant in the set C. We assume some initial non-empty set of constants $CONST$. But it will be convenient to consider languages in which a countably infinite number of *parameters* are included in the language to supplement the symbols in CONST.

In (1), $[\mathrm{M}]$ schematizes a modal operator of necessity; intuitively, such modal operators allow a specification to manipulate constrained sources of information. That is, a program statement $[\mathrm{M}]D$ explicitly indicates that D holds in the constrained source of information associated with the operator $[\mathrm{M}]$. Conversely, a goal $[\mathrm{M}]G$ can be satisfied only when G is established by using information from the constrained source associated with $[\mathrm{M}]$.

We will work in a multi-modal logic, in which any finite number m of distinct necessity operators or *modalities* may be admitted. (Necessity operators will also be written as $\Box$ or $\Box_i$.) In addition to ordinary program statements, a specification may contain any of the following axiom schemes describing the modalities to be used in program statements and goals:

$$\begin{array}{ll} \Box_i p \supset p & \textit{veridicality } (\mathrm{VER}) \\ \Box_i p \supset \Box_i \Box_i p & \textit{positive introspection } (\mathrm{PI}) \\ \Box_i p \supset \Box_j p & \textit{inclusion } (\mathrm{INC}) \end{array} \quad (2)$$

These axioms describe the nature of the information that an operator provides, and spell out relationships among the different sources of information in a specification. (VER) is needed for information that correctly reflects the world; (PI), for information that provides a complete picture of how things might be; and (INC), for a source of information, j, that elaborates on information from another source, i. Because we use this explicit axiomatization, we can take the names of the modal operators as arbitrary.

We appeal to the usual notions of *free* and *bound* occurrences of variables in formulas; we likewise invoke the *depth* of a formula—the largest number of nested logical connectives in it.

2.2 Semantics

As is standard, we describe the models for the modal language in two steps. The first step is to set up *frames* that describe the structure of any model; a full model can then be obtained by combining a frame with a way of assigning interpretations to formulas in a frame.

Definition 2.1 *Frame*. A *frame* consists of a tuple $\langle \mathbf{w}, \mathcal{R}, \mathcal{D} \rangle$ where: $\mathbf{w}$ is a non-empty set of *possible worlds*; $\mathcal{R}$ names a family of m binary *accessibility* relations on $\mathbf{w}$, a relation $\mathcal{R}_i$ for each modality i; and $\mathcal{D}$ is a *domain function* mapping members of $\mathbf{w}$ to non-empty sets.

Within the frame $\mathcal{F}$, the function $\mathcal{D}$ induces a set $\mathcal{D}(\mathcal{F})$, called the *domain of the frame*, as $\cup\{\mathcal{D}(w) \mid w \in \mathbf{w}\}$. In order to simplify the treatment of constant symbols, it is also convenient to define a set of objects that all the domains of the different possible worlds have in common, the *common domain of the frame* $\mathcal{F}$: $\mathcal{C}(\mathcal{F}) = \cap\{\mathcal{D}(w) \mid w \in \mathbf{w}\}$. We effectively insist that $\mathcal{C}(\mathcal{F})$ be non-empty as well, since CONST is non-empty and each symbol in CONST must be interpreted by an element of $\mathcal{C}(\mathcal{F})$.

The intermediate level of frames is useful in characterizing the meanings of modal operators and modal quantification. In particular, simply by putting constraints on $\mathcal{R}_i$ or on $\mathcal{D}$ at the level of frames, we can obtain representative classes of models in which certain general patterns of inference are validated. The constraints we will avail ourselves of are introduced in Definition 2.2.

Definition 2.2. Let $\langle \mathbf{w}, \mathcal{R}, \mathcal{D}\rangle$ be a *frame*. We say the frame is:

—*reflexive* at i if $w\mathcal{R}_i w'$ for every $w \in \mathbf{w}$;

—*transitive* at i if, for any $w, w'' \in \mathbf{w}$, $w\mathcal{R}_i w''$ whenever there is a $w' \in \mathbf{w}$ such that $w\mathcal{R}_i w'$ and $w'\mathcal{R}_i w''$;

—*narrowing* from i to j if $w\mathcal{R}_j w'$ implies $w\mathcal{R}_i w'$ for all $w, w' \in \mathbf{w}$;

—*increasing domain* if for all $w, w' \in \mathbf{w}$, $\mathcal{D}(w) \subseteq \mathcal{D}(w')$ whenever there is some accessibility relationship $w\mathcal{R}_i w'$.

Our scheme for using the constraints of Definition 2.2 depends on establishing a regime for the m modalities in the language, describing the inferences that should relate them. The regime is defined as follows.

Definition 2.3 *Regime*. A *regime* is a tuple $\langle \mathcal{A}, \mathcal{N}, \mathcal{Q}\rangle$, where: $\mathcal{A}$ is a function mapping each modality into one of the symbols K, K4, T and S4; $\mathcal{N}$ is a (strict) partial order on the modalities; and $\mathcal{Q}$ is the symbol *increasing*.

The reader will recognize the symbols in the image of $\mathcal{A}$ as the classic names for modal logics of a single modality. *S*4 is for modalities that are subject to (PI) and (VER). *T* is for modalities that are subject just to (VER). *K*4 is for modalities that are subject just to (PI). *K* is modalities subject to neither axiom. The interactions specified by (INC) are determined by the partial order on modalities: $j \leq i$ when $\Box_i p \supset \Box_j p$. This meaning for these symbols can be enforced by considering only frames that *respect* the given *regime*.

Definition 2.4 *Respect*. Let $\mathcal{F} = \langle \mathbf{w}, \mathcal{R}, \mathcal{D}\rangle$ be a frame, and let $\mathcal{S} = \langle \mathcal{A}, \mathcal{N}, \mathcal{Q}\rangle$ be a regime. We say $\mathcal{F}$ *respects* $\mathcal{S}$ whenever the following conditions are met for all modalities i and j:

—If $\mathcal{A}(i)$ is T or S4 then $\mathcal{R}_i$ is reflexive.

—If $\mathcal{A}(i)$ is K4 or S4 then $\mathcal{R}_i$ is transitive.

—If $j \leq i$ according to $\mathcal{N}$ then $\mathcal{F}$ is narrowing from i to j.

—If $\mathcal{Q}$ is *increasing*, then $\mathcal{F}$ is increasing domain.

From now on, we assume that some regime $\mathcal{S} = \langle \mathcal{A}, \mathcal{N}, \mathcal{Q}\rangle$ is fixed, and restrict our attention to frames that respect $\mathcal{S}$. Informally, now, a model consists of a frame together with an interpretation.

Definition 2.5 *Interpretation*. $\mathcal{I}$ is an *interpretation* in a frame $\langle \mathbf{w}, \mathcal{R}, \mathcal{D}\rangle$ if $\mathcal{I}$ satisfies these two conditions:

(1) $\mathcal{I}$ assigns to each n-place relation symbol p_i and each possible world $w \in \mathbf{w}$ some n-ary relation on the domain of the frame $\mathcal{D}(\mathcal{F})$.

(2) $\mathcal{I}$ assigns to each constant symbol c some element of the common domain of the frame $\mathcal{C}(\mathcal{F})$.

Thus we can define a *model* over $\mathcal{S}$ thus:

Definition 2.6 *Model*. A first-order *k-modal model* over a regime $\mathcal{S}$ is a tuple $\langle \mathbf{w}, \mathcal{R}, \mathcal{D}, \mathcal{I} \rangle$ where $\langle \mathbf{w}, \mathcal{R}, \mathcal{D} \rangle$ is a frame that respects $\mathcal{S}$ and $\mathcal{I}$ is an interpretation in $\langle \mathbf{w}, \mathcal{R}, \mathcal{D} \rangle$.

To define truth in a model, we need the usual notion of assignments and variants:

Definition 2.7 *Assignment*. Let $\mathcal{M} = \langle \mathbf{w}, \mathcal{R}, \mathcal{D}, \mathcal{I} \rangle$ be a model (that respects the regime $\mathcal{S}$). An *assignment* in $\mathcal{M}$ is a mapping g that assigns to each variable x some member $g(x)$ of the domain of the frame of the model $\mathcal{D}(\langle \mathbf{w}, \mathcal{R}, \mathcal{D} \rangle)$.

In proofs, we interpret formulas not just in the ordinary language $L(C)$ with a given set of modalities, relations, constants and variables, but in an expanded language $L(C \cup P)$ which also includes a set P of first-order *parameters*; we will want to use the same models for this interpretation. Over $L(C \cup P)$, we suppose that an assignment in $\mathcal{M}$ also assigns some member $g(p)$ of the domain of the frame of $\mathcal{M}$ to each parameter p in P.

Definition 2.8 *Variants*. Let g and g' be two assignments in a model $\mathcal{M} = \langle \mathbf{w}, \mathcal{R}, \mathcal{D}, \mathcal{I} \rangle$; g' is an *x-variant of g at a world w* $\in \mathbf{w}$ if g and g' agree on all variables except possibly for x and $g'(x) \in \mathcal{D}(w)$.

Definition 2.9 *Truth in a model*. Let $\mathcal{M} = \langle \mathbf{w}, \mathcal{R}, \mathcal{D}, \mathcal{I} \rangle$ be a model. Then the formula A is *true at world w of model $\mathcal{M}$ on assignment g*—written $\mathcal{M}, w \Vdash_g A$—just in case the clause below selected by syntactic structure of A is satisfied:

—A is $p_i(t_1, \ldots, t_n)$: Then $\mathcal{M}, w \Vdash_g A$ just in case $\langle e_1, \ldots e_n \rangle \in \mathcal{I}(p_i, w)$, where for each t_i, e_i is $\mathcal{I}(t_i)$ if t_i is a constant and $g(t_i)$ otherwise.

—A is $B_1 \wedge B_2$: Then $\mathcal{M}, w \Vdash_g A$ just in case both $\mathcal{M}, w \Vdash_g B_1$ and $\mathcal{M}, w \Vdash_g B_2$.

—A is $B_1 \vee B_2$: Then $\mathcal{M}, w \Vdash_g A$ just in case either $\mathcal{M}, w \Vdash_g B_1$ or $\mathcal{M}, w \Vdash_g B_1$.

—A is $\Box_i B$: Then $\mathcal{M}, w \Vdash_g A$ just in case for every $w' \in \mathbf{w}$, if $w \mathcal{R}_i w'$ then $\mathcal{M}, w' \Vdash_g B$.

—A is $\forall x B$: Then $\mathcal{M}, w \Vdash_g A$ just in case for every x-variant g' of g at w, $\mathcal{M}, w \Vdash_{g'} B$.

—A is $\exists x B$: Then $\mathcal{M}, w \Vdash_g A$ just in case there is some x-variant g' of g at w with $\mathcal{M}, w \Vdash_{g'} B$.

By a *sentence* we mean a formula of $L(\text{CONST})$ in which no variables occur free. For any sentence A, model $\mathcal{M}$ and world w of $\mathcal{M}$, a simple induction on depth guarantees that $\mathcal{M}, w \Vdash_g A$ for some assignment g in $\mathcal{M}$ exactly when $\mathcal{M}, w \Vdash_g A$ for all assignments g in $\mathcal{M}$. In this case, we can write simply $\mathcal{M}, w \Vdash A$ and say that A is *true in $\mathcal{M}$ at w*.

Definition 2.10 *Valid*. Let A be a sentence and $\mathcal{M} = \langle \mathbf{w}, \mathcal{R}, \mathcal{D}, \mathcal{I} \rangle$ be a model. A is *valid in $\mathcal{M}$* if for every world $w \in \mathbf{w}$, $\mathcal{M}, w \Vdash A$. A is *valid* (on the regime $\langle \mathcal{A}, \mathcal{N}, \mathcal{Q} \rangle$) if A is valid in any model $\mathcal{M}$ that respects the regime.

2.3 Proof theory

We now present our basic deductive system—a cut-free path-based sequent calculus for multi-modal deduction which uses Herbrand terms to reason correctly about parameterized instances of formulas. Since this calculus represents our basic *lifted sequent calculus* for modal logic, we refer to it as SCL here. Our starting point is Theorem 2.16 that SCL provides a sound and complete characterization of valid formulas.

SCL has the advantage that inferences can be freely interchanged, allowing arbitrary proofs to be transformed easily into goal-directed proofs; we show in Theorem 2.23, presented in Section 2.4, how to obtain goal-directed proofs in this calculus. The very same flexibility of inference, however, means that this calculus neither respects nor represents the potential of modal inference to give proofs an explicitly modular structure.

The basic constituent in SCL is a *tracked, prefixed formula*. The formulas extend the basic languages $D(C)$ and $G(C)$ of definitions and goals defined in (1) by allowing additional terms—representing arbitrary witnesses of first order quantifiers, and arbitrary transitions of modal accessibility among possible worlds—to be introduced into formulas for the purposes of proof. We begin by assuming two countable sets of symbols: a set H of *first-order Herbrand functions* and Υ of *modal Herbrand functions*. We can now define sets P_H of *first-order Herbrand terms,* κ_Υ of *modal Herbrand terms,* and $\Pi(\kappa_\Upsilon)$ of *Herbrand prefixes* by mutual recursion:

Definition 2.11 *Herbrand terms and prefixes*. Assume that t_0 is a Herbrand prefix and let $t_1,\ldots,t_n$ be a sequence (possibly empty), where each t_i is either an element of C, a first-order Herbrand term, or a Herbrand prefix. Then if h is a first-order Herbrand function then $h(t_0,t_1,\ldots,t_n)$ is a *first-order Herbrand term*. If η is a modal Herbrand function then $\eta(t_0,t_1,\ldots,t_n)$ is a *modal Herbrand term*. A *Herbrand prefix* is any finite sequence of modal Herbrand terms.

The rationale behind the use of a Herbrand term $h(X)$ at an existential inference R goes like this. At existential inferences, we need to reason about a generic individual. We need to have a suitable representation for a generic individual for R. Regardless of the order in which inferences are applied in a sequent deduction, there will be some parameters that must occur in the sequent where R applies. For example, some parameters must appear here as a result of the instantiations that must take place in deriving the formula to which R applies. We must be sure that the individual we introduce for R is different from all these parameters. The terms X which are supplied as an argument to the Herbrand term $h(X)$ identify these parameters indirectly. The structure $h(X)$ therefore serves as a placeholder for a new parameter that could be chosen to be different from each of the terms in X. The structure $h(X)$ thus packs all the information required to allow the inferences in the proof to be reordered and an appropriate parameter chosen so that the inference at R is truly generic.

In modal deduction, of course, we need generic individuals at modal inferences as well as existential ones. Modal Herbrand inference therefore requires that we introduce Herbrand terms to describe transitions among possible worlds and Herbrand prefixes to name possible worlds, in addition to introducing first-order Herbrand terms to represent first-order parameters. In this case, the arguments X to Herbrand terms must mix first-order Herbrand terms and Herbrand prefixes, since logical formulas can encode dependencies among first-order and modal parameters.

A *prefixed formula* is now an expression of the form A^μ with A a formula and μ a Herbrand prefix—we use $D(C \cup P_H)^{\Pi(\kappa_\Upsilon)}$ and $G(C \cup P_H)^{\Pi(\kappa_\Upsilon)}$ to refer to prefixed definitions and prefixed goals. For Herbrand calculi, formulas must also be *tracked* to indicate the sequence of instantiations that has taken place in the derivation of the formula.

Definition 2.12 *Tracked expressions*. If E denotes the expressions of some class, then the *tracked expressions* of that class are expressions of the form e_I where e is an expression of E and I is a finite sequence (possibly empty) of elements of $C \cup P_H \cup \Pi(\kappa_\Upsilon)$.

We say that a tracked expression e_I *tracks* a term t just in case t occurs as a subterm of some term in I.

In order to reason correctly about multiple modal operators, we need to keep track of the kinds of accessibility that any modal transition represents. To endow the system with correct first-order reasoning on increasing domains, we also need to keep track of the worlds where first-order terms are introduced. We use the following notation to record these judgments: $\mu/\nu : i$ indicates that world ν is accessible from world μ by the accessibility relation for modality i; and $t : \mu$ indicates that the entity associated with term t exists at world μ.

It is convenient to keep track of this information by anticipating the restricted reasoning required for our fragment $L(C)$ and exploiting the structure of Herbrand terms, as follows. It is clear that there are countably many first-order Herbrand terms, Herbrand prefixes, and formulas in $L(C \cup P_H)$. We can therefore describe a correspondence as follows. If A is a formula of the form $\forall x B$ or $\exists x B$ and u is a natural number, we define a corresponding first-order Herbrand function h^u_A so that each first-order Herbrand function is h_A for some A and no first-order Herbrand function is h^u_A and h^v_B for distinct A and B or distinct u and v. Likewise, if A is a formula of the form $\Box_i B$ and u is a natural number, we define a corresponding modal Herbrand function η^u_A so that each modal Herbrand function is η^u_A for some A and no modal Herbrand function is η^u_A and η^v_B for distinct A and B or distinct u and v. (Indexing Herbrand functions by natural numbers means that adapting a Herbrand proof to respect an eigenvariable condition can be as simple as reindexing its Herbrand functions.) Now we have:

Definition 2.13 *Herbrand typings*. A *Herbrand typing for the language* $L(C \cup P_H)$ (under a correspondence as just described) is a set Ξ of statements, each of which takes one of two forms:

(1) $\mu/\mu\eta : i$ where: μ is a Herbrand prefix and η is a modal Herbrand term of the form $\eta^u_A(\mu, \ldots)$ and A is $\Box_i B$.
(2) $t : \mu$ where t is a first-order Herbrand term of the form $h(\mu, \ldots)$.

A sequence of modal and first-order Herbrand terms X determines a Herbrand typing Ξ_X, consisting of the appropriate $\mu/\mu\eta : i$ for each modal Herbrand term η that occurs in X (possibly as a subterm) and the appropriate $h : \mu$ for each first-order Herbrand term h that occurs in X (possibly as a subterm).

Definition 2.14 *Typings*. Suppose that Ξ is a Herbrand typing over a language $L(C \cup P)^{\Pi(\kappa)}$, and that $\mathcal{S} = \langle \mathcal{A}, \mathcal{N}, increasing \rangle$ is a modal regime. We define the relation that E is a *derived typing* from Ξ with respect to $\mathcal{S}$, written $\mathcal{S}, \Xi \triangleright E$, as the smallest relation satisfying the following conditions:

—(K). $\mathcal{S}, \Xi \triangleright \mu/\nu : i$ if $\mu/\nu : i \in \Xi$.

—(T). $\mathcal{S},\Xi \triangleright \mu/\mu : i$ if $\mathcal{A}(i)$ is T or S4, and μ occurs in Ξ.

—(4). $\mathcal{S},\Xi \triangleright \mu/\nu : i$ if $\mu/\mu' : i \in \Xi$, $\mathcal{S},\Xi \triangleright \mu'/\nu : i$, and $\mathcal{A}(i)$ is K4 or S4.

—(Inc). $\mathcal{S},\Xi \triangleright \mu/\nu : i$ if $\mathcal{S},\Xi \triangleright \mu/\nu : j$ and $j \leq i$ according to $\mathcal{N}$.

—(V). $\mathcal{S},\Xi \triangleright t : \mu$ if $t : \mu \in \Xi$.

—(I). $\mathcal{S},\Xi \triangleright t : \nu$ if $\mathcal{S},\Xi \triangleright \mu/\nu : i$ for some i and $\mathcal{S},\Xi \triangleright t : \mu$.

Inspection of these rules shows that $\mathcal{S},\Xi \triangleright \mu/\nu : i$ only if ν and μ occur in Ξ. Moreover, given these rules, an easy induction on the length of typing derivations gives that $\mathcal{S},\Xi \triangleright \mu/\nu : i$ only if $\nu = \mu\nu'$ for some prefix ν'. Thus, suppose that $\mathcal{S},\Xi \triangleright \mu/\nu : i$ for some Herbrand typing Ξ: each step in the derivation must concern some prefix of ν and thus $\mathcal{S},\Xi_\nu \triangleright \mu/\nu : i$. These invariants permit some simplifications in reasoning in the fragment $L(C \cup P)$ over more expressive modal regimes containing other modal operators and other uses of connectives.

These definitions allow us to describe the modal Herbrand sequent calculus precisely. This calculus, SCL, is given in Definition 2.15. Note that for this fragment of modal logic, it suffices to consider sequents of the form $\Delta \longrightarrow \Gamma$, where Δ is a multiset of prefixed definitions (from $D(C \cup P_H)^{\Pi(\kappa_\Upsilon)}$), and Γ is a multiset of prefixed goals (from $G(C \cup P_H)^{\Pi(\kappa_\Upsilon)}$). Note also that $\mathcal{S},\Xi \triangleright \mu/\nu : i$ only if ν is of the form $\mu\nu'$.

Definition 2.15 *Herbrand sequent calculus*. For basic first-order multi-modal Herbrand deductions in our fragment over a regime $\mathcal{S}$, we will use the sequent rules defined here, which comprise the system SCL. The rules consist of an axiom rule and recursive rules—each recursive rule relates a *base* sequent below to one or more *spur* sequents above; it applies to the base in virtue of an occurrence of a distinguished tracked, prefixed formula in the sequent; we refer to this as the *principal expression* or simply the *principal* of the inference. Similarly, each of the sequent rules introduces new expressions onto each spur, which we refer to as the *side expressions* of the rule. We will also refer to the two named expression occurrences at axioms as the *principal expressions* or *principals* of the axiom. Now we have:

(1) axiom—A atomic:

$$\Delta, A^\mu_X \longrightarrow \Gamma, A^\mu_Y$$

(2) conjunctive:

$$\frac{\Delta, A \wedge B^\mu_X, A^\mu_X, B^\mu_X \longrightarrow \Gamma}{\Delta, A \wedge B^\mu_X \longrightarrow \Gamma} (\wedge \rightarrow)$$

$$\frac{\Delta \longrightarrow \Gamma, A \vee B^\mu_X, A^\mu_X, B^\mu_X}{\Delta \longrightarrow \Gamma, A \vee B^\mu_X} (\rightarrow \vee)$$

$$\frac{\Delta, A^\mu_X \longrightarrow \Gamma, A \supset B^\mu_X, B^\mu_X}{\Delta \longrightarrow \Gamma, A \supset B^\mu_X} (\rightarrow\supset)$$

(3) disjunctive:

$$\frac{\Delta \longrightarrow \Gamma, A \wedge B^\mu_X, A^\mu_X \qquad \Delta \longrightarrow \Gamma, A \wedge B^\mu_X, B^\mu_X}{\Delta \longrightarrow \Gamma, A \wedge B^\mu_X} (\rightarrow \wedge)$$

$$\frac{\Delta, A \vee B^\mu_X, A^\mu_X \longrightarrow \Gamma \qquad \Delta, A \vee B^\mu_X, B^\mu_X \longrightarrow \Gamma}{\Delta, A \vee B^\mu_X \longrightarrow \Gamma} (\vee \rightarrow)$$

$$\frac{\Delta, A \supset B_X^\mu \longrightarrow A_X^\mu, \Gamma \qquad \Delta, A \supset B_X^\mu, B_X^\mu \longrightarrow \Gamma}{\Delta, A \supset B_X^\mu \longrightarrow \Gamma} (\supset\rightarrow)$$

(4) possibility—where η is $\eta^u_{\Box_i A}(\mu, X)$ for some u:

$$\frac{\Delta \longrightarrow \Gamma, \Box_i A_X^\mu, A_{X,\mu\eta}^{\mu\eta}}{\Delta \longrightarrow \Gamma, \Box_i A_X^\mu} (\rightarrow \Box)$$

(5) necessity—subject to the side condition $\mathcal{S}, \Xi_\nu \rhd \mu/\mu\nu : i$:

$$\frac{\Delta, \Box_i A_X^\mu, A_{X,\mu\nu}^{\mu\nu} \longrightarrow \Gamma}{\Delta, \Box_i A_X^\mu \longrightarrow \Gamma} (\Box \rightarrow)$$

(6) existential—subject to the side condition that h is $h^u_B(\mu, X)$ for B_X^μ the principal of the rule (either $\exists x A$ or $\forall x A$) and some u:

$$\frac{\Delta, \exists x A^\mu, A[h/x]^\mu_{X,h} \longrightarrow \Gamma}{\Delta, \exists x A_X^\mu \longrightarrow \Gamma} (\exists \rightarrow) \quad \frac{\Delta \longrightarrow \Gamma, \forall x A_X^\mu, A[h/x]^\mu_{X,h}}{\Delta \longrightarrow \Gamma, \forall x A_X^\mu} (\rightarrow \forall)$$

(7) universal—subject to the side condition $\mathcal{S}, \Xi_{t,\mu} \rhd t : \mu$:

$$\frac{\Delta, \forall x A_X^\mu, A[t/x]^\mu_{X,t} \longrightarrow \Gamma}{\Delta, \forall x A_X^\mu \longrightarrow \Gamma} (\forall \rightarrow) \quad \frac{\Delta \longrightarrow \Gamma, \exists x A_X^\mu, A[t/x]^\mu_{X,t}}{\Delta \longrightarrow \Gamma, \exists x A_X^\mu} (\rightarrow \exists)$$

A $\mathcal{S}$-proof or $\mathcal{S}$-derivation for a sequent $\Delta \longrightarrow \Gamma$ is a tree built by application of these inference figures (in such a way that any side conditions are met for regime $\mathcal{S}$), with instances of the axiom as leaves and with the sequent $\Delta \longrightarrow \Gamma$ at the root. A tree similarly constructed except for containing some arbitrary sequent S as a leaf is a *derivation from S*.

I state the correctness theorem for this proof theory in a way that highlights the continuity with previous work on modal logic, particularly [Fitting 1983].

THEOREM 2.16 SOUNDNESS AND COMPLETENESS. *Suppose there is an $\mathcal{S}$-proof for a sequent $\longrightarrow A$. Then A is valid. Conversely, if there is no $\mathcal{S}$-proof for the sequent $\longrightarrow A$ then there is a model $\mathcal{M}$ (that respects $\mathcal{S}$) and world w such that $\mathcal{M}, w \not\Vdash A$.*

PROOF. I merely sketch a proof here, which involves simply applying the standard techniques of [Fitting 1983; Lincoln and Shankar 1994]. It is convenient to prove an intermediate result, using slightly modified sequent calculus SCE which imposes an eigenvariable condition on the possibility and existential rules—u must be new. We can show the soundness of SCE by adapting the arguments presented in [Fitting 1983, 2.3] and [Fitting and Mendelsohn 1998, 5.3]. Meanwhile, we can follow [Fitting 1983] in developing the completeness argument in terms of *analytic consistency properties* for the modal language. This argument can be seen as a formalization of the motivation for sequent calculi in the systematic search for models. Now, modal formulas may be satisfied only in infinite models, so the completeness theorem effectively requires us to consider infinite sequences of applications of sequent rules. In moving to infinite sets in this way, we must formally move from deductions, viewed as syntactic objects, to a more abstract, algebraic characterization of sets of modal formulas.

We can now establish the correctness of SCL by syntactic methods, which relate SCL proofs to SCE proofs. Suppose Γ and Δ contains sentences of $L(\text{CONST})$ (labeled with the empty prefix). Completeness is immediate: if there is an SCE proof for $\Gamma \longrightarrow \Delta$, that very proof is also an SCL proof of $\Gamma \longrightarrow \Delta$. Conversely, the soundness theorem says that if there is an SCL proof of $\Gamma \longrightarrow \Delta$, then there is an SCE proof for $\Gamma \longrightarrow \Delta$. We establish this by simply adapting the general Herbrand theorem of [Lincoln and Shankar 1994] to SCE. The idea behind the soundness result is that the structure of Herbrand terms provides enough information to reconfigure an SCL proof (by an inductive process of interchanges of inference, like that considered next in Section 2.4) so that equivalents of the eigenvariable conditions are enforced. The SCL proof may then be reindexed to respect SCE's eigenvariable requirements. □

2.4 Permutability of inference and uniform proofs

Our syntactic methods for reasoning about derivations exploit *permutability of inference*—the general ability to transform derivations so that inferences are interchanged [Kleene 1951]. To develop the notion of permutability of inference, we need to make some observations about the SCL sequent rules. First, the reasoning that is performed in subderivations is reasoning about subformulas (and vice versa). That is, in any spur sequent, the occurrence of the principal expression and the side expression all correspond to—or as we shall say, *are based in*—the occurrence of the principal expression in the base sequent. Likewise, each of the remaining expressions in the spur *are based in* an occurrence of an identical expression in the base. Here, as in [Kleene 1951], we are assuming an *analysis* of each inference to specify this correspondence in the case where the same expression has multiple occurrences in the base or in a spur. Thus, our proof techniques, where they involve copying derivations, sometimes involve (implicit) reanalyses of inferences.

Now, in any derivation, the spur of one inference serves as the base for an *adjacent* inference or an axiom. We can therefore associate any tracked prefixed formula occurrence E in any sequent in the derivation with the occurrence in the root (or *end-sequent*) which E is based in. A similar notion can relate inferences, as follows. Suppose O is the inference at the root of a (sub)derivation, and L is another inference in the (sub)derivation. Then L *is based in* an expression E in the spur of O if the principal expression of L is based in E; L *is based in* O itself if E is a side expression of O. An important special case is that of an axiom based in an inference O. In effect, such an axiom marks a contribution that inference O contributes to completing the deduction.

To define interchanges of inference, we appeal to the two basic operations of *contraction* and *weakening*, which we cast as transformations on proofs. (In other proof systems, contraction and weakening may be introduced as explicit *structural rules*.)

LEMMA 2.17 WEAKENING. *Let $\mathcal{D}$ be an SCL proof, let Δ_0 be a finite multiset of tracked prefixed definitions and let Γ_0 be a finite multiset of tracked prefixed goals (in the same language as $\mathcal{D}$). Denote by $\Delta_0 + \mathcal{D} + \Gamma_0$ a derivation exactly like $\mathcal{D}$, except that where any node in $\mathcal{D}$ carries $\Delta \longrightarrow \Gamma$, the corresponding node in $\Delta_0 + \mathcal{D} + \Gamma_0$ carries $\Delta, \Delta_0 \longrightarrow \Gamma, \Gamma_0$. (When Δ_0 or Γ_0 is empty, we drop the corresponding $+$ from the notation.) Then $\Delta_0 + \mathcal{D} + \Gamma_0$ is also an SCL proof.*

LEMMA 2.18 CONTRACTION. *Let $\mathcal{D}$ be an SCL proof whose root carries $\Delta \longrightarrow \Gamma, E, E$. Then we can construct an SCL proof $\mathcal{D}'$ whose root carries $\Delta \longrightarrow \Gamma, E$, whose height is at most the height of $\mathcal{D}$ and where there is a one-to-one correspondence (also preserving order of inferences) that takes any inference of $\mathcal{D}'$ to an inference with the same principal and side expressions in $\mathcal{D}$. We can likewise transform an SCL proof $\mathcal{D}$ whose root carries $\Delta, E, E \longrightarrow \Gamma$ into an SCL proof $\mathcal{D}'$ whose root carries $\Delta, E \longrightarrow \Gamma$.*

These lemmas follow from straightforward induction on the structure of derivations. These consequences continue to hold, suitably adapted, for the intermediate proof systems that we will construct from SCL in later sections.

Now consider two adjacent inferences in a derivation, a base inference *R* and an inference *S* (whose base is a spur of *R*). If *S* is not based in *R*, we may replace the derivation rooted at the base of *R* by a new derivation of the same end-sequent in which *S* applies at the root, *R* applies adjacent, and the remaining subderivations are copied from the original derivation (but possibly weakened to reflect the availability of additional logical premises). Performing such a replacement constitutes an interchange of rules *R* and *S* and demonstrates the permutability of *R* over *S*; see [Kleene 1951]. SCL is formulated so that any such pair of inferences may be exchanged in this way.

We also observe that we can correctly introduce an abbreviation for goal occurrences of $\Box_i(A \supset B)$ by a single formula $(A >_i B)$ and the consolidation of corresponding inferences $(\rightarrow \Box_i)$ and $(\rightarrow\supset)$ into a single figure $(\rightarrow>_i)$, while retaining unrestricted interchange of inference. Again when the inference applies to principal A^{μ}_{X}, the figure is formulated using η for $\eta^{u}_{A}(\mu, X)$ as:

$$\frac{\Gamma, A^{\mu\eta}_{X,\mu\eta} \longrightarrow B^{\mu\eta}_{X,\mu\eta}, A >_i B^{\mu}_{X}, \Delta}{\Gamma \longrightarrow A >_i B^{\mu}_{X}, \Delta} \rightarrow>_i$$

We will refer to the calculus using $(\rightarrow>_i)$ in place of $(\rightarrow \Box_i)$ and $(\rightarrow\supset)$ as SCLI, and consider SCLI in the sequel.

[Miller 1994; 1996] uses Definition 2.19 to characterize *abstract logic programming languages*.

Definition 2.19. A cut-free sequent proof $\mathcal{D}$ is *uniform* if for for every subproof $\mathcal{D}'$ of $\mathcal{D}$ and for every non-atomic formula occurrence *B* in the right-hand side of the end-sequent of $\mathcal{D}'$ there is a proof $\mathcal{D}''$ that is equal to $\mathcal{D}'$ up to a permutation of inferences and is such that the base inference in $\mathcal{D}''$ introduces the top-level logical connective of *B*.

Definition 2.20. A logic with a sequent calculus proof system is an *abstract logic programming language* if restricting to uniform proofs does not lose completeness.

It is easy to show that the sequent calculi SCL and SCLI are abstract logic programming languages in this sense. In fact, by this definition *every* SCL or SCLI derivation is uniform.

To anticipate our analysis of permutability in later sections, let us introduce the notion of an *eager* derivation in SCL or SCLI.

Definition 2.21. Consider a derivation $\mathcal{D}$ containing a right inference *R* that applies to principal E. *R* is *delayed* exactly when there is a subderivation $\mathcal{D}'$ of $\mathcal{D}$ where: $\mathcal{D}'$ contains R; $\mathcal{D}'$ has a left inference *L* at the root; and the principal E of *R* is based in an occurrence of E in the end-sequent of $\mathcal{D}'$.

Consider this schematic diagram of such a subderivation $\mathcal{D}'$:

$$\cfrac{\cfrac{\vdots}{\ldots E \ldots}\ R}{\cfrac{\downarrow}{\ldots E \ldots}\ L}$$

On an intuitive conception of a sequent proof as a record of proof search constructed from root upwards, *R* is delayed in that we have waited in $\mathcal{D}$ to apply *R* until after consulting the program by applying L, when we might have applied R earlier. Thus, we will also say in the circumstances of Definition 2.21 that R is delayed *with respect to L*.

Definition 2.22. $\mathcal{D}$ is *eager* exactly when it contains no delayed applications of right rules.

By transforming any derivation $\mathcal{D}$ into an eager derivation $\mathcal{D}'$ by permutations of inferences, we make it clear that reasoning about goals can always precede reasoning with program statements, and thereby provide a starting point for further analysis of goal-directed proof search.

THEOREM 2.23. *Any SCL(I) derivation $\mathcal{D}$ is equal to an eager derivation $\mathcal{D}'$ up to permutations of inferences.*

PROOF. The argument follows [Kleene 1951, Theorem 2]. A double induction transforms each derivation into an eager one; the inner induction rectifies the final rule of a derivation whose subderivations are eager by an interchange of inferences (and induction) [Kleene 1951, Lemma 10]; the outer one rectifies a derivation by rectifying the furthest violation from the root (and induction). The argument is presented in its entirety in the electronic appendix to this article. □

3. MODULAR GOAL-DIRECTED PROOF SEARCH

3.1 Overview

Eager derivations do not make a satisfactory specification for goal-directed proof in a logic programming sense, because they do not embody a particularly directed search strategy. For one thing, eager derivations are free to work in parallel on different disjuncts of a goal using different program statements; in logic programming we want *segments* in which a single program statement and a single goal is in force. Moreover, eager derivations can reuse work across separate case analyses; in logic programming we want *blocks* where particular cases are investigated separately. Finally, because of their classical formulation, eager derivations do not enforce or exploit any modularity in their underlying logic. This section shows how to remedy these faults of eager derivations.

The result takes the form of an alternative sequent calculus SCLP which is equivalent to SCL. SCLP enforces a strictly goal-directed proof search through the structure of its inferences. First, SCLP sequents take the form

$$\Gamma; U \longrightarrow V; \Delta$$

We understand Γ to specify the global program and Δ to specify the global restart goals; both are multisets of tracked, prefixed formulas. U is at most one tracked, prefix formula, representing the current program statement; V is at most one tracked, prefixed formula, representing the current goal.

Logical rules apply only to the current program statement and the current goal. In addition, if there is a current program statement U then the current goal V must be an atomic formula. Thus, the interpreter first breaks the goal down into its components. Once an atomic goal is derived, the program is consulted; the selected program statement is decomposed and matched against the current goal by applicable logical rules. The form of the $(\supset\rightarrow)$ figure ensures that the interpreter continues to work on at most one goal at any time; this gives SCLP proofs their segment structure. Meanwhile, the form of the $(\vee\rightarrow)$

figures specify no current goal in its second case. The new current goal can then be chosen flexibly from possible restart goals. This gives SCLP proofs their block structure.

The new inferences are presented in Definition 3.1 and 3.2. Definition 3.1 shows the rules for decomposing program statements; Definition 3.2 shows the rules for decomposing goals.

Definition 3.1 *Logic programming calculus—programs*. The following inference figures describe the logic programming sequent calculus SCLP as it applies to program statements.

(1) axiom—A atomic:

$$\Gamma; A^{\nu} \longrightarrow A^{\nu}; \Delta$$

(2) decision (program consultation)—again A atomic:

$$\frac{\Gamma, P^{\mu}_{X}; P^{\mu}_{X} \longrightarrow A^{\nu}_{Y}; \Delta}{\Gamma, P^{\mu}_{X}; \longrightarrow A^{\nu}_{Y}; \Delta}\ \text{decide}$$

(3) conjunctive:

$$\frac{\Gamma; P^{\mu}_{X} \longrightarrow A^{\nu}_{Y}; \Delta}{\Gamma; P \wedge Q^{\mu}_{X} \longrightarrow A^{\nu}_{Y}; \Delta}\ \wedge\!\rightarrow_{L}$$

$$\frac{\Gamma; Q^{\mu}_{X} \longrightarrow A^{\nu}_{Y}; \Delta}{\Gamma; P \wedge Q^{\mu}_{X} \longrightarrow A^{\nu}_{Y}; \Delta}\ \wedge\!\rightarrow_{R}$$

(4) disjunctive:

$$\frac{\Gamma; P^{\mu}_{X} \longrightarrow A^{\nu}_{Y}; \Delta \qquad \Gamma, Q^{\mu}_{X}; \longrightarrow; \Delta}{\Gamma; P \vee Q^{\mu}_{X} \longrightarrow A^{\nu}_{Y}; \Delta}\ \vee\!\rightarrow_{L}$$

$$\frac{\Gamma; Q^{\mu}_{X} \longrightarrow A^{\nu}_{Y}; \Delta \qquad \Gamma, P^{\mu}_{X}; \longrightarrow; \Delta}{\Gamma; P \vee Q^{\mu}_{X} \longrightarrow A^{\nu}_{Y}; \Delta}\ \vee\!\rightarrow_{R}$$

(5) implication:

$$\frac{\Gamma; \longrightarrow Q^{\mu}_{X}; \Delta \qquad \Gamma; P^{\mu}_{X} \longrightarrow A^{\nu}_{Y}; \Delta}{\Gamma; Q \supset P^{\mu}_{X} \longrightarrow A^{\nu}_{Y}; \Delta}\ \supset\!\rightarrow$$

(6) necessity—subject to the side condition that there is a typing derivation $\mathcal{S}, \Xi_{\nu} \triangleright \mu/\mu\nu : i$:

$$\frac{\Gamma; P^{\mu\nu}_{X,\mu\nu} \longrightarrow A^{\nu'}_{Y}; \Delta}{\Gamma, \Box_{i} P^{\mu}_{X} \longrightarrow A^{\nu'}_{Y}; \Delta}\ \Box_{i} \rightarrow$$

(7) existential—subject to the side condition that h is $h^{u}_{\exists xP}(\mu, X)$ for some u:

$$\frac{\Gamma; P[h/x])^{\mu}_{X,h} \longrightarrow A^{\nu}_{Y}; \Delta}{\Gamma; \exists x.P^{\mu}_{X} \longrightarrow A^{\nu}_{Y}; \Delta}\ \exists \rightarrow$$

(8) universal—subject to the side condition that there is a typing derivation $\mathcal{S}, \Xi_{t,\mu} \triangleright t : \mu$:

$$\frac{\Gamma; P[t/x]^{\mu}_{X,t} \longrightarrow A^{\nu}_{Y}; \Delta}{\Gamma; \forall x.P^{\mu}_{X} \longrightarrow A^{\nu}_{Y}; \Delta}\ \forall \rightarrow$$

Definition 3.2 *Logic programming calculus—goals*. The following inference figures describe the logic programming sequent calculus SCLP as it applies to goals.

(1) restart:

$$\frac{\Gamma; \longrightarrow G^{\nu}_{X}; G^{\nu}_{X}, \Delta}{\Gamma; \longrightarrow ; G^{\nu}_{X}, \Delta}\ \text{restart}$$

(2) conjunctive goals:

$$\frac{\Gamma; \longrightarrow F^{\mu}_{X}; \Delta \qquad \Gamma; \longrightarrow G^{\mu}_{X}; \Delta}{\Gamma; \longrightarrow F \wedge G^{\mu}_{X}; \Delta} \rightarrow \wedge$$

(3) disjunctive goals:

$$\frac{\Gamma; \longrightarrow F^{\mu}_{X}; \Delta}{\Gamma; \longrightarrow F \vee G^{\mu}_{X}; \Delta} \rightarrow \vee_L$$

$$\frac{\Gamma; \longrightarrow G^{\mu}_{X}; \Delta}{\Gamma; \longrightarrow F \vee G^{\mu}_{X}; \Delta} \rightarrow \vee_R$$

(4) necessary goals—where η is $\eta^{u}_{A}(\mu, X)$ for A^{μ}_{X} the principal of the rule and for some u for which η^{u}_{A} does not occur in Δ or Γ:

$$\frac{\Gamma, F^{\mu\eta}_{X,\mu\eta}; \longrightarrow G^{\mu\eta}_{X,\mu\eta}; G^{\mu\eta}_{X,\mu\eta}, \Delta}{\Gamma; \longrightarrow F >_i G^{\mu}_{X}; \Delta} \rightarrow \Box_i \supset$$

$$\frac{\Gamma; \longrightarrow G^{\mu\eta}_{X,\eta}; G^{\mu\eta}_{X,\mu\eta}, \Delta}{\Gamma; \longrightarrow \Box_i G^{\mu}_{X}; \Delta} \rightarrow \Box_i$$

(5) universal goals—subject to the side condition that h is $h^{u}_{\forall x G}(\mu, X)$ for some u:

$$\frac{\Gamma; \longrightarrow G[h/x]^{\mu}_{X,h}; \Delta}{\Gamma; \longrightarrow \forall x. G^{\mu}_{X}; \Delta} \rightarrow \forall$$

(6) existential goals—subject to the side condition that there is a typing derivation $\mathcal{S}, \Xi_{t,\mu} \triangleright t : \mu$:

$$\frac{\Gamma; \longrightarrow G[t/x]^{\mu}_{X,t}; \Delta}{\Gamma; \longrightarrow \exists x. G^{\mu}_{X}; \Delta} \rightarrow \exists$$

Inspection of the figures of Definitions 3.1 and 3.2 reveals the following generalization of modularity and locality: in any derivation, the label of the current program statement must be a prefix of the label of the current goal. Moreover, goal labels are always extended with novel symbols, because of the eigenvariable condition in the $(\rightarrow \Box)$ figure. Inductively, these facts determine a strong invariant—consider a block beginning with a restart inference whose spur is

$$\Gamma; \longrightarrow G^{\nu}_{X}; \Delta$$

and consider any expression P^{μ}_{Y} in Γ. If μ is not a prefix of ν, then μ will not be a prefix of the label of any goal formula in the block. Thus P^{μ}_{Y} cannot be used in the block. (Compare [Stone 1999, Lemma 2].)

This is why the (restart) rule of SCLP can be made modular, so that it limits the work that is reanalyzed to the scope of the ambiguity just introduced. We must simply show that the new disjunct will contribute to its restart goal. In particular, define canceled blocks as in Definition 3.4.

$$\boxed{2'}\quad \dfrac{\dfrac{\dfrac{\dfrac{\ldots;C \longrightarrow C;F}{\ldots,C;\longrightarrow C;F}\,(\text{decide}) \quad \ldots;F \longrightarrow F;F}{C,\ldots;C\supset F \longrightarrow F;F}\,(\supset\rightarrow)}{C\supset F,C\ldots;\longrightarrow F;F}\,(\text{decide})}{A\vee B,C\vee D,A\supset F,C\supset F,(B\wedge D)\supset F,B,C;\longrightarrow;F}\,(\text{restart})$$

$$\boxed{3'}\quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\ldots;B \longrightarrow B;F}{B,\ldots;\longrightarrow B;F}\,(\text{decide}) \quad \dfrac{\dfrac{\ldots;D \longrightarrow D;F \quad \boxed{2'}}{\ldots;C\vee D \longrightarrow D;F}\,(\vee\rightarrow_R)}{C\vee D,\ldots;\longrightarrow D;F}\,(\text{decide})}{C\vee D,B,\ldots;\longrightarrow B\wedge D;F}\,(\rightarrow\wedge) \quad \ldots;F \longrightarrow F;F}{C\vee D,B,\ldots;(B\wedge D)\supset F \longrightarrow F;F}\,(\supset\rightarrow)}{C\vee D,(B\wedge D)\supset F,B,\ldots;\longrightarrow F;F}\,(\text{decide})}{A\vee B,C\vee D,A\supset F,C\supset F,(B\wedge D)\supset F,B;\longrightarrow;F}\,(\text{restart})$$

$$\boxed{1}\quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\ldots;A \longrightarrow A;F \quad \boxed{3'}}{\ldots;A\vee B \longrightarrow A;F}\,(\vee\rightarrow_L)}{A\vee B,\ldots;\longrightarrow A;F}\,(\text{decide}) \quad \ldots;F \longrightarrow F;F}{A\vee B,\ldots;A\supset F \longrightarrow F;F}\,(\supset\rightarrow)}{A\vee B,A\supset F,\ldots;\longrightarrow F;F}\,(\text{decide})}{A\vee B,C\vee D,A\supset F,C\supset F,(B\wedge D)\supset F;\longrightarrow;F}\,(\text{restart})$$

Fig. 6. The SCLP presentation of the proof of Figure 5.

Definition 3.3 *Linked*. An expression E in a sequent in an SCLU derivation $\mathcal{D}$ is *linked* if the principal formula of an axiom in the same block of $\mathcal{D}$ as that sequent is based in E. An inference R is *linked* in $\mathcal{D}$ if some side expression of R is linked in each spur of R. A derivation or block is *linked* iff all of the inferences in it are linked.

Definition 3.4 *Canceled*. A block is *canceled* if it contains the root of $\mathcal{D}$, or if the side expression E of the $(\vee\rightarrow)$ inference whose spur is the root of the block is linked.

Thus a canceled block includes a use of any disjunctive case introduced in the block. The key fact about SCLP is that it suffices to consider only canceled blocks in proof search.

THEOREM 3.5. *Let Γ and Δ be multisets of tracked prefixed expression in which each formula is tracked by the empty set and prefixed by the empty prefix. There is a proof of $\Gamma \longrightarrow \Delta$ in SCL exactly when there is a proof of $\Gamma;\longrightarrow;\Delta$ in SCLP in which every block is canceled.*

The discussion of the following subsections represents an outline of the proof of this result. The strategy is to transform eager proofs from SCL to SCLP by a series of refinements of sequent rules that make the logic programming strategy explicit. We give force to the idea that the interpreter has a current goal and current program statement, in Section 3.2. Then we create blocks for case analysis, in Section 3.3. Finally, we enforce modularity, in Section 3.4. See also the electronic appendix.

Figure 6 shows how the proof of Figure 5 is recast in SCLP. Figure 6 extends Figure 5 to make the bookkeeping of goal-directed proof explicit. In Figure 6, the informal underline of Figure 5 is gone, and instead the current goal and the current program statement are displayed at distinguished positions in sequents. New (restart) and (decide) inferences

$$\boxed{5}\quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dots;B^{\alpha} \longrightarrow B^{\alpha};\dots}{\dots,B^{\alpha};\longrightarrow B^{\alpha};\dots}\,(\text{decide}) \qquad \dots;A^{\alpha}\longrightarrow A^{\alpha};\dots}{B^{\alpha},\dots;B\supset A^{\alpha}\longrightarrow A^{\alpha};\dots}\,(\supset\rightarrow)}{B^{\alpha},\dots;\Box(B\supset A)\longrightarrow A^{\alpha};\dots}\,(\Box\rightarrow)}{\Box(B\supset A),B^{\alpha};\longrightarrow A^{\alpha};\dots}\,(\text{decide})}{\Box(A\vee B),\Box(B\supset A),\Box(C\vee D),\Box(D\supset C),B^{\alpha};\longrightarrow;A^{\alpha},\Box A\wedge\Box C}\,(\text{restart})}{}$$

$$\boxed{6}\quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dots;D^{\beta} \longrightarrow D^{\beta};\dots}{\dots,D^{\beta};\longrightarrow D^{\beta};\dots}\,(\text{decide}) \qquad \dots;C^{\beta}\longrightarrow C^{\beta};\dots}{D^{\beta},\dots;D\supset C^{\beta}\longrightarrow C^{\beta};\dots}\,(\supset\rightarrow)}{D^{\beta},\dots;\Box(D\supset C)\longrightarrow C^{\beta};\dots}\,(\Box\rightarrow)}{\Box(D\supset C),D^{\beta};\longrightarrow C^{\beta};\dots}\,(\text{decide})}{\Box(A\vee B),\Box(B\supset A),\Box(C\vee D),\Box(D\supset C),D^{\beta};\longrightarrow;C^{\beta},\Box A\wedge\Box C}\,(\text{restart})$$

$$\boxed{4}\quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dots;A^{\alpha}\longrightarrow A^{\alpha};\dots \quad \boxed{5}}{\dots;A\vee B^{\alpha}\longrightarrow A^{\alpha};\dots}\,(\vee\rightarrow_L)}{\dots;\Box(A\vee B)\longrightarrow A^{\alpha};\dots}\,(\Box\rightarrow)}{\Box(A\vee B),\dots;\longrightarrow A^{\alpha};\dots}\,(\text{decide})}{\Box(A\vee B),\dots;\longrightarrow \Box A;\dots}\,(\rightarrow\Box) \quad \dfrac{\dfrac{\dfrac{\dfrac{\dots;C^{\beta}\longrightarrow C^{\beta};\dots \quad \boxed{6}}{\dots;C\vee D^{\beta}\longrightarrow C^{\beta};\dots}\,(\vee\rightarrow_L)}{\dots;\Box(C\vee D)\longrightarrow C^{\beta};\dots}\,(\Box\rightarrow)}{\Box(C\vee D),\dots;\longrightarrow C^{\beta};\dots}\,(\text{decide})}{\Box(C\vee D),\dots;\longrightarrow \Box C;\dots}\,(\rightarrow\Box)}{\Box(A\vee B),\Box(C\vee D),\dots;\longrightarrow \Box A\wedge\Box C;\dots}\,(\rightarrow\wedge)}{\Box(A\vee B),\Box(B\supset A),\Box(C\vee D),\Box(D\supset C);\longrightarrow;\Box A\wedge\Box C}\,(\text{restart})$$

Fig. 7. The SCLP presentation of the proof of Figure 2. We suppress tracking of formulas and hide the internal structure of Herbrand terms.

mark the consideration of new goals and new program statements. Of course, the logical content of the two inferences is identical. Applying Definition 3.4, block $\boxed{1}$ is canceled because it contains the root; there is no new disjunct to discharge here. Block $\boxed{3'}$ is canceled: the inference whose spur is the root of block $\boxed{3'}$ is the $(\vee\rightarrow_L)$ and its side expression is an occurrence of B, the new disjunct in the block. This occurrence is linked in the block because of the leftmost axiom $\dots;B\longrightarrow B;F$ which is based in it; the inference $(\vee\rightarrow_L)$ is linked in the block for the same reason. Similarly block $\boxed{2'}$ is canceled because the new disjunct C (the side expression of the $(\vee\rightarrow_R)$ inference whose spur is the root of block $\boxed{2'}$) contributes to the leftmost axiom $\dots;C\longrightarrow C;F$ in the block.

Figure 7 shows how the proof of Figure 2 is recast in SCLP. The most dramatic change here is that the inferences of Figure 7 are segmented out into three blocks. Another change is the discipline of explicit scope; we introduce a suitable term α to represent the generic context in which we prove $\Box A$ and another suitable term β to represent the generic context in which we prove $\Box C$. Correspondingly, we transition to α in using $\Box(A\vee B)$ and transition to β in using $\Box(C\vee D)$. In the (restarts) of $\boxed{5}$ and $\boxed{6}$ the changes interact. In $\boxed{5}$ we pick the modular restart A^{α} in order to permit a contribution by the new assumption B^{α}. In $\boxed{6}$ we pick the modular restart C^{β} in order to permit a contribution by the new assumption D^{β}.

3.2 Segment structure

Our first task is to formalize goal-directed search that directs attention to a single goal at a time. To distinguish such goals, we begin with a trick that for now is purely formal—introducing an *articulated* SCLI. We represent assumptions as a pair $\Pi;\Gamma$ with Π encoding the global program and Γ encoding local program statements; eventually local statements will be processed only in the current segment and then discarded. (Compare the similar notation and treatment from [Girard 1993].) Similarly, we represent goals as a pair $\Delta;\Theta$, with Θ encoding the restart goals and Δ encoding the local goals; ultimately, we will also describe inference rules which will discard Δ between segments. With this representation, principal formulas of logical rules are local formulas, in Γ or Δ; so are the side formulas—with these exceptions: the $(\rightarrow\Box)$ and $(\rightarrow>)$ rules augment Π instead of Γ (when they add a new program statement) and Θ instead of Δ (when they add new restart goals).

New (decide) and (restart) rules keep this change general; they allow a global formula—a program statement or restart goal—to be selected and added to the local state.

$$\frac{\Pi,A^{\mu}_{X};\Gamma,A^{\mu}_{X} \longrightarrow \Delta;\Theta}{\Pi,A^{\mu}_{X};\Gamma \longrightarrow \Delta;\Theta}\text{(decide)} \qquad \frac{\Pi;\Gamma \longrightarrow \Delta,G^{\mu}_{X};\Theta,G^{\mu}_{X}}{\Pi;\Gamma \longrightarrow \Delta;\Theta,G^{\mu}_{X}}\text{(restart)}$$

LEMMA 3.6 ARTICULATION. *Every SCLI deduction can be converted into an articulated SCLI deduction with an end-sequent of the form* $\Pi;\longrightarrow;\Theta$ *in such a way that if the initial derivation is eager then so is the resulting derivation (and vice versa).*

PROOF. Straightforward structural induction. □

The next step is to introduce an inference figure $(\supset\rightarrow^{S})$ that imposes a *segment* structure on derivations, thus:

$$\frac{\Pi;\longrightarrow A^{\mu}_{X},\Delta;\Theta \qquad \Pi;\Gamma,A\supset B^{\mu}_{X},B^{\mu}_{X}\longrightarrow \Delta;\Theta}{\Pi;\Gamma,A\supset B^{\mu}_{X}\longrightarrow \Delta;\Theta}(\supset\rightarrow^{S})$$

Definition 3.7 *Segment*. A *segment* in a derivation $\mathcal{D}$ is a maximal tree of contiguous inferences in which the left subtree of any $(\supset\rightarrow^{S})$ inference is omitted.

The distinctive feature of the $(\supset\rightarrow^{S})$ figure is that the local results inferred from the program are discarded in the subderivation where the new goal is introduced. In an eager derivation, this will begin a new segment where first the new goal will be considered and then a new program statement will be selected to establish that goal.

We will define two calculi using $(\supset\rightarrow^{S})$. The first, SCLS, eliminates the $(\supset\rightarrow)$ inference of the articulated SCLI and instead has $(\supset\rightarrow^{S})$. The second, SCLV, is a calculus like the articulated SCLI but also allows $(\supset\rightarrow^{S})$; $(\supset\rightarrow)$ and $(\supset\rightarrow^{S})$ can appear anywhere in an SCLV derivation. We introduce SCLV to facilitate the incremental transformation of articulated SCLI proofs into SCLS proofs.

LEMMA 3.8. *An eager articulated SCLI derivation whose end-sequent is of the form*

$$\Pi;\rightarrow\Delta;\Theta$$

can be transformed to an eager SCLS derivation of the same end-sequent.

PROOF. We proceed with an inductive construction that iteratively adapts the lowest $(\supset\rightarrow)$ inference L in the proof to match the $(\supset\rightarrow^{S})$ figure. The construction first copies inferences into the left subderivation at L to reintroduce local goals while keeping the reasoning eager; with this subderivation, we no longer preserve work across the segment boundary. The extended subderivation is then transformed inductively to use SCLS inferences (we have not copied L). L now satisfies $(\supset\rightarrow^S)$ and no new $(\supset\rightarrow)$ inferences have been introduced, so we can proceed to transform the resulting derivation to SCLS. The argument is presented in its entirety in the electronic appendix to this article. □

3.3 Block structure

We now revise how we perform case analysis from assumptions. We introduce new rules where all local work is discarded in the subderivation written on the right. This corresponds to a sequent of the form $\Pi; \longrightarrow; \Theta$. In addition, some *global* work may be discarded in the right subderivation; this helps clarify the structure of derivations. Accordingly, there may be additional formula occurrences Π' and Θ' in the base sequent that are not copied up to the right subderivation. Finally, the right subderivation may address either the (textually) first disjunct or the second disjunct. This leads to the two inference figures below.

$$\frac{\Pi,\Pi';\Gamma,A\vee B^{\mu},A^{\mu} \longrightarrow \Delta;\Theta,\Theta' \qquad \Pi,B^{\mu};\longrightarrow;\Theta}{\Pi,\Pi';\Gamma,A\vee B^{\mu} \longrightarrow \Delta;\Theta,\Theta'}\vee\rightarrow^{B}_{L}$$

$$\frac{\Pi,\Pi';\Gamma,A\vee B^{\mu},B^{\mu} \longrightarrow \Delta;\Theta,\Theta' \qquad \Pi,A^{\mu};\longrightarrow;\Theta}{\Pi,\Pi';\Gamma,A\vee B^{\mu} \longrightarrow \Delta;\Theta,\Theta'}\vee\rightarrow^{B}_{R}$$

We call these inferences *blocking* $(\vee\rightarrow)$ inferences, or $(\vee\rightarrow^{B})$ inferences. We will appeal to two calculi in which these inferences appear. The first, SCLU, permits both ordinary $(\vee\rightarrow)$ and $(\vee\rightarrow^{B})$ inferences, without restriction. SCLU is convenient for describing transformations between proofs. The second, SCLB, permits $(\vee\rightarrow^{B})$ inferences but not ordinary $(\vee\rightarrow)$ inferences.

Blocks are more than just boundaries in the proof; they provide a locus for enforcing modularity. We will ensure that a disjunct contributes inferences to the new block where it is introduced. Thanks to this contribution, we can narrow down the choice of goals to restart in a modular way.

This result is made possible only by maintaining the right structure as we introduce $(\vee\rightarrow^{B})$ inferences. We use path prefixes to make explicit connections between program statements and any goals that they help establish. The key notions are *spanning*, *simplicity* and *balance* for sequents. Spanned, simple and balanced sequents represent a consistent evolution of the state of proof search, which records a full set of restart goals and the corresponding assumptions, with no redundancy.

Definition 3.9 *Carrier*. The *carrier* of a non-empty Herbrand prefix $\mu\eta$ is $B^{\mu\eta}_{X,\mu\eta}$ if η is $\eta^{u}_{A>_i B}(\mu,X)$ and otherwise, when η is $\eta^{u}_{\Box_i A}(\mu,X)$, is $A^{\mu\eta}_{X,\mu\eta}$.

Definition 3.10 *Spanned*. Say one multiset of tracked prefixed formulas, Π, is *spanned* by another, Θ, if for every expression occurrence A^{μ}_{X} in Π and every non-empty prefix ν of μ there is an occurrence of the carrier of ν in Θ. It is easy to see there is a minimal set Θ that spans Π and that such Θ spans itself. A sequent $\Pi;\Gamma \longrightarrow \Delta;\Theta$ is *spanned* if Π is spanned by Θ, Γ is spanned by Θ, Δ is spanned by Θ and Θ is spanned by Θ. A derivation or block is *spanned* if every sequent in it is spanned.

Definition 3.11 *Simple*. A multiset Ψ is *simple* if no expression occurs multiple times in Ψ; a sequent of the form $\Pi;\Gamma \longrightarrow \Delta;\Theta$ is *simple* if Π and Θ are simple. A derivation or block is *simple* iff every sequent in it is simple.

Definition 3.12 *Balanced*. A pair of multisets of tracked, prefixed formulas Π, Θ is *balanced* if

—for any $\eta = \eta^u_{B >_i C}(\mu, X)$, η occurs in Θ exactly when the expression $B^{\mu\eta}_{X,\mu\eta}$ occurs in Π and exactly when the expression $C^{\mu\eta}_{X,\mu\eta}$ occurs in Θ; and

—for any $\eta = \eta^u_{\Box A}(\mu, X)$, η occurs in Θ exactly when the expression $A^{\mu}_{X,\mu\eta}$ occurs in Θ.

A sequent $\Pi; \Gamma \longrightarrow \Delta; \Theta$ is *balanced* if the pair Π, Θ is balanced. A block or derivation is *balanced* if every sequent in the block is balanced.

We use the notion of an *isolated block* to obtain an even stronger characterization of proof search that proceeds in a well-regimented way. In an isolated block, the only expressions preserved across a blocking inference are those that are in some sense intrinsic to the restart problem created by that inference. Specifically, each nested block must begin with the same end-sequent as the outer block, except for additional program statements that have to be added in order to introduce the newly-assumed disjunct, and the further goal and program statements required to obtain a balanced and spanned sequent.

Definition 3.13 *Isolated*. Let $\mathcal{D}$ be an SCLU derivation, and let $\mathcal{B}$ be a block of $\mathcal{D}$. Write the end-sequent of $\mathcal{B}$ as $\Pi; \Gamma \longrightarrow \Delta; \Theta$ and consider the right subproof of some $(\vee \rightarrow^B)$ inference L at the boundary of $\mathcal{B}$—it will have an end-sequent of the form $\Pi', E; \longrightarrow; \Theta'$ where E is the side-expression of L. Under these conditions, the *exported* expressions in Π', Π'_e, consist of the occurrences of expressions F in Π' such that either is F based in an occurrence of F in Π or is based in an occurrence of F as the side expression of an inference in which E is also based.

$\mathcal{B}$ is *isolated* if the right subproof of each $(\vee \rightarrow^B)$ inference L at the boundary of $\mathcal{B}$ has an end-sequent of the form $\Pi', E; \longrightarrow; \Theta'$ meeting the following conditions: E is the side-expression of L; Θ' is the minimal multiset of expressions which spans Π'_e, E, Θ and includes Θ; and Π' is the smallest multiset including Π'_e, E for which Π', Θ' is balanced. $\mathcal{D}$ is *isolated* iff every block of $\mathcal{D}$ is isolated.

Isolation allows us to keep close tabs on the uses of formulas within blocks, which is important for establishing modularity later. In particular, isolation provides a key notion in formalizing the obvious fact that an inference that makes no contribution to an SCLU derivation can be omitted.

Finally at this stage, we refine the form of proofs which we are willing to count as goal-directed. Now it will often happen that, while each block of a derivation may be eager, the derivation as a whole will not be eager. As observed in [Nadathur and Loveland 1995], derivations with blocks can nevertheless be seen as eager throughout by reconstructing the (restart) rule as backchaining against the negation of a subgoal. But we will simply consider *blockwise eager* derivations from now on.

Definition 3.14 *Blockwise delayed*. R is *blockwise delayed* exactly when there is a tree of contiguous inferences $\mathcal{D}'$ within a single block of $\mathcal{D}$ where: $\mathcal{D}'$ contains R; $\mathcal{D}'$ has a left inference L at the root; and the principal E of R is based in an occurrence of E in the end-sequent of $\mathcal{D}'$.

Definition 3.15 *Blockwise eager*. D is *blockwise eager* exactly when it contains no blockwise delayed applications of right rules.

Obviously, we can use weakening to transform an SCLB or SCLU derivation into a SCLS derivation, so the blocking inference figures are sound. The completeness of SCLB is a consequence of Lemma 3.16.

LEMMA 3.16. *We are given a blockwise eager SCLS derivation $\mathcal{D}$ whose end-sequent is spanned and balanced and takes the form:*

$$\Pi; \longrightarrow; \Theta$$

We transform $\mathcal{D}$ into a blockwise eager SCLB derivation in which every block is canceled, linked, isolated, simple, balanced and spanned.

PROOF. We can use an inductive transformation to streamline individual blocks to eliminate unneeded inferences; these streamlined blocks implicitly reflect the logic programming search strategy of focused search on particular goals and program statements. This transformation provides the basis for iteratively adapting the lowest $(\vee \rightarrow)$ inference L in the derivation to match either $(\vee \rightarrow^B_L)$ or $(\vee \rightarrow^B_R)$, while maintaining that a distinguished formula is linked in the lowest block. As before, we copy inferences into subderivations of L to reintroduce local work while keeping reasoning eager; this way we will not preserve any local work across the new block boundary. The extended subderivations are then transformed inductively to use SCLB inferences; the transformation also ensures that the side expressions of L are linked in the new blocks. We substitute only one of these new derivations, so that L satisfies either $(\vee \rightarrow^B_L)$ or $(\vee \rightarrow^B_R)$. With a suitable choice, we can retain the use of our own distinguished formula in the lowest block. L is now a blocking inference and no new $(\vee \rightarrow)$ inferences have been introduced, so we can proceed to transform the resulting derivation to SCLB. The argument is presented in its entirety in the electronic appendix to this article. □

3.4 Modularity

We now derive SCLP from SCLB. SCLP proofs can be rewritten to SCLB rules by a weakening transformation. Conversely, rewriting SCLB proofs to SCLP proofs is accomplished by induction on the structure of proofs. The transformation is possible because multiple formulas in sequents are needed only for passing ambiguities and work done across branches in the search; this is ruled out by the use of $(\vee \rightarrow^B_L)$, $(\vee \rightarrow^B_R)$ and $(\supset\rightarrow^S)$.

LEMMA 3.17. *Given a blockwise eager SCLB derivation $\mathcal{D}$, with end-sequent*

$$\Pi; \longrightarrow; \Theta$$

in which every block is linked, simple and spanned, we can construct a corresponding SCLP derivation of the same end-sequent in which every block remains linked.

PROOF. The argument proceeds directly by induction on the structure of proofs and is presented in its entirety in the electronic appendix to this article. □

4. PRACTICAL CONSEQUENCES

Modular goal-directed proof search is not just theoretically possible but practically feasible; a range of applications can now draw on it. In [Stone 1998b], I describe a preliminary implementation of proof search in SCLP as a logic programming interpreter DIALUP. I close by summarizing *how* (Section 4.1) and no less importantly *why* (Section 4.2) I developed this implementation.

4.1 Implementation

An effective implementation of SCLP requires further treatments of *unification* and *search control*.

In general, to implement first-order sequent calculus proof search, we must *lift* the inference figures. That is, we adapt the inferences that require instantiation to specific terms so that they introduce *logic variables* instead. As we construct derivations, we now accumulate *constraints* on the values of these variables—for example, we get constraints when an axiom link requires two formulas to be identical. Derivations in the lifted systems are thus the *skeletons* for proofs; each derivation represents a set of ground proofs, and these ground proofs are obtained by instantiating the variables to values that satisfy the constraints. Lifting is the essence of the resolution procedure [Robinson 1965] but can be regarded as a general metatheoretical strategy. [Lincoln and Shankar 1994; Voronkov 1996] offer particularly general discussions of this strategy at its most sophisticated.

For first-order modal inference in prefixed calculi, lifting introduces two kinds of logic variables, and two corresponding kinds of constraints. First-order quantifiers introduce logic variables over individuals, subject to the familiar constraints that give rise to term unification problems. Modal inferences, meanwhile, introduce logic variables over prefixes, subject to path equations. This leads to specialized problems of equational unification; good solutions are known for the general setting of multi-modal logic; see for example [Auffray and Enjalbert 1992; Debart et al. 1992; Otten and Kreitz 1996; Schmidt 1998].

The logical fragment of SCLP makes path equations particularly simple. Inspection of the SCLP proof rules shows that, at any point in proof search, we have enough path constraints to determine *ground* substitutions for all the path variables in the sequent except possibly for variables in the current program statement that are about to be unified with a goal. In many cases, this makes path equations easy to solve—a compact representation of all possible solutions can be computed in polynomial time. The details are beyond the scope of this paper, but see [Stone 1998b].

Search control is the other issue. An implementation has to make commitments about what statements to try and what rules to use to process those statements. The fact that SCLP program and goal statements are labeled with ground prefixes means that we can easily test that a statement's label is a prefix of the goal label before attempting to match the statement and the goal. We can also identify an atomic subformula of the statement nondeterministically as the *head*, and commit to match that head with the goal. Before doing so, we can for example test that the head and the goal share the same predicate symbol.

In the case of disjunction, we also want to make sure that we avoid reporting duplicate proofs, despite the duplicate rules for disjunction that we have. Loveland considers a number of heuristics for this [Loveland 1991], which should apply in SCLP as well as in Near-Horn Prolog. But here is another heuristic. As motivated in Section 1.1.3, $(\vee \rightarrow_R)$ is required only for cancellation. When we use it, we expect to cancel an assumption (like B in Figure 5) that could not be canceled otherwise. We can make this precise: $(\vee \rightarrow_R)$ should only be used in a restart block, and the assumption that is canceled in that block ought not to be used in the subsequent restart block initiated by the $(\vee \rightarrow_R)$ inference. Otherwise, we will independently construct an alternative proof that uses $(\vee \rightarrow_L)$ instead. Naturally, the kind of block analysis illustrated in the proof of Theorem 3.5 can be used to show that this restriction is complete.

4.2 Applications in modal representation

In classical logic, indefinite information is a bit exotic. Rather than developing an indefinite specification, we much prefer to collect the additional information required to describe the world in a precise, definite way. This is not true at all with modal specifications. Modal specifications get much of their interest from their ability to contrast different perspectives or sources of information. What one source of information represents with specific, definite information, another source represents with abstract, indefinite information. Computation from modal specifications involves the coordinated exchange of information between these sources.

In particular, problems of *planning* [Stone 1998a] and problems of *communication* [Stone 2000] depend on indefinite modal specifications. In planning, one agent, the *scheduler*, has to allocate a task to another agent, the *executive*. (The executive may just be the scheduler at a later point in time!) It is unrealistic to expect that the scheduler will know *exactly* what the executive *will* do; this almost certainly requires information that is not available to the scheduler. Rather, the scheduler should merely know what the executive *can* do. This means that, to be useful, the scheduler must have an *indefinite* modal specification that abstractly describes the information that will be available to the executive. For examples, see [Moore 1985; Morgenstern 1987; Scherl and Levesque 1993; Davis 1994] as well as [Stone 1998a].

In communication, the task of one agent, the *speaker*, is to formulate an utterance that allows another agent, the *hearer*, to answer a question. There are many cases where the speaker does not have enough information to answer the question directly. However, the speaker can still design an utterance that allows the hearer to infer the right answer, because the hearer knows something the speaker does not. Concretely, a user of a computer interface might want to know what action to take next. The right answer might be for the user to type *jdoe* into a certain text box. The speaker might know to say *enter your user ID*, even if the speaker does not know what the user ID is. Again, the speaker can make such choices meaningfully only from an indefinite modal specification that says what the hearer knows abstractly but not definitely. See [Stone 2000] for a worked-out formal case study.

5. CONCLUSION

To execute modal specifications requires leveraging both the flexibility of efficient classical theorem-proving and the distinctive modularity of modal logic. This is a significant problem because the two are at odds. On the one hand, flexible search strategies impose no constraints on the relationships among inferences. By ignoring modularity, they can leave open inappropriate possibilities for search. On the other hand, brute-force modular systems may place such strong constraints on the order in which search must proceed that it becomes impossible to guide that search in a predictable, goal-directed way.

This paper has explored one strategy for balancing the flexibility of classical goal-directed search with the modularity of modal logic. This strategy pursues a characterization of logic programming in terms of uniform proof, but embraces path-based representations of explicit scope, Herbrand terms to represent generic instances of quantifiers and modal operators, and a restart rule to characterize disjunction. Within this framework, proofs organize inferences not only to respect logic programming discipline but also to keep close tabs on local assumptions of disjunctive information. The strategy culminates with the development of a modular logic programming sequent calculus SCLP.

SCLP adopts a goal-directed proof regime in which formulas rather than proof problems are assigned scopes, and reasoning is not confined to within a single scope. Nevertheless, SCLP uses a prefix test to guarantee that only inferences that respect the modularity and locality of modal logic are considered. Crucially, SCLP adopts a modular restart regime for analyzing disjunctive programs. When reasoning from a local ambiguity in a specification, SCLP restarts a local goal. Because of this operational behavior, specifications can use modularity and locality to set up proof search problems for SCLP in which multiple logically independent ambiguities will not interact in proof search. Because it forces these ambiguities to be considered independently, SCLP can be used to construct efficiently executable specifications in reasoning tasks involving partial information that otherwise might require prohibitive search.

ELECTRONIC APPENDIX

The electronic appendix for this article can be accessed in the ACM Digital Library by visiting the following URL: `http://www.acm.org/pubs/citations/journals/tocl/2004-V-N/p1-Stone`.

ACKNOWLEDGMENTS

Thanks to Michael Fourman, L. Thorne McCarty, Mark Steedman, Rich Thomason, and three anonymous reviewers for extensive comments, and to Jean Gallier, Aravind Joshi, Dale Miller and Scott Weinstein, who also encouraged this work in its early stages. This work has been supported by an NSF graduate fellowship, an IRCS graduate fellowship, and a postdoctoral fellowship from RUCCS, as well as NSF grants CISE CDA 9818322, IRI95-04372, ARPA grant N6601-94-C6043, and ARO grant DAAH05-94-G0426.

REFERENCES

ANDREOLI, J.-M. 1992. Logic programming with focusing proofs in linear logic. *J. Logic and Comput. 2,* 3, 297–347.

AUFFRAY, Y. AND ENJALBERT, P. 1992. Modal theorem proving: an equational viewpoint. *J. Logic and Comput. 2,* 3, 247–295.

BALDONI, M. 2000. Normal multimodal logics with interaction axioms. In *Labelled Deduction*, D. Basin, M. D'Agostino, D. M. Gabbay, S. Matthews, and L. Viganò, Eds. Kluwer Academic Publishers, Dordrecht, NL, 33–57.

BALDONI, M., GIORDANO, L., AND MARTELLI, A. 1993. A multimodal logic to define modules in logic programming. In *Logic Programming: Proceedings of the 1993 International Symposium*. MIT Press, Cambridge, MA, 473–487.

BALDONI, M., GIORDANO, L., AND MARTELLI, A. 1996. A framework for modal logic programming. In *Logic Programming: Proceedings of the 1996 Joint International Conference and Symposium*, M. Maher, Ed. MIT Press, Cambridge, MA, 52–66.

BALDONI, M., GIORDANO, L., AND MARTELLI, A. 1998. A modal extension of logic programming: Modularity, beliefs and hypothetical reasoning. *J. Logic and Comput. 8,* 5, 597–635.

BASIN, D., MATTHEWS, S., AND VIGANÒ, L. 1998. Labelled modal logics: Quantifiers. *J. Logic Lang. and Inf. 7,* 3, 237–263.

BECKERT, B. AND GORÉ, R. 1997. Free variable tableaux for propositional modal logics. In *Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX '97*. Lecture Notes in Artificial Intelligence, vol. 1227. Springer Verlag, Heidelberg, DE, 91–106.

CATACH, L. 1991. TABLEAUX, a general theorem prover for modal logics. *J. of Automated Reasoning 7,* 489–510.

DAVIS, E. 1994. Knowledge preconditions for plans. *J. Logic and Comput. 4,* 5, 721–766.

DEBART, F., ENJALBERT, P., AND LESCOT, M. 1992. Multimodal logic programming using equational and order-sorted logic. *Theor. Comput. Sci. 105*, 141–166.

FARIÑAS DEL CERRO, L. 1986. MOLOG: A system that extends PROLOG with modal logic. *New Gen. Comput. 4*, 35–50.

FITTING, M. 1972. Tableau methods of proof for modal logics. *Notre Dame J. of Formal Logic 13,* 2, 237–247.

FITTING, M. 1983. *Proof Methods for Modal and Intuitionistic Logics*. Synthese Library, vol. 169. D. Reidel, Dordrecht.

FITTING, M. AND MENDELSOHN, R. L. 1998. *First-order Modal Logic*. Synthese Library, vol. 277. Kluwer Academic Publishers, Dordrecht, NL.

FRISCH, A. M. AND SCHERL, R. B. 1991. A general framework for modal deduction. In *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*. Morgan Kaufmann, Los Altos, CA, 196–207.

GABBAY, D. M. 1985. N-Prolog: an extension of Prolog with hypothetical implications, Part 2. *J. Logic Program. 5*, 251–283.

GABBAY, D. M. 1987. Modal and temporal logic programming. In *Temporal Logics and their Applications*, A. Galton, Ed. Academic Press, New York, NY, 197–237.

GABBAY, D. M. 1992. Elements of algorithmic proof. In *Handbook of Logic in Computer Science (volume 2): Background: Computational Structures*, S. Abramsky, D. M. Gabbay, and T. S. E. Mailbaum, Eds. Oxford University Press, Oxford, 311–413.

GABBAY, D. M. 1996. *Labelled Deductive Systems*. Oxford University Press, Oxford.

GABBAY, D. M. AND OLIVETTI, N. 1998. Algorithmic proof methods and cut elimintation for implicational logics: Part I: Modal implication. *Stud. Logic. 61*, 237–280.

GABBAY, D. M. AND REYLE, U. 1984. N-Prolog: an extension of Prolog with hypothetical implications, Part 1. *J. Logic Program. 4*, 319–355.

GIORDANO, L. AND MARTELLI, A. 1994. Structuring logic programs: A modal approach. *J. Logic Program. 21*, 59–94.

GIRARD, J.-Y. 1993. On the unity of logic. *Ann. of Pure and Appl. Logic 59*, 201–217.

GORÉ, R. 1992. Cut-free sequent and tableau systems for propositional normal modal logics. Ph.D. thesis, University of Cambridge.

GORÉ, R. 1999. Tableau methods for modal and temporal logics. In *Handbook of Tableau Methods*, M. D'Agostino, D. Gabbay, R. Hähnle, and J. Posegga, Eds. Kluwer Academic Publishers, Dordrecht, NL, 297–396.

HARLAND, J. 1994. A proof-theoretic analysis of goal-directed provability. *J. Logic and Comput. 4,* 1, 69–88.

HARLAND, J. 1997. On goal-directed provability in classical logic. *Computer Languages 23,* 161–178.

HARLAND, J., LUTOVAC, T., AND WINIKOFF, M. 2000. Goal-directed proof search in multiple-conclusioned intuitionistc logic. In *Computational Logic—CL 2000, First International Conference*. Lecture Notes in Artificial Intelligence, vol. 1861. Springer Verlag, Heidelberg, DE, 254–268.

HINTIKKA, J. 1971. Semantics for propositional attitudes. In *Reference and Modality*, Linsky, Ed. Oxford University Press, Oxford, 145–167.

HODAS, J. S. AND MILLER, D. 1994. Logic programming in a fragment of intuitionistic linear logic. *Information and Computation 110,* 2, 327–365.

HOWARD, W. A. 1980. The formulae-as-types notion of construction. In *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus, and Formalism*, J. P. Seldin and J. R. Hindley, Eds. Academic Press, New York, NY, 479–490.

JACKSON, P. AND REICHGELT, H. 1987. A general proof method for first-order modal logic. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, Los Altos, CA, 942–944.

KLEENE, S. C. 1951. Permutation of inferences in Gentzen's calculi LK and LJ. In *Two papers on the predicate calculus*. American Mathematical Society, Providence, RI, 1–26.

KOBAYASHI, N., SHIMIZU, T., AND YONEZAWA, A. 1999. Distributed concurrent linear logic programming. *Theor. Comput. Sci. 227*, 185–220.

KRIPKE, S. A. 1963. Semantical analysis of modal logic. I. Normal modal propositional calculi. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik 9*, 67–96.

LINCOLN, P. D. AND SHANKAR, N. 1994. Proof search in first-order linear logic and other cut-free sequent calculi. In *Proceedings, 9th Annual IEEE Symposium on Logic in Computer Science*. IEEE Computer Society Press, Los Alamitos, CA, 282–291.

LOBO, J., MINKER, J., AND RAJASEKAR, A. 1992. *Foundations of Disjunctive Logic Programming*. MIT Press, Cambridge, MA.

LOVELAND, D. W. 1991. Near-horn Prolog and beyond. *J. of Automated Reasoning 7,* 1–26.

MASSACCI, F. 1998a. Single step tableaux for modal logics. Tech. Rep. DIS TR-04-98, Unviersity of Rome "La Sapienza".

MASSACCI, F. 1998b. Single step tableaux for modal logics: methodology, computations, algorithms. Tech. Rep. TR-04, DIS, University of Rome "La Sapienza".

MCCARTHY, J. 1993. Notes on formalizing context. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, Los Altos, CA, 555–560.

MCCARTHY, J. 1997. Modality, si! modal logic, no! *Stud. Logic. 59,* 1, 29–32.

MCCARTHY, J. AND BUVAČ, S. 1994. Formalizing context (expanded notes). Tech. Rep. STAN-CS-TN-94-13, Stanford University.

MILLER, D. 1989. A logical analysis of modules in logic programming. *J. Logic Program. 6,* 1–2, 79–108.

MILLER, D. 1994. A multiple-conclusion meta-logic. In *Proceedings, 9th Annual IEEE Symposium on Logic in Computer Science*, S. Abramsky, Ed. IEEE Computer Society Press, Los Alamitos, CA, 272–281.

MILLER, D. 1996. Forum: A multiple-conclusion specification logic. *Theor. Comput. Sci. 165*, 201–232.

MILLER, D., NADATHUR, G., PFENNING, F., AND SCEDROV, A. 1991. Uniform proofs as a foundation for logic programming. *Ann. of Pure and Appl. Logic 51,* 125–157.

MOORE, R. C. 1985. A formal theory of knowledge and action. In *Formal Theories of the Commonsense World*, J. R. Hobbs and R. C. Moore, Eds. Ablex, Norwood NJ, 319–358.

MORGENSTERN, L. 1987. Knowledge preconditions for actions and plans. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, Los Altos, CA, 867–874.

NADATHUR, G. 1998. Uniform provability in classical logic. *J. Logic and Comput. 8,* 2, 209–229.

NADATHUR, G. AND LOVELAND, D. W. 1995. Uniform proofs and disjunctive logic programming. In *Proceedings, 10th Annual IEEE Symposium on Logic in Computer Science*. IEEE Computer Society Press, Los Alamitos, CA, 148–155.

NONNENGART, A. 1993. First-order modal logic theorem proving and functional simulation. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, Los Altos, CA, 80–87.

OHLBACH, H. J. 1991. Semantics-based translation methods for modal logics. *J. Logic and Comput. 1,* 5, 691–746.

OHLBACH, H. J. 1993. Optimized translation of multi modal logic into predicate logic. In *Logic Programming and Automated Reasoning*, A. Voronkov, Ed. LNCS, vol. 698. Springer, Berlin, 253–264.

ORGUN, M. A. AND WADGE, W. W. 1992. Towards a unified theory of intensional logic programming. *J. Logic Program. 13,* 4, 413–440.

OTTEN, J. AND KREITZ, C. 1996. T-string-unification: unifying prefixes in non-classical proof methods. In *Theorem Proving with Analytic Tableaux and Related Methods, 5th International Workshop, TABLEAUX '96*. Lecture Notes in Artificial Intelligence, vol. 1071. Springer Verlag, Heidelberg, DE, 244–260.

PRIOR, A. N. 1967. *Past, Present and Future.* Clarendon Press, Oxford.

PYM, D. AND HARLAND, J. 1994. A uniform proof-theoretic investigation of linear logic programming. *J. Logic and Comput. 4*, 175–207.

ROBINSON, J. A. 1965. A machine oriented logic based on the resolution principle. *J. ACM 12,* 1, 23–45.

SAKAKIBARA, Y. 1987. Programming in modal logic: An extension of PROLOG based on modal logic. In *Logic Programming '86, Proceedings of the 5th Conference,* E. Wada, Ed. Number 264 in LNCS. Springer Verlag, Heidelberg, DE, 81–91.

SCHERL, R. B. AND LEVESQUE, H. J. 1993. The frame problem and knowledge-producing actions. In *Proceedings of the 11th National Conference on Artificial Intelligence*. AAAI Press, Menlo Park, CA, 689–695.

SCHILD, K. 1991. A correspondence theory for terminological logics: preliminary report. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, Los Altos, CA, 466–471.

SCHMIDT, R. A. 1998. E-Unification for subsystems of S4. In *Rewriting Techniques and Applications, 9th International Conference, RTA-98*. Lecture Notes in Computer Science, vol. 1379. Springer Verlag, Heidelberg, DE, 106–120.

STONE, M. 1998a. Abductive planning with sensing. In *Proceedings of the 15th National Conference on Artificial Intelligence*. AAAI Press, Menlo Park, CA, 631–636.

STONE, M. 1998b. Modality in dialogue: Planning, pragmatics and computation. Ph.D. thesis, University of Pennsylvania.

STONE, M. 1999. Representing scope in intuitionistic deductions. *Theor. Comput. Sci. 211,* 1–2, 129–188.

STONE, M. 2000. Towards a computational account of knowledge, action and inference in instructions. *J. Lang. and Comput. 1*, 231–246.

VORONKOV, A. 1996. Proof-search in intuitionistic logic based on constraint satisfaction. In *Theorem Proving with Analytic Tableaux and Related Methods, 5th International Workshop, TABLEAUX '96*. Lecture Notes in Artificial Intelligence, vol. 1071. Springer Verlag, Heidelberg, DE, 312–329.

WALLEN, L. A. 1990. *Automated Proof Search in Non-Classical Logics: Efficient Matrix Proof Methods for Modal and Intuitionistic Logics*. MIT Press, Cambridge, MA.

Received July 2001; revised June 2002 and February 2004 accepted February 2004