Compiled Only Knowing

Bjorn Bjurling, Krysia Broda

Department of Computing

Imperial College of Science Technology and Medicine
180 Queen's Gate, London SW7 2BZ
email: {bgb,kb}@doc.ic.ac.uk

Departmental Technical Report 2002/9

Abstract

We report on a sound and complete proof system, COOL, for the propositional fragment of Hector Levesque's nonmonotonic logic 'The Logic of Only Knowing' [Lev90]. The proof system is devised using the framework of compiled labelled deductive systems [BrGaRu00], which enables a translation of COOL-theories into theories of first order logic. With this first order translation, we are able to perform OL-derivations in standard first order theorem provers.

The main events in the report are the soundness and completeness theorems for COOL.

Contents

1 The Logic of Only Knowing **2**
1.1 The language of OL 2
1.2 Semantics 2
1.3 OL Axiomatic Proof System 4

2 COOL **5**
2.1 Preliminary Definitions 5
2.1.1 The consequence relation $\vdash_{\text{COOL}}$ 7
2.1.2 Completing the definition of COOL 11
2.2 Cool-Semantic 12

3 Properties of configurations **12**

4 Soundness and Completeness **13**
4.1 Soundness 13
4.2 Completeness 17

5 Future Work **20**

A A Sample Proof in COOL **21**

B A Sample Script from Otter **21**

1 The Logic of Only Knowing

In 1990 Hector Levesque [Lev90] presented an approach to Autoepistemic logic called 'the logic of only knowing', OL. OL is a first order language but in this paper we only consider the propositional fragment of OL. We call that fragment OL as well.

1.1 The language of OL

The alphabet consists of a countable set of propositional letters (which we call p or q possibly with subscripts); two propositional connectives ($\neg$ and $\wedge$); and three unary modal operators, O, B, and N. The OL-atoms are the propositional letters and the OL-formulas are built in the normal way and we use standard abbreviations as we like.

A formula without any modal operators is called *objective*. A formula that has all of its objective subformulas within the scope of a modal operator is called *subjective*. A subjective formula without the operators O and N is called *basic*. A formula that is neither subjective, basic nor objective hasn't got any special name but is still potentially quite interesting.

The operators B and N stand for belief in two different ways. The formula $\mathsf{B}\alpha$ says that 'at least α is believed' in the sense that α is true in all the worlds that are accessible, and possibly some inaccessible worlds as well. The formula $\mathsf{N}\alpha$ says that 'α is at most believed' in the sense that α is true in all inaccessible worlds and possibly in some accessible as well. The formula $\mathsf{O}\alpha$ says that α is true in exactly the accessible worlds.

The operators B and N work as weak *S5* (*K45*) operators, hence belief is closed under logical consequence, and anything believed is believed to be believed, and anything not believed is believed not to be believed. The operator O is not normal in the sense of ordinary monotonic modal logic as defined in [Che80].

1.2 Semantics

From now on, let L_A stand for the set of atoms in the language OL.

We define an *assignment* to be a subset of L_A. Levesque defines assignments as functions from L_A to $\{0, 1\}$, but that is equivalent to the characteristic functions of our subsets.

The structures of OL are pairs (W, w), where w is an assignment and W is a set of assignments. We now proceed to define the satisfaction relation.

Levesque defines satisfaction of formulas with respect only to *maximal* structures (W, w). Both maximality and satisfiability are defined in terms of the following weaker relation.

Definition 1.1. Let (W, w) be a pair such that w is an assignment and W is a set of assignments. Let α and β be sentences of OL. Then the (weak satisfaction) relation $\Vdash$ between such pairs and sentences of OL is defined as follows.

(i) for any atomic p, $(W, w) \Vdash p$ iff $p \in w$

(ii) $(W, w) \Vdash \neg\alpha$ iff $(W, w) \nVdash \alpha$

(iii) $(W, w) \Vdash \alpha \wedge \beta$ iff $(W, w) \Vdash \alpha$ and $(W, w) \Vdash \beta$

(iv) $(W, w) \Vdash \mathsf{B}\alpha$ iff $\forall x(x \in W \rightarrow (W, x) \Vdash \alpha)$

(v) $(W, w) \Vdash \mathsf{N}\alpha$ iff $\forall x(x \notin W \rightarrow (W, x) \Vdash \alpha)$

(vi) $(W, w) \Vdash \mathsf{O}\alpha$ iff $\forall x(x \in W \leftrightarrow (W, x) \Vdash \alpha)$ □

Definition 1.2. Two sets W_1 and W_2 of assignments are said to be *equivalent* iff for every basic α it holds that $W_1 \Vdash \mathsf{B}\alpha$ iff $W_2 \Vdash \mathsf{B}\alpha$.

Theorem 1.3. *For any set W of assignments there exists a unique largest superset W^+ of W such that W^+ is equivalent to W.*

Proof. See [Lev90]. □

Definition 1.4. Let W be a set of assignments. The unique superset W^+ of W, such that W^+ and W are equivalent is said to be *maximal.* A structure (W, w) where W is maximal is also said to be maximal.

It follows from the theorem that for any set A of basic sentences, any w, and any W, that

$$(W, w) \Vdash A \text{ iff } (W^+, w) \Vdash A$$

Hence, it follows from the theorem that it suffices to consider maximal sets when investigating the satisfiability and validity of any OL-sentence.

Definition 1.5.

(i) a sentence α is *satisfiable* iff there is a maximal set W and some w such that $(W, w) \Vdash \alpha$

(ii) a set A of sentences is *satisfiable* iff each element of A is satisfiable

(iii) a set A *entails* a sentence α (written $A \models \alpha$) iff $A \cup \{\neg\alpha\}$ is not satisfiable

(iv) α is *valid* iff $\emptyset \models \alpha$.

If α is satisfiable by a structure (W, w), we say that (W, w) is a *model* for α and we write $(W, w) \models \alpha$. □

In the definition, the truth of objective sentences does not depend on W, and the truth of subjective sentences does not depend on w. Hence, when convenient and appropriate, we can write just $w \models \alpha$ or $W \models \alpha$ as it suits us. (Also, we treat the structure (W, x) as an unordered pair.)

1.3 OL Axiomatic Proof System

Levesque gives an axiomatic proof system for OL. We just restate it here. It includes:

1. all propositional tautologies are theorems of OL.
2. weak S5 axioms for the operators B and N
   (i) $\mathsf{B}(\alpha \to \beta) \to (\mathsf{B}\alpha \to \mathsf{B}\beta)$, $\mathsf{N}(\alpha \to \beta) \to (\mathsf{N}\alpha \to \mathsf{N}\beta)$
   (ii) $\mathsf{B}\alpha \to \mathsf{BB}\alpha$, $\mathsf{N}\alpha \to \mathsf{NN}\alpha$
   (iii) $\neg\mathsf{B}\alpha \to \mathsf{B}\neg\mathsf{B}\alpha$, $\neg\mathsf{N}\alpha \to \mathsf{N}\neg\mathsf{N}\alpha$
3. cross axioms
   (i) let σ be a subjective sentence, then $\sigma \to \mathsf{B}\sigma$, and $\sigma \to \mathsf{N}\sigma$
   (ii) let ϕ be a falsifiable objective sentence, then $\mathsf{N}\phi \to \neg\mathsf{B}\phi$
4. the definition of O. $\mathsf{O}\alpha \leftrightarrow (\mathsf{B}\alpha \wedge \mathsf{N}\neg\alpha)$
5. and has two inference rules
   (i) if α is a valid objective sentence, then $\mathsf{B}\alpha$ and $\mathsf{N}\alpha$ are valid subjective sentences.
   (ii) if $\alpha \to \beta$ and α are valid, then so is β □

The formulas in the above definition are of course only schemata. Observe that the first cross axiom subsumes the last two weak S5 axioms.

2 COOL

We shall employ the framework of Compiled Labelled Deductive Systems CLDS [Ru96, BrGaRu00] and devise a sound and complete proof system for the propositional fragment of the logic of only knowing. There are two reasons for employing such a framework. First, any CLDS translates a logic to first order logic and so makes is possible to use standard first order theorem provers on the logic of only knowing. Second, CLDS provides a general setting for analysing the logic of only knowing, in which possibly other nonmonotonic formalisms are analysable as well. A CLDS is a labelled deductive system with some added features, as described bellow.

A labelled deductive system [Gab96] is given by a triple $((L, L_L), \mathcal{A}, \rho)$, where L and L_L are any two languages (in the LDS literature, L_L is called *the labelling language*, often taken to be a first order language), $\mathcal{A}$ is a first order theory written in the language L_L (in the literature called *the labelling algebra*), and ρ is a relation defined on the set of configurations. (configurations to be defined shortly).

Basically and simplistically, to form a *compiled* labelled deductive system from some LDS, we do the following:

1. choose a third language L_C;
2. define a translation from the language (L, L_L) to L_C;
3. maybe add a couple of elements to the theory $\mathcal{A}$.

2.1 Preliminary Definitions

In this subsection we define the compiled labelled deductive system COOL, Compiled OL, $((L, L_L), \mathcal{A}^+, \mathcal{R})$.

The language L is the propositional fragment of OL extended with the propositional constant $\perp_{\text{COOL}}$. We will later on have a first order $\perp_{\text{FOL}}$ as well. Often, we shall omit the subscripts on $\perp$ and let the context indicate which one we are talking about.

Definition 2.1. The labelling language L_L is a two-sorted (sorts S_1 and S_2) first order language with the following signature.

(i) a set N of constant symbols of the first sort, denoted by natural numbers

(ii) a set M of constant symbols of the second sort, denoted by m with subscripts

(iii) two binary relation symbols R and $\bar{R}$ of sort $S_1 \times S_2$.

(iv) for each formula α of OL, a unary function symbol f_α, and a unary function symbol g_α. Both are of sort $S_2 \times S_1$, that is, their arguments are of sort S_2 and their values are of sort S_1. □

Variables of sort S_1 will be denoted by x with subscripts, and variables of sort S_2 will be denoted by y with subscripts. The set of variables of sort S_1 is denoted by X and the set of variables of sort S_2 is denoted by Y.

The system we are defining is more general than we need in this paper. In effect, we will never use more than one constant m from M. This constant is denoted by y throughout the paper. In this sense, all references to that constant y may seem redundant. Indeed, y is redundant throughout, but we keep it because we have a more general approach in the back of our minds.

Definition 2.2. The terms of the language L_L are pairs of elements (η, ζ) where $\eta \in X \cup N$ and $\zeta \in Y \cup M$. The terms are called labels. □

Definition 2.3. A *declarative unit* (with respect to an LDS $((L, L_L), \mathcal{A}, \rho)$) is a pair (α, λ) where α is a formula in the language L and λ is a term in the labelling language L_L. We write the declarative unit (α, λ) as $\alpha : \lambda$. □

In the construction of the labelling language and labelling theory $\mathcal{A}$, we aim to reflect the semantics of OL. In the labelling language we do it by defining the labels as pairs, informally corresponding to the OL-structures that are pairs (w, W) of an assignment and a set of assignments. Note that, as with our treatment of y in COOL, the W in (w, W) is also held constant throughout derivations in OL.

The labelling theory, $\mathcal{A}$, is consequently defined to reflect an important property of the the OL-semantics. The labelling theory informally reflects the property that if an OL-structure (w, W) is a model for a formula α, then for any assignment u, either $u \in W$ or u is in the complement of W, which equivalently is to say that $W \cap \bar{W} = \emptyset$ and $W \cup \bar{W} = \mathcal{U}$, where $\mathcal{U}$ is the universe of assignments. Incidently, our labelling theory is very simple:

Definition 2.4. The *labelling theory* $\mathcal{A}$ is the first order theory, written in the language L_L, defined as

$$\mathcal{A} = \{\forall x \forall y(\neg Rxy \leftrightarrow \bar{R}xy)\}$$

.

Definition 2.5. An *R-atom* is a formula of the form Rxy or $\bar{R}xy$ where $x \in X \cup N$ and $y \in Y \cup M$. An *R-literal* is an R-atom or a negated R-atom. An *R-formula* is an R-literal or an existentially or universally quantified R-literal, $\pm\exists x \pm Rxy$, or $\pm\exists x \pm \bar{R}xy$.

Definition 2.6. A *diagram* is an at most countable set of R-formulas.

Definition 2.7. A *configuration* is the union of an at most countable set of declarative units and a diagram. The set of configurations is denoted by $\mathcal{C}$. The *union* of two configurations is the union of the two diagrams and the two sets of declarative units. The difference of two configurations, $c - c'$ is the ordinary set difference $c \setminus c'$.

Proposition 2.8. *Let c and c' be configurations. Let $\mathcal{K}$ be a collection of configurations. Then $c - c'$ and $\bigcup \mathcal{K}$ are also configurations.*

Proof. Easy consequence of elementary set theory. □

Definition 2.9. Let c be a configuration. The set of labels of c, denoted by $t(c)$ is defined as

$$t(c) = \{(\eta, \zeta) \mid \alpha : (\eta, \zeta) \in c \text{ or } R\eta\zeta \in c \text{ or } \bar{R}\eta\zeta \in c\}$$

2.1.1 The consequence relation $\vdash_{\text{COOL}}$

The consequence relation $\vdash_{\text{COOL}}$ is defined in terms of a relation $\mathcal{R}$, which will be defined next.

$\mathcal{R}$, is defined as the union of four other relations. We call those four relations (*i*) the Propositional relation, (*ii*) the OL relation, (*iii*) the First order relation, and (*iv*) the Splitting relation. We treat those four relations separately below, and then we define $\mathcal{R}$, and eventually $\vdash_{\text{COOL}}$.

Convention and Notation When we want to talk about the items in a configuration, we just call them *formulas*. When we want to explicitly say that a configuration c contains formulas $\phi_0, ...\phi_n$, we write $c(\phi_0, ..., \phi_n)$.

The Propositional Relation Let c be any configuration and c' a configuration as specified. Let χ by any declarative unit or any R-formula. Then $(c, c') \in \mathcal{R}_P$ if one of the following conditions hold

(i) $c((\alpha \wedge \beta):(x,y))$ and $c' = c \cup \{\alpha:(x,y), \beta:(x,y)\}$

(ii) $c(\neg\neg\alpha:(x,y))$ and $c' = c \cup \{\alpha:(x,y)\}$

(iii) $c(\perp_{\text{COOL}})$, for any c'

(iv) $c(\chi, \neg\chi)$ and $c' = c \cup \{(\perp_{\text{COOL}})\}$. □

The OL Relation Let c be any configuration and c' a configuration as specified. Let n be a constant of the first sort. Then $(c, c') \in \mathcal{R}_O$ if one of the following conditions hold

(i) $c(\mathsf{B}\alpha:(x,y), Rny)$, and $c' = c \cup \{\alpha:(n,y)\}$

(ii) $c(\mathsf{N}\alpha:(x,y), \bar{R}ny)$, and $c' = c \cup \{\alpha:(n,y)\}$

(iii) $c(\mathsf{O}\alpha:(x,y))$, and $c' = c \cup \{\mathsf{B}\alpha:(x,y),\ \mathsf{N}\neg\alpha:(x,y)\}$

(iv) $c(\neg\mathsf{B}\alpha:(x,y))$, and $c' = c \cup \{(Rf_\alpha(y)y,\ \neg\alpha:(f_\alpha(y),y)\}$

(v) $c(\neg\mathsf{N}\alpha:(x,y))$, and $c' = c \cup \{\bar{R}g_\alpha(y)y,\ \neg\alpha:(g_\alpha(y),y)\}$

The First Order Relation Let c be any configuration and c' a configuration as specified. Let n be a constant of the first sort. Let ϕ be either $\pm R$ or $\pm\bar{R}$. Then $(c, c') \in \mathcal{R}_F$ if

(i) $c(\neg Rny)$ and $c' = c \cup \{\bar{R}ny\}$

(ii) $c(Rny)$ and $c' = c \cup \{\neg\bar{R}ny\}$

(iii) $c(\neg\bar{R}ny)$ and $c' = c \cup \{Rny\}$

(iv) $c(\bar{R}ny)$ and $c' = c \cup \{\neg Rny\}$

(v) $c(\exists x\phi(x,y))$ and $c' = c \cup \{\phi(n,y)\}$ for some constant n such that $(n,y) \notin t(c)$.

(vi) $c(\forall x\phi(x,y), \alpha:(n,y))$ and $c' = c \cup \{\phi(n,y)\}$.

The Splitting Relation Let c be any configuration and c' a configuration as specified. Let n be a constant of the first sort. Then $(c, c') \in \mathcal{R}_S$ if one of the following conditions hold

(i) if $c(\neg(\alpha \wedge \beta) : (x, y))$ and $c' = c \cup \{\neg\alpha : (x, y)\}$ or $c' = c \cup \{\neg\beta : (x, y)\}$

(ii) if $c(\neg\mathsf{O}\alpha : (x, y))$ and $c' = c \cup \{\neg\mathsf{B}\alpha : (x, y)\}$ or $c' = c \cup \{\neg\mathsf{N}\neg\alpha : (x, y)\}$

(iii) if $c(\alpha : (x, y))$ for some $\alpha : (x, y)$ and $c' = c \cup \{Rxy\}$ or $c' = c \cup \{\bar{R}xy\}$

(iv) if $c' = c \cup \{\exists xRxy\}$ or $c' = c \cup \{\forall x\bar{R}xy\}$

(v) if $c' = c \cup \{\exists x\bar{R}xy\}$ or $c' = c \cup \{\forall xRxy\}$

Definition 2.10. $\mathcal{R}$ is the subset of $\mathcal{C} \times \mathcal{C}$ such that $(c, c') \in \mathcal{R}$ iff

$$(c, c') \text{ is an element in } \mathcal{R}_P, \mathcal{R}_O, \mathcal{R}_F \text{ or } \mathcal{R}_S$$

Definition 2.11. A finite $\mathcal{R}$-*path* from a configuration c to a configuration c' is a sequence $(c_0, ..., c_n)$ such that $c_0 = c$, for all $i < n$ $\mathcal{R}c_ic_{i+1}$, and $c_n = c'$. An infinite $\mathcal{R}$-*path* is an infinite sequence $\{c_i\}_{i \geq 0}$ such that $\mathcal{R}c_{i-1}c_i$ for each $i > 0$. We will say that c' is *reachable* from c if there is a finite $\mathcal{R}$-path from c to c'.

Definition 2.12. Let $p(c)$ be a path from a configuration c, and let $\cup p(c)$ be the configuration that is the union of the configurations in $p(c)$.

(i) If $p(c)$ contains a configuration that contains $\bot_{\text{COOL}}$, or simply if $\bot_{\text{COOL}} \in \cup p(c)$, then $p(c)$ is said to be *closed.*

(ii) If $p(c)$ is not closed, it is said to be *open.*

(iii) If $p(c)$ is open and if

$$\forall c'(c' \in \mathcal{C} \to (\mathcal{R}(\cup p(c))c' \to c' \subseteq (\cup p(c)))),$$

then $p(c)$ is said to be *saturated.*

Saturation means informally that no $\mathcal{R}$-rule that is applicable to a configuration c results in a formula that is not already in c, as stated in definition 2.12.

The next definition provides our view on proof trees. A proof tree is a set of branches. Since the relation $\mathcal{R}$ cumulates the configurations in a path, the union of a finite path is the same as its last element. Also, as noted earlier, the union of a path (a collection of configurations) is itself a configuration. We will use this dual view on branches in our proof trees: when it is convenient we see them as sequences of configurations, and otherwise we see them as the unions of such sequences. We define proof tree though to be a set of configurations.

Definition 2.13. Let c be a configuration and let $p(c)$ denote a path from c. Then we define the set $P(c)$ as

$$P(c) = \{\cup p(c) \mid p(c) \text{ is a path from } c\}$$

.

The elements of $P(c)$ will sometimes be called branches. If all elements of $P(c)$ are closed, $P(c)$ is said to be closed. Otherwise it is said to be open.

Definition 2.14. Let c be a configuration and χ a declarative unit or an R-formula. We say that there is a *COOL-proof* of χ from c if and only if all elements in $P(c \cup \{\neg\chi\})$ contain $\bot_{\text{COOL}}$. When there is a COOL-proof of χ from c we write $c \vdash_{\text{COOL}} \chi$.

Lemma 2.15. *Let c be a configuration. There is a systematic and fair procedure for constructing the proof tree $P(c)$ such that every branch in $P(c)$ is either closed or saturated.*

We omit the proof of the lemma and describe the procedure informally.

Given a configuration c, we define an order $\mathcal{O}$ on the elements in c. The proof tree $P(c)$ is systematically built in stages, by keeping track of the set Λ of current configurations (the leaves of the tree, i.e., the branches), so that in the i-th stage the i-th formula in the order $\mathcal{O}$ is processed and where each branch of $P(c)$ (element of Λ) is processed in each stage. The procedure starts with processing the first formula χ_0 in the order by computing its consequences by applying the rules in $\mathcal{R}$ on χ_0. In each subsequent stage, stage i say, the i-th formula χ_i of the order is added to each branch b and the consequences of χ_i and of all the formulas in b are computed. The consequences that are not already in b are then added to the branch (with splits where appropriate) to form the next configuration. The number of branches is always finite and the set of formulas to process in each stage stays finite because each formula has only finitely many consequences, so each stage always terminates.

The procedure is fair because the set of formulas to process in each stage is finite, so each formula in c will eventually be processed.

The proof of the lemma rests upon the facts that (*i*) each formula in the configuration c has only finitely many consequences, (*ii*) any proof tree contains finitely many branches, (*iii*) branches in finite configurations gets closed or saturated in finitely many steps, (*iv*) we can apply rules to the formulas in c in a specific order and the elements of the set (which is finite as already said) of consequences to the application of the rules to any formula in c can be inserted in the beginning of that order, and (*v*) we don't insert an inferred formula into the order if it is already there.

In the following we assume that all proof trees $P(c)$ for some configuration c are constructed according to the systematic and fair procedure.

2.1.2 Completing the definition of COOL

We have now defined the labelled deductive system. It remains to turn it into a *compiled* labelled deductive system.

Definition 2.16. The *extended labelling signature* τ^+ is obtained by adding, for each OL-formula α, a two-place relation symbol $[\alpha]^*$ of the sort $S_1 \times S_2$ to the signature of $\mathcal{A}$.

Definition 2.17. The *extended labelling theory* $\mathcal{A}^+$ is a τ^+-theory that includes $\mathcal{A}$ and an infinite number of axioms given by the following schemata.

1. $\forall x \forall y(\neg[\alpha]^*(x, y) \vee \neg[\neg\alpha]^*(x, y))$
2. $\forall x \forall y([\neg\neg\alpha]^*(x, y) \to [\alpha]^*(x, y))$
3. $\forall x \forall y([\alpha \wedge \beta]^*(x, y) \to [\alpha]^*(x, y) \wedge [\beta]^*(x, y))$
4. $\forall x \forall y([\neg(\alpha \wedge \beta)]^*(x, y) \to [\neg\alpha]^*(x, y) \vee [\neg\beta]^*(x, y))$
5. $\forall x \forall y([\mathsf{B}\alpha]^*(x, y) \to \forall z(Rzy \to [\alpha]^*(z, y)))$
6. $\forall x \forall y([\neg\mathsf{B}\alpha]^*(x, y) \to Rf_\alpha(y)y \wedge [\neg\alpha]^*(f_\alpha(y), y))$
7. $\forall x \forall y([\mathsf{N}\alpha]^*(x, y) \to \forall z(\bar{R}zy \to [\alpha]^*(z, y)))$
8. $\forall x \forall y([\neg\mathsf{N}\alpha]^*(x, y) \to \bar{R}g_\alpha(y)y \wedge [\neg\alpha]^*(g_\alpha(y), y))$
9. $\forall x \forall y([\mathsf{O}\alpha]^*(x, y) \to [\mathsf{B}\alpha]^*(x, y) \wedge [\mathsf{N}\neg\alpha]^*(x, y))$
10. $\forall x \forall y([\neg\mathsf{O}\alpha]^*(x, y) \to [\neg\mathsf{B}\alpha]^*(x, y) \vee [\neg\mathsf{N}\neg\alpha]^*(x, y))$

Theorem 2.18. *Let $\alpha:(x,y)$ be a declarative unit, and A a τ^+-structure such that $A \models_{FOL} \mathcal{A}^+$. Then*

(i) $A \models_{FOL} \forall x \forall y([\alpha]^*(x,y) \to \neg[\neg\alpha]^*(x,y))$

(ii) $A \models_{FOL} \forall x \forall y([\neg\alpha]^*(x,y) \to \neg[\alpha]^*(x,y))$

Proof. Follows immediately from the first item in the definition of $\mathcal{A}^+$. □

Definition 2.19. A first order theory T with the same signature as $\mathcal{A}^+$ is said to be a *first order translation* of a configuration c if it is the smallest set such that the following holds.

(i) if $\bot_{\text{COOL}} \in c$, then $\bot_{\text{FOL}} \in T$

(ii) for each declarative unit (α, λ) in c, T contains the formula $[\alpha]^*(\lambda)$

(iii) the subset D of R-formulas of c is also a subset of T. □

The first order translation of a configuration c is denoted by $F(c)$.

Notation When writing $[\pi]^*$ or π^* for some π that is a declarative unit, or an R-formula, we mean the single element x in the first order translation of the configuration consisting of only π. R-formulas are translated to themselves, so often we will omit the star in, for example, Rxy^*.

2.2 Cool-Semantic

Definition 2.20. A configuration c is said to *entail a declarative unit* (α, λ) iff $F(c) \cup \mathcal{A}^+ \models_{\text{FOL}} [\alpha]^*(\lambda)$. We write $c \models_{\text{COOL}} \alpha : \lambda$ if c entails (α, λ). □

Definition 2.21. A configuration c is said to *entail an R-formula* ρ iff $F(c) \cup \mathcal{A}^+ \models_{\text{FOL}} \rho$. We write $c \models_{\text{COOL}} \rho$ if c entails ρ. □

Instead of writing $F(c) \cup \mathcal{A}^+ \models_{\text{FOL}} \chi^*$, we write $F(c) \models_{A^+} \chi^*$

3 Properties of configurations

The specification of COOL is now complete. COOL is the triple

$$((L, L_L), \mathcal{A}^+, \vdash_{\text{COOL}})$$

as defined in the last section. In this section we make some essential definitions.

Definition 3.1. If a first order formula ϕ is first order derivable from a first order theory Γ and $\mathcal{A}^+$, we write

$$\Gamma \vdash_{\mathcal{A}^+} \phi$$

Definition 3.2. A configuration c is said to be *inconsistent* if its first order translation is first order inconsistent, $F(c) \vdash_{\mathcal{A}^+} \bot_{\text{FOL}}$. If a configuration is not inconsistent, it is said to be *consistent*.

The following result isn't needed in the rest of the paper, but it is quite neat.

Theorem 3.3. *If a configuration c is inconsistent, there is a finite subset c' of c that is inconsistent.*

Proof. Assume that c is an inconsistent configuration, and that c is infinite. (If c is finite, we are done.) Then $F(c)$ is first order inconsistent by definition. So by compactness of first order logic there is a finite subset Σ of $F(c)$ that is inconsistent. let $\Delta = \Sigma \setminus \mathcal{A}^+$. Then Δ is finite.

Take c' to be a configuration such that $F(c') = \Delta$ (This is OK since any subset of a configuration also must be a configuration, according to the definition of configurations). Since Δ is finite, so is c', by the definition of first order translation. Now, $\Delta \cup \mathcal{A}^+ = F(c')$ is inconsistent, so by definition of consistent configurations, c' is inconsistent. □

4 Soundness and Completeness

In the first subsection we show soundness of the relation $\vdash_{\text{COOL}}$ with respect to the COOL-semantics, and in the second subsection we show completeness.

4.1 Soundness

We need a few lemmas for the proof of soundness.

Lemma 4.1. *If $c \nvdash_{COOL} \bot_{COOL}$, then there is an open element in $P(c)$.*

Proof. We show the contrapositive. Assume that $P(c)$ contains only closed branches. Then for any branch $b \in P(c)$, $\bot_{\text{COOL}} \in b$. Then $P(c \cup \{\neg \bot_{\text{COOL}}\})$ contains only closed branches. Thus, $c \vdash_{\text{COOL}} \bot_{\text{COOL}}$ by definition. □

Lemma 4.2. *Let c and c' be configurations and let $\mathcal{R}^-$ be $\mathcal{R} \setminus \mathcal{R}_S$*

(i) If $\mathcal{R}^- cc'$ then $F(c) \vdash_{\mathcal{A}^+} F(c')$

(ii) Let π and π' be a pair of formulas such that $(c, c \cup \{\pi\}) \in \mathcal{R}_S$ and $(c, c \cup \{\pi'\}) \in \mathcal{R}_S$ by the same application of a $\mathcal{R}_S$-rule. Then at least one of $c \cup \{\pi\}$ and $c \cup \{\pi'\}$ is consistent if c is consistent.

Proof. We start with the first part of the lemma. We show by cases (corresponding to the cases in the definition of $\mathcal{R}$) that if $\mathcal{R}^- cc'$ holds, then $F(c) \vdash_{A^+} F(c')$.

Assume in each of the following cases that $\mathcal{R}^- cc'$ holds.

1. Assume that $c((\alpha \wedge \beta) : (x, y))$ and $c' = c \cup \{\alpha : (x, y), \beta : (x, y)\}$.

   Then $F(c)$ contains $[\alpha \wedge \beta]^*(x, y)$. Then the $\mathcal{A}^+(3)$ gives $F(c) \vdash_{A^+} [\alpha]^*(x, y) \wedge [\beta]^*(x, y)$. So by first order derivability, $F(c) \vdash_{A^+} F(c) \cup \{[\alpha]^*(x, y), [\beta]^*(x, y)\}$. But this is to say that $F(c) \vdash_{A^+} F(c')$.

2. Assume that $c(\neg\neg\alpha : (x, y))$ and $c' = c \cup \{\alpha : (x, y)\}$.

   Immediate from $\mathcal{A}^+(2)$.

3. Assume that $c(\bot_{\text{COOL}})$ and that c' is any configuration.

   Since $\bot_{\text{FOL}}$then is an element of $F(c)$, anything can be deduced from $F(c)$, in particular all elements of $F(c')$.

4. Assume that $c(\chi, \neg\chi)$ for some COOL-formula χ, and that $c' = c \cup \{\bot_{\text{COOL}}\}$.

   Then $[\chi]^*(x, y), [\neg\chi]^*(x, y) \in F(c)$. By theorem 2.18, $F(c) \vdash_{A^+} \neg[\chi]^*(x, y)$. By first order logic, $F(c) \vdash_{A^+} [\chi]^*(x, y) \wedge \neg[\chi]^*(x, y)$, and so $F(c) \vdash_{A^+} \bot_{\text{FOL}}$. Hence $F(c) \vdash_{A^+} F(c')$.

5. Assume that $c(\mathsf{B}\alpha : (x, y),\ Rny)$, and $c' = c \cup \{\alpha : (n, y)\}$.

   Then $F(c)$ contains $[\mathsf{B}\alpha]^*(x, y)$ and Rny. By $\mathcal{A}^+(5)$ and $F(c) \vdash_{A^+} [\mathsf{B}\alpha]^*(x, y)$, we have $F(c) \vdash_{A^+} \forall z(Rzy \to [\alpha]^*(z, y))$. Hence $F(c) \vdash_{A^+} F(c')$ follows by $\forall$-elimination and modus ponens.

6. Assume that $c(\mathsf{N}\alpha : (x, y),\ \bar{R}ny)$, and $c' = c \cup \{\alpha : (n, y)\}$.

   Proved in a way similar to the previous case.

7. Assume that $c(\mathsf{O}\alpha : (x, y))$, and $c' = c \cup \{\mathsf{B}\alpha : (x, y),\ \mathsf{N}\neg\alpha : (x, y)\}$.

   Immediate from $\mathcal{A}^+(9)$.

8. Assume that $c(\neg \mathsf{B}\alpha : (x,y))$, and $c' = c \cup \{Rf_\alpha(y)y,\ \neg\alpha : (f_\alpha(y), y)\}$.

   $F(c)$ then contains $[\neg \mathsf{B}\alpha]^*(x,y)$. Hence, using $\mathcal{A}^+(6)$, we get $F(c) \vdash_{A^+} Rf_\alpha(y)y \wedge [\neg\alpha]^*(f_\alpha(y), y)$. So $F(c) \vdash_{A^+} F(c')$.

9. Assume that $c(\neg \mathsf{N}\alpha : (x,y))$, and $c' = c \cup \{\bar{R}g_\alpha(y)y,\ \neg\alpha : (g_\alpha(y), y)\}$.

   Similar proof again.

10. Assume that $c(\neg Rny)$ and $c' = c \cup \{\bar{R}ny\}$ for some n.

    $F(c)$ then contains $\neg Rny$. This gives us immediately $F(c) \vdash_{A^+} \bar{R}ny$ by $\mathcal{A}$. So $F(c) \vdash_{A^+} F(c')$.

    The three cases with (i) $c(\neg\bar{R}ny)$, $c' = c \cup \{Rny\}$, (ii) $c(Rny)$ and $c' = c \cup \{\neg\bar{R}ny\}$, and (iii) $c(\bar{R}ny)$ and $c' = c \cup \{\neg Rny\}$ for some n are all shown in the same way.

11. the cases for (i) $c(\exists x \phi(x,y))$ and $c' = c \cup \{\phi(n,y)\}$ for some term (n,y), respectively (ii) $c(\forall x \phi(x,y))$ and $c' = c \cup \{\phi(n,y)\}$ for all terms (n,y) in $t(c)$, where ϕ is either R or $\bar{R}$, follow immediately.

To prove the second part, let c, c', and c'' be configurations such that that $\mathcal{R}_S cc'$ and $\mathcal{R}_S cc''$. Assume that c is consistent. Recall that this means that $\mathcal{A}^+ \cup F(c)$ is first order consistent. We show that at least one of c' and c'' is consistent.

1. Assume that c contains $\neg(\alpha \wedge \beta) : (x,y)$, and that $c' = c \cup \{\neg\alpha : (x,y)\}$ and $c'' = c \cup \{\neg\beta : (x,y)\}$. Assume that both c' and c'' are inconsistent. By first order logic, since $F(c') = F(c) \cup \{[\neg\alpha]^*(x,y)\}$ and $F(c'') = F(c) \cup \{[\neg\beta]^*(x,y)\}$, we have $F(c) \vdash_{A^+} \neg[\neg\alpha]^*(x,y)$ and $F(c) \vdash_{A^+} \neg[\neg\beta]^*(x,y)$. So $F(c) \vdash_{A^+} \neg[\neg\alpha]^*(x,y) \wedge \neg[\neg\beta]^*(x,y)$, which gives $F(c) \vdash_{A^+} \neg([\neg\alpha]^*(x,y) \vee [\neg\beta]^*(x,y))$. But $F(c)$ contains $[\neg(\alpha \wedge \beta)]^*(x,y)$, so by $\mathcal{A}^+(4)$ $F(c) \vdash_{A^+} [\neg\alpha]^*(x,y) \vee \neg[\neg\beta]^*(x,y)$, which gives a contradiction since we had assumed that c was consistent. Hence at least one of c' and c'' is consistent.

2. Assume that $c(\neg \mathsf{O}\alpha : (x,y))$, $c' = c \cup \{\neg \mathsf{B}\alpha : (x,y)\}$ and $c'' = c \cup \{\neg \mathsf{N}\neg\alpha : (x,y)\}$. Assume that both c' and c'' are inconsistent. Then a similar argument to that above gives that $F(c) \vdash_{A^+} \neg([\neg \mathsf{B}\alpha]^*(x,y) \vee [\neg \mathsf{N}\neg\alpha]^*(x,y))$. By $[\neg \mathsf{O}\alpha]^*(x,y) \in F(c)$ and $\mathcal{A}^+(10)$ we get a contradiction. Hence, at least one of c' and c'' is consistent.

3. Assume that $c' = c \cup \{Rny\}$ and $c'' = c \cup \{\bar{R}ny\}$ for some $(n, y) \in t(c)$. Assume that both c' and c'' are inconsistent. Then by the inconsistency of c' we get $F(c) \vdash_{A^+} \neg Rny$, and by the inconsistency of c'' we get $F(c) \vdash_{A^+} \neg \bar{R}ny$. From $\mathcal{A}$ and $F(c) \vdash_{A^+} \neg Rny$ we have $F(c) \vdash_{A^+} \bar{R}ny$, so c is inconsistent which contradicts the assumption that c was consistent.

4. Assume that $c' = c \cup \{\exists x Rxy\}$ and $c'' = c \cup \{\forall x \bar{R}xy\}$ Assume that both c' and c'' are inconsistent. Then $F(c) \vdash_{A^+} \neg \exists x Rxy$ and $F(c) \vdash_{A^+} \neg \forall x \bar{R}xy$. Then $F(c) \vdash_{A^+} \forall x \neg Rxy$ which is equivalent to $F(c) \vdash_{A^+} \forall x \bar{R}xy$. This contradict the assumption that c was consistent. Hence at least one of c' and c'' is consistent.

5. Assume that $c' = c \cup \{\exists x \bar{R}xy\}$ and $c'' = c \cup \{\forall x Rxy\}$. Then a similar argument to that above proves that at least one of c' and c'' is consistent.

□

In the proof below we make extensive use of the dual view on branches in $P(c)$. We treat them as sequences or configurations without warning.

Lemma 4.3. *Let c be a configuration. If $F(c) \nvdash_{A^+} \bot$ then there is a branch b in $P(c)$ such that $F(b) \nvdash_{A^+} \bot$.*

Proof. We use induction on the structure of the 'proof tree'. Assume first that $P(c)$ contains only one branch $b = \{c_i\}_{i \geq 0}$, with $c = c_0$. Then $\mathcal{R}^- c_{i-1} c_i$ for all $i > 0$. (Otherwise $P(c)$ would have contained at least two branches.)

Assume $F(b) \vdash_{A^+} \bot$ for contradiction. Then for some $c_k \in b$, $F(c_k) \vdash_{A^+} \bot$. But $\vdash_{A^+}$ is transitive and by lemma 4.2, $F(c_{i-1}) \vdash_{A^+} F(c_i)$ for all $i > 0$. Hence, $F(c_0) \vdash_{A^+} F(c_k)$, so since $F(c_k) \vdash_{A^+} \bot$ we have $F(c_0) \vdash_{A^+} \bot$ or equivalently $F(c) \vdash_{A^+} \bot$. Contradiction, so $F(b) \nvdash_{A^+} \bot$. Then there is a branch b in $P(c)$ such that $F(b) \nvdash_{A^+} \bot$.

Assume now that $P(c)$ contains k branches and that there is a branch $b \in P(c)$ such that $F(b) \nvdash_{A^+} \bot$. Assume without loss of generality that b is the only such branch. We show that if $P(c)'$ containing $k+1$ branches is obtained from $P(c)$ by applying a split rule on b, then also $P(c)'$ contains at least one consistent branch. $P(c)' = (P(c) \setminus \{b\}) \cup \{b + c', b + c''\}$, where '+' is a concatenation operation.

Now, by lemma 4.2, since $F(b) \nvdash_{A^+} \bot$, at least one of $b + c'$ and $b + c''$ is consistent, or in other words, $F(b + c') \nvdash_{A^+} \bot$ or $F(b + c'') \nvdash_{A^+} \bot$.

□

Theorem 4.4. *Let c be a configuration. Then*

$$c \vdash_{COOL} \bot \Rightarrow F(c) \vdash_{\mathcal{A}^+} \bot$$

Proof. Assume $F(c) \nvdash_{\mathcal{A}^+} \bot$. By lemma 4.3, we then have that $P(c)$ contains at least one branch b such that $F(b) \nvdash_{\mathcal{A}^+} \bot$. In particular, this means that $\bot \notin b$, so there is an open branch in $P(c)$ by definition. Again by definition, this means that $c \nvdash_{\text{COOL}} \bot$.

□

Corollary 4.5 (Soundness). *Let c be a configuration.*

$$\textit{If } c \vdash_{COOL} \bot, \textit{ then } c \models_{COOL} \bot$$

Proof. Assume $c \vdash_{\text{COOL}} \bot$. By Theorem 4.4 we have $F(c) \vdash_{\mathcal{A}^+} \bot$, which implies $F(c) \models_{\mathcal{A}^+} \bot$ by first order soundness. By the definition of semantic entailment, we then have $c \models_{\text{COOL}} \bot$.

□

4.2 Completeness

In the following proof, we assume that $P(c \cup \{\neg\bot\})$ is constructed with the systematic and fair proof procedure from lemma 2.15.

Theorem 4.6 (Completeness). *Let c be a configuration. Then*

$$c \models_{COOL} \bot \Rightarrow c \vdash_{COOL} \bot$$

Proof. We show the contrapositive. Assume $c \nvdash_{\text{COOL}} \bot$. Then there is an open branch in $P(c \cup \{\neg\bot\})$. By lemma 2.15 there is then a saturated open branch $b \in P(c \cup \{\neg\bot\})$.

Recall the definition of $\models_{\text{COOL}}$. To show that $c \not\models_{\text{COOL}} \bot$, we have to show that $\mathcal{A}^+ \cup F(c) \not\models_{\text{FOL}} \bot$. In essence, that $\mathcal{A}^+ \cup F(c)$ is satisfiable.

Consider now the first order translation $F(b)$ of b. We first show that $F(b)$ is a Hintikka set and thus has a model $\mathcal{M}$. Then we show that $\mathcal{M}$ also is a model for $\mathcal{A}^+$. Observe that by the definition of first order translation, $\chi^* \in F(b) \leftrightarrow \chi \in b$. We now consider the elements of $F(b)$.

(i) Observe that for declarative units $\alpha : (x, y)$ in b, $[\alpha]^*(x, y)$ is always positive in $F(b)$, by the definition of first order translation. Hence, if $[\alpha]^*(x, y) \in F(b)$ then $\neg[\alpha]^*(x, y) \notin F(b)$.

(ii) Assume that $Rxy \in F(b)$. Then $Rxy \in b$. Since b is open, $\neg Rxy \notin b$, and hence $\neg Rxy \notin F(b)$. Same argument for the cases where (*i*) $\neg Rxy \in F(b)$, (*ii*) $\bar{R}xy \in F(b)$, and (*iii*) $\neg \bar{R}xy \in F(b)$.

(iii) Let $\phi(x) \in \{Rxy, \neg Rxy, \bar{R}xy, \neg \bar{R}xy\}$. Assume $\exists x \phi(x) \in F(b)$. Then $\exists x \phi(x) \in b$. Then by $\mathcal{R}_F$-rules and saturation of b, $\phi(a) \in b$ for some constant a. Hence, then $\phi(a) \in F(b)$ by definition of first order translation.

(iv) Let $\phi(x) \in \{Rxy, \neg Rxy, \bar{R}xy, \neg \bar{R}xy\}$. Assume $\neg \exists x \phi(x) \in F(b)$. Then $\neg \exists x \phi(x) \in b$, so by $\mathcal{R}_F$-rules and saturation of b, $\phi(a) \in b$ for all constant a in $t(b)$. But then $\phi(a) \in F(b)$ for all constants a in $t(b)$.

Observe now that formulas of no other form than those listed above will ever be elements of the first order translation of an open branch. This shows that $F(b)$ is indeed a Hintikka set, and thus, as we know from model theory, has a model $\mathcal{M}$. We restrict $\mathcal{M}$ so that for literals ϕ, $\mathcal{M} \models_{\text{FOL}} \phi$ iff $\phi \in F(b)$.

It now remains to show that $\mathcal{M} \models_{\text{FOL}} \mathcal{A}^+$. We need to consider all the axiom schemata in $\mathcal{A}^+$ and the single axiom that is in $\mathcal{A}$ but not in $\mathcal{A}^+$.

(i) $\forall x \forall y (\neg[\alpha]^*(x,y) \vee \neg[\neg\alpha]^*(x,y))$

Assume that $\mathcal{M} \not\models_{\text{FOL}} \neg[\alpha]^*(x,y)$. Then $\mathcal{M} \models_{\text{FOL}} [\alpha]^*(x,y)$. So $[\alpha]^*(x,y) \in F(b)$, and thereby $\alpha : (x,y) \in b$. Since b is open we have $\neg\alpha : (x,y) \notin b$, so $[\neg\alpha]^*(x,y) \notin F(b)$. Then by the restriction on $\mathcal{M}$, $\mathcal{M} \not\models_{\text{FOL}} [\neg\alpha]^*(x,y)$, and thus $\mathcal{M} \models_{\text{FOL}} \neg[\neg\alpha]^*(x,y)$.
Assuming instead that $\mathcal{M} \not\models_{\text{FOL}} \neg[\neg\alpha]^*(x,y)$, we arrive at $\mathcal{M} \models_{\text{FOL}} \neg[\alpha]^*(x,y)$ by a similar argument.

(ii) $\forall x \forall y ([\neg\neg\alpha]^*(x,y) \rightarrow [\alpha]^*(x,y))$

Assume that $\mathcal{M} \models_{\text{FOL}} [\neg\neg\alpha]^*(x,y)$. Then $\neg\neg\alpha : (x,y) \in b$. Since b is saturated, b also contains $\alpha : (x,y)$, so $\mathcal{M} \models_{\text{FOL}} [\alpha]^*(x,y)$.

(iii) $\forall x \forall y ([\alpha \wedge \beta]^*(x,y) \rightarrow [\alpha]^*(x,y) \wedge [\beta]^*(x,y))$

Assume that $\mathcal{M} \models_{\text{FOL}} [\alpha \wedge \beta]^*(x,y)$. Then $[\alpha \wedge \beta]^*(x,y) \in F(b)$. Since b is saturated, also $[\alpha]^*(x,y)$ and $[\beta]^*(x,y)$ are in $F(b)$. But then $\mathcal{M} \models_{\text{FOL}} [\alpha]^*(x,y)$ and $\mathcal{M} \models_{\text{FOL}} [\beta]^*(x,y)$.

(iv) $\forall x \forall y ([\neg(\alpha \wedge \beta)]^*(x,y) \rightarrow [\neg\alpha]^*(x,y) \vee [\neg\beta]^*(x,y))$

Assume that $\mathcal{M} \models_{\text{FOL}} [\neg(\alpha \wedge \beta)]^*(x,y)$. Then $[\neg(\alpha \wedge \beta)]^*(x,y) \in F(b)$ so at least one of $[\neg\alpha]^*(x,y)$ and $[\neg\beta]^*(x,y)$ is also in $F(b)$ by saturation and openness of b. The axiom follows.

(v) $\forall x \forall y([\mathsf{B}\alpha]^*(x,y) \to \forall z(Rzy \to [\alpha]^*(z,y)))$

Assume $\mathcal{M} \models_{\text{FOL}} [\mathsf{B}\alpha]^*(x,y)$. Assume for some n that $\mathcal{M} \models_{\text{FOL}} Rny$. Then $\mathsf{B}\alpha : (x,y) \in b$ and $Rny \in b$. Then by saturation also $\alpha : (n,y) \in b$, so $[\alpha]^*(n,y) \in F(b)$. n was chosen arbitrarily, so $\mathcal{M} \models_{\text{FOL}} \forall z(Rzy \to [\alpha]^*(z,y))$. The axiom follows.

(vi) $\forall x \forall y([\mathsf{N}\alpha]^*(x,y) \to \forall z(\bar{R}zy \to [\alpha]^*(z,y)))$

Proved similarly to the previous case.

(vii) $\forall x \forall y([\neg\mathsf{B}\alpha]^*(x,y) \to Rf_\alpha(y)y \wedge [\neg\alpha]^*(f_\alpha(y),y))$

Assume $\mathcal{M} \models_{\text{FOL}} [\neg\mathsf{B}\alpha]^*(x,y)$. Then $\neg\mathsf{B}\alpha : (x,y) \in b$. So by saturation of b, both $Rf_\alpha(y)y$ and $\neg\alpha : (f_\alpha(y),y) \in b$. But then both $Rf_\alpha(y)y$ and $[\neg\alpha]^*(f_\alpha(y),y))$ are in $F(b)$, and so $\mathcal{M} \models_{\text{FOL}} Rf_\alpha(y)y$ and $\mathcal{M} \models_{\text{FOL}} [\neg\alpha]^*(f_\alpha(y),y)$.

(viii) $\forall x \forall y([\neg\mathsf{N}\alpha]^*(x,y) \to \bar{R}g_\alpha(y)y \wedge [\neg\alpha]^*(g_\alpha(y),y))$

Proved similarly to the previous case.

(ix) $\forall x \forall y([\mathsf{O}\alpha]^*(x,y) \to [\mathsf{B}\alpha]^*(x,y) \wedge [\mathsf{N}\neg\alpha]^*(x,y))$

Assume $\mathcal{M} \models_{\text{FOL}} [\mathsf{O}\alpha]^*(x,y)$. Then $\mathsf{O}\alpha : (x,y) \in b$, so also $\mathsf{B}\alpha : (x,y) \in b$ and $\mathsf{N}\neg\alpha : (x,y) \in b$ by saturation of b. Hence, $\mathcal{M} \models_{\text{FOL}} [\mathsf{B}\alpha]^*(x,y)$ and $\mathcal{M} \models_{\text{FOL}} [\mathsf{N}\neg\alpha]^*(x,y)$.

(x) $\forall x \forall y([\neg\mathsf{O}\alpha]^*(x,y) \to [\neg\mathsf{B}\alpha]^*(x,y) \vee [\neg\mathsf{N}\neg\alpha]^*(x,y))$

Assume that $\mathcal{M} \models_{\text{FOL}} [\neg\mathsf{O}\alpha]^*(x,y)$. Then $[\neg\mathsf{O}\alpha]^*(x,y) \in F(b)$. So $\neg\mathsf{O}\alpha : (x,y) \in b$. Then by saturation, at least one of $\neg\mathsf{B}\alpha : (x,y)$ and $\neg\mathsf{N}\neg\alpha : (x,y)$ is in b. Hence $\mathcal{M}$ is a model for at least one of $[\neg\mathsf{B}\alpha]^*(x,y)$ and $[\neg\mathsf{N}\neg\alpha]^*(x,y)$.

(xi) The $\mathcal{A}$-axiom: $\forall x \forall y(\neg Rxy \leftrightarrow \bar{R}xy)$

We have that $\mathcal{M} \models_{\text{FOL}} Rxy$ iff $Rxy \in b$ iff $\neg\bar{R}xy \in b$ iff $\mathcal{M} \models_{\text{FOL}} \neg\bar{R}xy$.

In conclusion, $\mathcal{M}$ is a model both for $F(b)$ and $\mathcal{A}^+$, and in particular for $F(c)$, since $F(c) \subseteq F(b)$. But then $\mathcal{M} \models_{\text{FOL}} F(c) \cup \mathcal{A}^+$, so we have indeed $F(c) \cup \mathcal{A}^+ \not\models_{\text{FOL}} \bot$, which by definition is

$$c \not\models_{\text{COOL}} \bot$$

$\square$

5 Future Work

We have devised a sound an complete proof system for a logic we call COOL, for Compiled Only Knowing, with the intension to simulate Levesque's logic of only knowing (OL). Therefore, the first future task is to prove that COOL indeed corresponds to OL. We formulate the theorem here.

Theorem *α is a theorem in* OL *if, and only if,* $\alpha : (0, y)$ *is a theorem in* COOL *for some constant* 0.

The second task is to implement a theorem prover for COOL. Work on this has already begun and a few tests have been conducted in Otter [McCun94]. See appendix for an example script. Further, we have as an objective to generalise OL to incorporate reasoning with varying sizes of the set of worlds M in the OL-structures (x, M). The work we have done with COOL so far might provide a platform for that task. Lastly, it would be interesting to extend COOL to simulate the full language version of only knowing.

References

[Lev90] Levesque, H. J., *All I Know: A Study in Autoepistemic Logic.* Artificial Intelligence, North Holland, 42, 1990, pp. 263-309.

[Ru96] Russo, A., *Modal Logics as Labelled Deductive Systems.* PhD. Thesis, Department of Computing, Imperial College, London, 1996.

[BrGaRu00] Broda, K., Gabbay D., Russo, A. *A Unified Compilation Style Labelled Deductive System for Modal, Substructural, and Fuzzy Logics.* in *Discovering the World with Fuzzy Logic: Perspectives and Approaches to Formalization of Human-Consistent Logical Systems.* Editors: Vilem Novak, Irina Perfilieva. Springer Verlag, 2000.

[Gab96] Gabbay, D., *Labelled Deductive Systems, Volume 1 - Foundations.* Oxford University Press, 1996.

[McCun94] McCune W., Otter 3.0 Reference Manual and Guide. Argonne National Laboratory, Argonne, Illinois 60439-4801 1994.

[Che80] B.F. Chellas *Modal Logic*, Cambridge University Press, Cambridge, UK, 1980

Appendix

A A Sample Proof in COOL

(1)	$O(Bp \vee \neg p) : (0, y)$	(data)
(2)	$\neg B \neg p : (0, y)$	(data)
(3)	$\neg p : (1, y)$	(p is not $\top$)
(4)	$B(Bp \vee \neg p) : (0, y)$	(defn. of O)
(5)	$N(\neg Bp \wedge p) : (0, y)$	(similarly)
(6)	$\{p : (f_{\neg p}, y), R(f_{\neg p}, y)\}$	(2)
(7)	$(Bp \vee \neg p) : (f_{\neg p}, y)$	(4)
(8)	$Bp : (f_{\neg p}, y)$	(Split-1a (7))
(9)	$R(1, y)$	(Split-2a)
(10)	$p : (1, y)$	(8,9)
(11)	$\perp : (1, y)$	(3,10)
(12)	$\overline{R}(1, y)$	(Split-2b)
(13)	$(\neg Bp \wedge p) : (1, y)$	(5,12)
(14)	$\neg Bp : (1, y)$	(13)
(15)	$p : (1, y)$	(13)
(16)	$\perp : (1, y)$	(3,15)
(17)	$\neg p : (f_{\neg p}, y)$	(Split-1b)
(18)	$\perp : (f_{\neg p}, y)$	(17,6)

B A Sample Script from Otter

```
INPUT: set(hyper_res). set(prolog_style_variables). list(sos).
p1(0).                      %p1 represents [O(Bp or ~p)]
p3(1).                      %p2 represents [~B~p]
p2(0).                      %p3 represents [~p]
end_of_list. list(usable).
- p1(X) | p4(X).            %p4 represents [B(Bp or ~p)]
- p1(X) | p5(X).            %p5 represents [N(~(Bp or ~p))]
- p2(X) | p9(f).            %p9 represents [~~p]
- p2(X) | R1(f).     %R1 represents R and R2 represents R-bar
- p9(X) | p(X). - p3(X) | - p(X).
- p4(X) | - R1(Y) | p6(Y).  %p6 represents [Bp or ~p]
- p5(X) | - R2(Y) | p10(Y). %p10 represents [~(Bp or ~p)]
```

```
- p6(X) | p7(X) | p3(X).    %p7 represents [Bp]
- p7(X) | - R1(Y) | p(Y).
- p10(X) | p8(X).           %p8 represents [~Bp] (don't need the schema)
- p10(X) | p9(X). R1(X) | R2(X). end_of_list.
```

```
OUTPUT:

1 [] p1(0). 2 [] p3(1). 3 [] p2(0). 4 [] -p1(X) | p4(X).
5 [] -p1(X) | p5(X). 6 [] -p2(X) | p9(f). 7 [] -p2(X) | R1(f).
8 [] -p9(X) | p(X). 9 [] -p3(X) | -p(X). 10 [] -p4(X) | -R1(Y) |
 p6(Y). 11 [] -p5(X) | -R2(Y) | p10(Y). 12 [] -p6(X)
|p7(X) | p3(X). 13 [] -p7(X) | -R1(Y) | p(Y). 15 [] -p10(X) | p9(X).
 16 [] R1(X) | R2(X). 17 [hyper,1,5] p5(0). 18
[hyper,1,4] p4(0). 19 [hyper,3,7] R1(f). 20 [hyper,3,6] p9(f).
 21 [hyper,17,11,16] p10(A) | R1(A). 23 [hyper,19,10,18]
p6(f). 24 [hyper,20,8] p(f). 25 [hyper,23,12] p7(f) | p3(f).
32 [hyper,25,9,24] p7(f). 33 [hyper,32,13,21] p(A) |
p10(A). 41 [hyper,33,9,2] p10(1). 42 [hyper,41,15] p9(1).
44 [hyper,42,8] p(1). 45 [hyper,44,9,2] .
```