A Reasoning System for a First-Order Logic of Limited Belief

Christoph Schwering

UNSW Sydney
What is limited belief? And why?

**Task:** Robot has a KB and a query:

Does the KB *logically entail* the query?
What is limited belief? And why?

**Task:** Robot has a KB and a query:

Does the KB *logically entail* the query?

Which logic?
What is limited belief? And why?

**Task**: Robot has a KB and a query:

Does the KB *logically entail* the query?

Which logic?

**Classical logic**:

- Unrealistic: omniscient agent
- Undecidable *(first-order) / intractable (propositional)*
What is limited belief? And why?

**Task:** Robot has a KB and a query:

Does the KB *logically entail* the query?

Which logic?

**Limited belief:**

- Belief level 0: explicitly written down in the KB
- Belief level $k > 0$: derivable from KB with effort $k$

**Hope:** good results at *small* belief level

Builds on Lakemeyer & Levesque, KR-2016
Language

**FOL** with equality + functions + sorts +

- **Knowledge:** $K_0 \alpha \ K_1 \alpha \ K_2 \alpha \ldots$
- **Possibility:** $M_0 \alpha \ M_1 \alpha \ M_2 \alpha \ldots$

**Example:**

- $K_1 (\text{Rich(Frank)} \lor \text{Rich(Fred)})$  
  know that Frank or Fred is rich
- $\forall x \ M_1 \text{fatherOf}(\text{Sally}) \neq x$  
  don’t know who Sally’s father is
- $K_1 \exists x (\text{fatherOf}(\text{Sally}) = x \land \text{Rich}(x) \land \ M_1 \text{fatherOf}(\text{Sally}) \neq x)$  
  know that Sally’s father is rich, but don’t know who he is
Semantics

**Model:** set of clauses closed under unit propagation

- Belief level 0: subsumption
- Belief level \( k > 0 \): \( k \) case splits

**Example:**
If all we know is (a) \( \text{fatherOf(Sally)} = \text{Frank} \lor \text{fatherOf(Sally)} = \text{Fred} \)
and (b) \( \forall x (\text{fatherOf(Sally)} \neq x \lor \text{Rich(x)}) \)
then \( K_1 (\text{Rich(Frank)} \lor \text{Rich(Fred)}) \)?

**Yes!** Branch on \( \text{fatherOf(Sally)} \):

- \( \{(a), (b), \text{fatherOf(Sally)} = \text{Frank}\} \ni \text{Rich(Frank)} \) by UP with (b)
- \( \{(a), (b), \text{fatherOf(Sally)} = \text{Fred}\} \ni \text{Rich(Fred)} \) by UP with (b)
- \( \{(a), (b), \text{fatherOf(Sally)} = n\} \ni \perp \) by UP with (a) for \( n \neq \text{Frank, Fred} \)
KB entails query at some belief level $\implies$ KB classically entails query
if no $\neg K, \neg M$
Soundness  Compleness  Decidability  Tractability

KB entails query at some belief level $\iff$ KB classically entails query
if no $\neg K$, $\neg M$ and no $\exists, \forall$
KB entails query at some belief level is \textit{decidable}
KB entails query at some belief level is tractable if no ∃ Y and belief level fixed
Experiments: Sudoku Minesweeper

Hypothesis: good results at small belief level
Hypothesis: good results at *small* belief level

Experiments:

Sudoku

Minesweeper

**Average # of cells solved at...**

- clues
- level 0
- level 1
- level 2
- level 3
- level 4
- level 5
Experiments: Sudoku Minesweeper

**Hypothesis:** good results at *small* belief level ✓ ✓

![Bar chart showing winning percentage of games at different sizes (Small, Medium, Large, Huge) with color coding for levels 0, 1, 2, 3 and loss.]

- Small: Green (level 0) and yellow (level 1) segments.
- Medium: Green (level 0) and yellow (level 1) segments with red (level 2) and black (level 3) small segments.
- Large: Yellow (level 1) segments with a red (level 2) and black (level 3) small segment.
- Huge: Green (level 0) and yellow (level 1) segments with red (level 2) and black (level 3) small segments.

**Winning % of games at...**
- level 0
- level 1
- level 2
- level 3
- loss
Limbo = Limited Belief

Demos: www.cse.unsw.edu.au/~cschwering/limbo  
Fri 10:00–12:00

Code: www.github.com/schwering/limbo

Next: 1. actions  2. belief change  3. multiple agents
Appendix
Language in detail

Terms:
■ First-order variables
■ Functions \( f(t_1, \ldots, t_m) \) where each \( t_i \) is a name or variable
■ Standard names infinitely many and sorted

Formulas:
■ FOL: \( t_1 = t_2 \quad \neg \alpha \quad \alpha \lor \beta \quad \exists x \alpha \)
■ Knowledge: \( K_0 \alpha \quad K_1 \alpha \quad K_2 \alpha \quad \ldots \)
■ Possibility: \( M_0 \alpha \quad M_1 \alpha \quad M_2 \alpha \quad \ldots \)
■ Knowledge base: \( O \alpha \) where \( \alpha \) is in universal CNF

\[ \alpha \land \beta \quad \alpha \supset \beta \quad \alpha \equiv \beta \quad \forall x \alpha \] are abbreviations

Predicates are simulated with functions

Existentials in KBs are simulated with Skolem functions

Functions on the right-hand side and within functions are flattened:

\[
\begin{align*}
\text{if } f(\cdot) = g(\cdot) & \quad \text{then } \forall x (g(\cdot) = x \supset f(\cdot) = x) \\
\text{if } f(g(\cdot)) = t & \quad \text{then } \forall x (g(\cdot) = x \supset f(x) = t)
\end{align*}
\]
Literal encoding

- Functions cannot appear on rhs
  \[ f(\cdot) = g(\cdot) \quad \rightarrow \quad \forall x \ g(\cdot) = x \supset f(\cdot) = x \]

- Functions cannot be nested
  \[ f(g(\cdot)) = t \quad \rightarrow \quad \forall x \ g(\cdot) = x \supset f(x) = t \]

- Term is 30-bit number
  - points to full representation
  - this pointer is unique (interning)

- Literal is 64-bit number
  - 30 + 30 bits for lhs + rhs
  - 1 + 1 bits to indicate if lhs + rhs is name
  - 1 bit to indicate whether = or ≠

- Conditions for literal subsumption and complementarity:
  - \( \ell \) subsumes \( \ell \)
  - \( t = n_1 \) subsumes \( t \neq n_2 \)
  - \( t = t' \) and \( t \neq t' \) are complementary
  - \( t = n_1 \) and \( t = n_2 \) are complementary

- Sound and complete

- Bitwise op’s on 64-bit numbers suffice

- Fast clause subsumption and unit propagation

\[ \text{no term dereferencing} \]
"I don’t know Sally’s father, but I know he’s rich"

\[ c_1 = f(S) = \text{Frank} \lor f(S) = \text{Fred} \]
\[ c_2 = \forall x (f(S) \neq x \lor r(x) = \top) \]
\[ O(c_1 \land c_2) \models K\exists x (f(S) = x \land r(x) = \top \land Mf(S) \neq x) \]
“I don’t know Sally’s father, but I know he’s rich”

- \( e = \{ w \mid w \models f(S) = \text{Frank} \lor f(S) = \text{Fred} \land \forall x (f(S) \neq x \lor r(x) = \top) \} \)
- \( e \models \text{K}\exists x (f(S) = x \land r(x) = \top \land \text{M}f(S) \neq x) \)
“I don’t know Sally’s father, but I know he’s rich”

\[ e = \{ w \mid w \models f(S) = \text{Frank} \lor f(S) = \text{Fred} \land \forall x (f(S) \neq x \lor r(x) = \top) \}\]

\[ e \models K\exists x (f(S) = x \land r(x) = \top \land Mf(S) \neq x) \]

For every \( w \in e \), for some \( n \), \( w \models f(S) = n \land R(n) \)

For some \( w' \in e \), \( w \models f(S) \neq n \)
"I don’t know Sally’s father, but I know he’s rich”

\[
\begin{align*}
c_1 &= f(S) = \text{Frank} \lor f(S) = \text{Fred} \\
c_2 &= \forall x (f(S) \neq x \lor r(x) = \top)
\end{align*}
\]

\[
\begin{align*}
\mathbf{O}(c_1 \land c_2) &\models \mathbf{K}_1 \exists x (f(S) = x \land r(x) = \top \land M_1 f(S) \neq x)
\end{align*}
\]
“I don’t know Sally’s father, but I know he’s rich”

\[ s = \{ f(S) = \text{Frank} \lor f(S) = \text{Fred}, \]
\[ f(S) \neq n \lor r(n) = \top \mid n \text{ is a name} \]
\[ s \models K_1 \exists x (f(S) = x \land r(x) = \top \land M_1 f(S) \neq x) \]
“I don’t know Sally’s father, but I know he’s rich”

\[ s = \{ f(S) = \text{Frank} \lor f(S) = \text{Fred}, \]
\[ f(S) \neq n \lor r(n) = \top \mid n \text{ is a name} \}
\[ s \models K_1 \exists x (f(S) = x \land r(x) = \top \land M_1 f(S) \neq x) \]

\[ \iff \]

for some \( t_1 \), for all \( n_1 \), for some \( n \),
\[ s \cup \{ t_1 = n_1 \} \models f(S) = n \land r(n) = \top \land M_1 f(S) \neq n \]
"I don’t know Sally’s father, but I know he’s rich"

\[ s = \{ f(S) = \text{Frank} \lor f(S) = \text{Fred}, \]
\[ f(S) \neq n \lor r(n) = \top \mid n \text{ is a name} \}

\[ s \models K_1 \exists x (f(S) = x \land r(x) = \top \land M_1 f(S) \neq x) \]

\[ \iff \text{for some } t_1, \text{ for all } n_1, \text{ for some } n, \]
\[ s \cup \{ t_1 = n_1 \} \models f(S) = n \land r(n) = \top \land M_1 f(S) \neq n \]

\[ \iff \text{for some } t_1, \text{ for all } n_1, \text{ for some } n, s \cup \{ t_1 = n_1 \} \models f(S) = n \land r(n) = \top \]
\[ \text{for some } t_2 \text{ and } n_2, s \cup \{ t_2 = n_2 \} \models f(S) \neq n \]
"I don’t know Sally’s father, but I know he’s rich"

\[ s = \{ f(S) = \text{Frank} \lor f(S) = \text{Fred}, \]
\[ f(S) \neq n \lor r(n) = \top \mid n \text{ is a name} \]
\[ s \models K_1 \exists x (f(S) = x \land r(x) = \top \land M_1 f(S) \neq x) \]

\[ \iff \]

(a) for some \( t_1 \), for all \( n_1 \), for some \( n \), \( s \cup \{ t_1 = n_1 \} \models f(S) = n \land r(n) = \top \)

(b) for some \( t_2 \) and \( n_2 \), \( s \cup \{ t_2 = n_2 \} \models f(S) \neq n \)

\[ \iff \]

(a) choose \( t_1 = f(S) \):

if \( n_1 = \text{Frank} \), choose \( n = \text{Frank} \):

\( s \cup \{ f(S) = \text{Frank} \} \) contains \( f(S) = \text{Frank}, r(\text{Frank}) = \top \)

(b) choose \( t_2 = f(S) \) and \( n_2 = \text{Fred} \):

\( s \cup \{ f(S) = \text{Fred} \} \) contains \( f(S) \neq \text{Frank} \)

if \( n_1 = \text{Fred} \): analogous

if \( n_1 \neq \text{Frank, Fred} \): \( s \cup \{ f(S) = n_1 \} \) is obv. inconsistent
Theorems in detail

- $\models$ is classical entailment
- $\models^*$ is limited entailment
- $\sigma$ contains no $O$, $\neg K_k$, $\neg M_k$
- $\sigma^*$ removes belief levels
- $\sigma_k$ sets belief levels to $k$

### Soundness & Eventual Completeness

<table>
<thead>
<tr>
<th>Condition</th>
<th>Implication</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O\alpha \models^* \sigma$</td>
<td>$O\alpha \models \sigma^*$</td>
<td>if $\sigma$ without $\neg K_k$, $\neg M_k$</td>
</tr>
<tr>
<td>$O\alpha \models^* \sigma_k$ for some $k$</td>
<td>$O\alpha \models \sigma^*$</td>
<td>if $\alpha$, $\sigma$ quantifier-free</td>
</tr>
</tbody>
</table>

### Complexity

- $O\alpha \models^* \sigma$ is decidable
- $O\alpha \models^* \sigma_k$ is tractable in $O(2^k(|\alpha| + |\sigma|)^{k+3})$ if $\alpha$, $\sigma$ quantifier-free
Semantics in detail

- \((\neg) t = n\)
- \((\alpha \lor \beta)\)
- \(\neg(\alpha \lor \beta)\)
- \(\exists x \alpha\)
- \(\neg \exists x \alpha\)
- \(\neg \neg \alpha\)
- \(K_0 \alpha\)
- \(K_{k+1} \alpha\)
- \(M_0 \alpha\)
- \(M_{k+1} \alpha\)
- \(O \alpha\)
Semantics in detail

- \( s \models (¬)t = n \) iff \( (¬)t = n \in s \)
- \( s \models (\alpha \lor \beta) \) iff \((\alpha \lor \beta) \in s \) or \( s \models \alpha \) or \( s \models \beta \)
- \( s \models ¬(\alpha \lor \beta) \) iff \( s \models ¬\alpha \) and \( s \models ¬\beta \)
- \( s \models \exists x \alpha \) iff \( s \models \alpha^x_n \) for some name \( n \)
- \( s \models ¬\exists x \alpha \) iff \( s \models ¬\alpha^x_n \) for every name \( n \)
- \( s \models ¬¬\alpha \) iff \( s \models \alpha \)

- \( K_0 \alpha \)
- \( K_{k+1} \alpha \)
- \( M_0 \alpha \)
- \( M_{k+1} \alpha \)
- \( O \alpha \)
Semantics in detail

- $s \models (\neg) t = n$ iff $(\neg) t = n \in s$
- $s \models (\alpha \lor \beta)$ iff $(\alpha \lor \beta) \in s$ or $s \models \alpha$ or $s \models \beta$
- $s \models \neg (\alpha \lor \beta)$ iff $s \models \neg \alpha$ and $s \models \neg \beta$
- $s \models \exists x \alpha$ iff $s \models \alpha_n$ for some name $n$
- $s \models \neg \exists x \alpha$ iff $s \models \neg \alpha_n$ for every name $n$
- $s \models \neg \neg \alpha$ iff $s \models \alpha$

- $s \models K_0 \alpha$ iff $s$ is obviously inconsistent or $s \models \alpha$
- $s \models K_{k+1} \alpha$ iff for some $t$ and all $n$, $s \cup \{t = n\} \models K_k \alpha$
- $s \models M_0 \alpha$ iff $s$ is obviously consistent and $s \models \alpha$
- $s \models M_{k+1} \alpha$ iff for some $t$ and $n$, $s \cup \{t = n\} \models M_k \alpha$
- $s \models O \alpha$ iff $s$ is minimal s.t. $s \models \alpha$
Semantics in detail

- $s \models (\neg t = n)$ iff $(\neg) t = n \in s$
- $s \models (\alpha \lor \beta)$ iff $(\alpha \lor \beta) \in s$ or $s \models \alpha$ or $s \models \beta$
- $s \models \neg(\alpha \lor \beta)$ iff $s \models \neg \alpha$ and $s \models \neg \beta$
- $s \models \exists x \alpha$ iff $s \models \alpha^x_n$ for some name $n$
- $s \models \neg \exists x \alpha$ iff $s \models \neg \alpha^x_n$ for every name $n$
- $s \models \neg \neg \alpha$ iff $s \models \alpha$

- $s \models K_0 \alpha$ iff $s$ is obviously inconsistent or $s \models \alpha$
- $s \models K_{k+1} \alpha$ iff for some $t$ and all $n$, $s \cup \{t = n\} \models K_k \alpha$
- $s \models M_0 \alpha$ iff $s$ is obviously consistent and $s \models \alpha$
- $s \models M_{k+1} \alpha$ iff for some $t$ and $n$, $s \cup \{t = n\} \models M_k \alpha$
- $s \models O \alpha$ iff $s$ is minimal s.t. $s \models \alpha$

obviously inconsistent \(\wedge\) contains the empty clause
obviously consistent \(\wedge\) not potentially inconsistent
potentially inconsistent \(\wedge\)

(a) obviously consistent
(b) two unsubsumed clauses mention two complementary literals
(c) for every name $n$, $t \neq n$ occurs in an unsubsumed clause
Semantics in detail

- \( s_0, s, \nu \models (\neg)t = n \iff (\neg)t = n \in s \)
- \( s_0, s, \nu \models (\alpha \lor \beta) \iff (\alpha \lor \beta) \in s \) or \( s_0, s, \nu \models \alpha \) or \( s_0, s, \nu \models \beta \)
- \( s_0, s, \nu \models \neg(\alpha \lor \beta) \iff s_0, s, \nu \models \neg\alpha \) and \( s_0, s, \nu \models \neg\beta \)
- \( s_0, s, \nu \models \exists x \alpha \iff s_0, s, \nu \models \alpha_n^x \) for some name \( n \)
- \( s_0, s, \nu \models \neg\exists x \alpha \iff s_0, s, \nu \models \neg\alpha_n^x \) for every name \( n \)
- \( s_0, s, \nu \models \neg\neg\alpha \iff s_0, s, \nu \models \alpha \)

- \( s_0, s, \nu \models K_0\alpha \iff s_0 \cup \nu \) is obv. inconsistent or \( s_0, s_0 \cup \nu, \emptyset \models \alpha \)
- \( s_0, s, \nu \models K_{k+1}\alpha \iff \) for some \( t \) and all \( n \), \( s_0, s, \nu \cup \{t = n\} \models K_k\alpha \)
- \( s_0, s, \nu \models M_0\alpha \iff s_0 \cup \nu \) is obv. consistent and \( s_0, s_0 \cup \nu, \emptyset \models \alpha \)
- \( s_0, s, \nu \models M_{k+1}\alpha \iff \) for some \( t \) and \( n \), \( s_0, s, \nu \cup \{t = n\} \models M_k\alpha \)
- \( s_0, s, \nu \models O\alpha \iff s_0 \) is minimal s.t. \( s_0, s_0, \emptyset \models \alpha \)

obviously inconsistent \( \widehat{=} \) contains the empty clause
obviously consistent \( \widehat{=} \) not potentially inconsistent
potentially inconsistent \( \hat{=} \)

(a) obviously consistent
(b) two unsubsumed clauses mention two complementary literals
(c) for every name \( n \), \( t \neq n \) occurs in an unsubsumed clause