The Propositional Calculus via Logical Journey of the Zoombinis

Selmer Bringsjord

Rensselaer AI & Reasoning (RAIR) Lab
Department of Cognitive Science
Department of Computer Science
Lally School of Management & Technology
Rensselaer Polytechnic Institute (RPI)
Troy, New York 12180 USA

Intro to Logic-based AI
9/12/2024

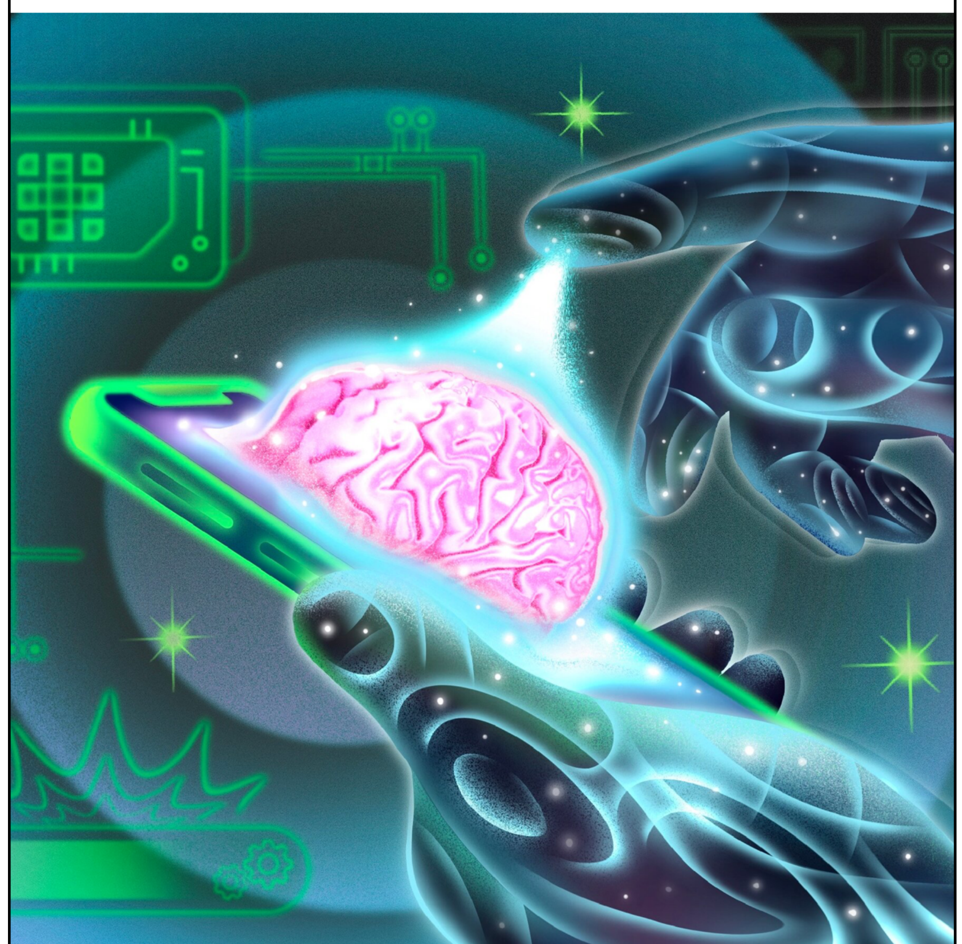
Logic-and-AI in the news

. . .

How to Turn Your Old iPhone Into an A.I. Phone (and Skip the Upgrade)

Apple is using Apple Intelligence, a suite of tools for generating images and text, to upsell the iPhone 16. But you can get similar features elsewhere.

Listen to this article · 5:38 min Learn more



Sisi Yu

By **Brian X. Chen**
Brian X. Chen, The Times's lead consumer

The Propositional Calculus …

CHAPTER 7

LOGICAL AGENTS

In which we design agents that can form representations of a complex world, use a process of inference to derive new representations about the world, and use these new representations to deduce what to do.

Knowledge-based agents
Reasoning
Representation

Humans, it seems, know things; and what they know helps them do things. In AI, **knowledge-based agents** use a process of **reasoning** over an internal **representation** of knowledge to decide what actions to take.

The problem-solving agents of Chapters 3 and 4 know things, but only in a very limited, inflexible sense. They know what actions are available and what the result of performing a specific action from a specific state will be, but they don't know general facts. A route-finding agent doesn't know that it is impossible for a road to be a negative number of kilometers long. An 8-puzzle agent doesn't know that two tiles cannot occupy the same space. The knowledge they have is very useful for finding a path from the start to a goal, but not for anything else.

The atomic representations used by problem-solving agents are also very limiting. In a partially observable environment, for example, a problem-solving agent's only choice for representing what it knows about the current state is to list all possible concrete states. I could give a human the goal of driving to a U.S. town with population less than 10,000, but to say that to a problem-solving agent, I could formally describe the goal only as an explicit set of the 16,000 or so towns that satisfy the description.

Chapter 6 introduced our first factored representation, whereby states are represented as assignments of values to variables; this is a step in the right direction, enabling some parts of the agent to work in a domain-independent way and allowing for more efficient algorithms. In this chapter, we take this step to its logical conclusion, so to speak—we develop **logic** as a general class of representations to support knowledge-based agents. These agents can combine and recombine information to suit myriad purposes. This can be far removed from the needs of the moment—as when a mathematician proves a theorem or an astronomer calculates the Earth's life expectancy. Knowledge-based agents can accept new tasks in the form of explicitly described goals; they can achieve competence quickly by being told or learning new knowledge about the environment; and they can adapt to changes in the environment by updating the relevant knowledge.

We begin in Section 7.1 with the overall agent design. Section 7.2 introduces a simple new environment, the wumpus world, and illustrates the operation of a knowledge-based agent without going into any technical detail. Then we explain the general principles of **logic** in Section 7.3 and the specifics of **propositional logic** in Section 7.4. Propositional logic is a factored representation; while less expressive than **first-order logic** (Chapter 8), which is the canonical structured representation, propositional logic illustrates all the basic concepts

CHAPTER 7

LOGICAL AGEN

In which we design agents that can f
of inference to derive new representa
tions to deduce what to do.

Knowledge-based agents
Reasoning
Representation

Humans, it seems, know things; and w
based agents use a process of **reaso**
decide what actions to take.

The problem-solving agents of Ch
inflexible sense. They know what ac
specific action from a specific state wi
agent doesn't know that it is impossib
An 8-puzzle agent doesn't know that t
they have is very useful for finding a p

The atomic representations used
a partially observable environment, f
representing what it knows about the c
give a human the goal of driving to a
that to a problem-solving agent, I cou
the 16,000 or so towns that satisfy the

Chapter 6 introduced our first fac
assignments of values to variables; thi
the agent to work in a domain-indepe
In this chapter, we take this step to its
general class of representations to sup
bine and recombine information to su
needs of the moment—as when a ma
lates the Earth's life expectancy. Kno
of explicitly described goals; they can
new knowledge about the environmen
updating the relevant knowledge.

We begin in Section 7.1 with the
new environment, the wumpus worl
agent without going into any technica
in Section 7.3 and the specifics of **pr**
a factored representation; while less c
the canonical structured representatio

KB is true in the real world? (After all, *KB* is just "syntax" inside the agent's head.) This is a philosophical question about which many, many books have been written. (See Chapter 27.) A simple answer is that the agent's sensors create the connection. For example, our wumpus-world agent has a smell sensor. The agent program creates a suitable sentence whenever there is a smell. Then, whenever that sentence is in the knowledge base, it is true in the real world. Thus, the meaning and truth of percept sentences are defined by the processes of sensing and sentence construction that produce them. What about the rest of the agent's knowledge, such as its belief that wumpuses cause smells in adjacent squares? This is not a direct representation of a single percept, but a general rule—derived, perhaps, from perceptual experience but not identical to a statement of that experience. General rules like this are produced by a sentence construction process called **learning**, which is the subject of Part V. Learning is fallible. It could be the case that wumpuses cause smells *except on February 29 in leap years*, which is when they take their baths. Thus, *KB* may not be true in the real world, but with good learning procedures, there is reason for optimism.

7.4 Propositional Logic: A Very Simple Logic

We now present **propositional logic**. We describe its syntax (the structure of sentences) and its semantics (the way in which the truth of sentences is determined). From these, we derive a simple, syntactic algorithm for logical inference that implements the semantic notion of entailment. Everything takes place, of course, in the wumpus world. Propositional logic

7.4.1 Syntax

The **syntax** of propositional logic defines the allowable sentences. The **atomic sentences** consist of a single **proposition symbol**. Each such symbol stands for a proposition that can be true or false. We use symbols that start with an uppercase letter and may contain other letters or subscripts, for example: P, Q, R, $W_{1,3}$ and *FacingEast*. The names are arbitrary but are often chosen to have some mnemonic value—we use $W_{1,3}$ to stand for the proposition that the wumpus is in [1,3]. (Remember that symbols such as $W_{1,3}$ are *atomic*, i.e., W, 1, and 3 are not meaningful parts of the symbol.) There are two proposition symbols with fixed meanings: *True* is the always-true proposition and *False* is the always-false proposition. **Complex sentences** are constructed from simpler sentences, using parentheses and operators called **logical connectives**. There are five connectives in common use: Atomic sentences Proposition symbol Complex sentences Logical connectives

- $\neg$ (not). A sentence such as $\neg W_{1,3}$ is called the **negation** of $W_{1,3}$. A **literal** is either an atomic sentence (a **positive literal**) or a negated atomic sentence (a **negative literal**). Negation Literal
- $\wedge$ (and). A sentence whose main connective is $\wedge$, such as $W_{1,3} \wedge P_{3,1}$, is called a **conjunction**; its parts are the **conjuncts**. (The $\wedge$ looks like an "A" for "And.") Conjunction
- $\vee$ (or). A sentence whose main connective is $\vee$, such as $(W_{1,3} \wedge P_{3,1}) \vee W_{2,2}$, is a **disjunction**; its parts are **disjuncts**—in this example, $(W_{1,3} \wedge P_{3,1})$ and $W_{2,2}$. Disjunction
- $\Rightarrow$ (implies). A sentence such as $(W_{1,3} \wedge P_{3,1}) \Rightarrow \neg W_{2,2}$ is called an **implication** (or conditional). Its **premise** or **antecedent** is $(W_{1,3} \wedge P_{3,1})$, and its **conclusion** or **consequent** is $\neg W_{2,2}$. Implications are also known as **rules** or **if–then** statements. The implication symbol is sometimes written in other books as $\supset$ or $\rightarrow$. Implication Premise Conclusion Rules
- $\Leftrightarrow$ (if and only if). The sentence $W_{1,3} \Leftrightarrow \neg W_{2,2}$ is a **biconditional**. Biconditional

CHAPTER 7

LOGICAL AGEN

In which we design agents that can f
of inference to derive new representa
tions to deduce what to do.

Knowledge-based agents
Reasoning
Representation

Humans, it seems, know things; and w
based agents use a process of **reaso**
decide what actions to take.
The problem-solving agents of Ch
inflexible sense. They know what act
specific action from a specific state wi
agent doesn't know that it is impossibl
An 8-puzzle agent doesn't know that t
they have is very useful for finding a p
The atomic representations used
a partially observable environment, fo
representing what it knows about the c
give a human the goal of driving to a
that to a problem-solving agent, I cou
the 16,000 or so towns that satisfy the
Chapter 6 introduced our first fac
assignments of values to variables; thi
the agent to work in a domain-indepe
In this chapter, we take this step to its
general class of representations to sup
bine and recombine information to su
needs of the moment—as when a ma
lates the Earth's life expectancy. Kno
of explicitly described goals; they can
new knowledge about the environmen
updating the relevant knowledge.
We begin in Section 7.1 with the
new environment, the wumpus world
agent without going into any technical
in Section 7.3 and the specifics of **pr**
a factored representation; while less c
the canonical structured representatio

KB is true in the real world? (After all, *KB* is
philosophical question about which many, ma
A simple answer is that the agent's sensors cr
world agent has a smell sensor. The agent pro
is a smell. Then, whenever that sentence is in
Thus, the meaning and truth of percept senten
sentence construction that produce them. Wh
as its belief that wumpuses cause smells in a
tation of a single percept, but a general rule-
but not identical to a statement of that exper
a sentence construction process called **learni**
fallible. It could be the case that wumpuses ca
which is when they take their baths. Thus, a
good learning procedures, there is reason for

7.4 Propositional Logic: A Ve

We now present **propositional logic**. We des
its semantics (the way in which the truth of s
a simple, syntactic algorithm for logical infe
entailment. Everything takes place, of course

7.4.1 Syntax

The **syntax** of propositional logic defines th
consist of a single **proposition symbol**. Eac
be true or false. We use symbols that start w
letters or subscripts, for example: P, Q, R,
but are often chosen to have some mnemonic
that the wumpus is in [1,3]. (Remember tha
and 3 are not meaningful parts of the symb
fixed meanings: *True* is the always-true propo
Complex sentences are constructed from sim
called **logical connectives**. There are five con

- $\neg$ (not). A sentence such as $\neg W_{1,3}$ is call
atomic sentence (a **positive literal**) or a
- $\wedge$ (and). A sentence whose main connecti
tion; its parts are the **conjuncts**. (The $\wedge$
- $\vee$ (or). A sentence whose main connective
tion; its parts are **disjuncts**—in this exa
- $\Rightarrow$ (implies). A sentence such as $(W_{1,3} \wedge P$
ditional). Its **premise** or **antecedent** is
is $\neg W_{2,2}$. Implications are also known a
symbol is sometimes written in other bo
- $\Leftrightarrow$ (if and only if). The sentence $W_{1,3} \Leftrightarrow$

$$
\begin{aligned}
\textit{Sentence} &\rightarrow \textit{AtomicSentence} \mid \textit{ComplexSentence} \\
\textit{AtomicSentence} &\rightarrow \textit{True} \mid \textit{False} \mid P \mid Q \mid R \mid \ldots \\
\textit{ComplexSentence} &\rightarrow (\ \textit{Sentence}\ ) \\
&\quad \mid \ \neg\ \textit{Sentence} \\
&\quad \mid \ \textit{Sentence} \wedge \textit{Sentence} \\
&\quad \mid \ \textit{Sentence} \vee \textit{Sentence} \\
&\quad \mid \ \textit{Sentence} \Rightarrow \textit{Sentence} \\
&\quad \mid \ \textit{Sentence} \Leftrightarrow \textit{Sentence} \\
\textsc{Operator Precedence} &: \ \neg, \wedge, \vee, \Rightarrow, \Leftrightarrow
\end{aligned}
$$

Figure 7.7 A BNF (Backus–Naur Form) grammar of sentences in propositional logic, along with operator precedences, from highest to lowest.

Figure 7.7 gives a formal grammar of propositional logic. (BNF notation is explained on page 1030.) The BNF grammar is augmented with an operator precedence list to remove ambiguity when multiple operators are used. The "not" operator ($\neg$) has the highest precedence, which means that in the sentence $\neg A \wedge B$ the $\neg$ binds most tightly, giving us the equivalent of $(\neg A) \wedge B$ rather than $\neg(A \wedge B)$. (The notation for ordinary arithmetic is the same: $-2+4$ is 2, not –6.) When appropriate, we also use parentheses and square brackets to clarify the intended sentence structure and improve readability.

7.4.2 Semantics

Truth value

Having specified the syntax of propositional logic, we now specify its semantics. The semantics defines the rules for determining the truth of a sentence with respect to a particular model. In propositional logic, a model simply sets the **truth value**—*true* or *false*—for every proposition symbol. For example, if the sentences in the knowledge base make use of the proposition symbols $P_{1,2}$, $P_{2,2}$, and $P_{3,1}$, then one possible model is

$$m_1 = \{P_{1,2} = \textit{false}, P_{2,2} = \textit{false}, P_{3,1} = \textit{true}\}.$$

With three proposition symbols, there are $2^3 = 8$ possible models—exactly those depicted in Figure 7.5. Notice, however, that the models are purely mathematical objects with no necessary connection to wumpus worlds. $P_{1,2}$ is just a symbol; it might mean "there is a pit in [1,2]" or "I'm in Paris today and tomorrow."

The semantics for propositional logic must specify how to compute the truth value of *any* sentence, given a model. This is done recursively. All sentences are constructed from atomic sentences and the five connectives; therefore, we need to specify how to compute the truth of atomic sentences and how to compute the truth of sentences formed with each of the five connectives. Atomic sentences are easy:

- *True* is true in every model and *False* is false in every model.
- The truth value of every other proposition symbol must be specified directly in the model. For example, in the model m_1 given earlier, $P_{1,2}$ is false.

Zoombinis! …

Palette for Prop-Calc Binary Vectors

A Prop-Calc Binary Vector, Built

The Formal Language of the Propositional Calculus

Syntax	Formula Type	Sample Representation
$P, P_1, P_2, Q, Q_1, \ldots$	Atomic Formulas	“Larry is lucky.” as L_l
$\neg\phi$	Negation	“Gary isn't lucky.” as $\neg L_g$
$\phi_1 \wedge \ldots \wedge \phi_n$	Conjunction	“Both Larry and Carl are lucky.” as $L_l \wedge L_c$
$\phi_1 \vee \ldots \vee \phi_n$	Disjunction	“Either Billy is lucky or Alvin is.” as $L_b \vee L_a$
$\phi \rightarrow \psi$	Conditional (Implication)	“If Ron is lucky, so is Frank.” as $L_r \rightarrow L_f$
$\phi \leftrightarrow \psi$	Biconditional (Coimplication)	“Tim is lucky if and only if Kim is.” as $L_t \longleftrightarrow L_k$

Table 2.1: Syntax of the Propositional Calculus. Note that ϕ, ψ, and ϕ_i stand for arbitrary formulas.

The Formal Language
(presented as formal grammar)

$$
\begin{array}{lcl}
\textit{Formula} & \Rightarrow & \textit{AtomicFormula} \\
& | & (\textit{Formula Connective Formula}) \\
& | & \neg\ \textit{Formula}
\end{array}
$$

$$\textit{AtomicFormula} \Rightarrow \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots$$

$$\textit{Connective} \Rightarrow \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow$$

As S-expressions

$$
\begin{array}{lcl}
\textit{Formula} & \Rightarrow & \textit{AtomicFormula} \\
 & | & (\textit{Formula}\ \ \textit{Connective}\ \ \textit{Formula}) \\
 & | & \neg\ \textit{Formula} \\
\hline
\textit{AtomicFormula} & \Rightarrow & \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ \ldots \\
\hline
\textit{Connective} & \Rightarrow & \wedge \mid \ \vee \ \mid \ \rightarrow \ \mid \ \leftrightarrow
\end{array}
$$

As S-expressions

$$\begin{array}{lcl} \textit{Formula} & \Rightarrow & \textit{AtomicFormula} \\ & \mid & (\textit{Formula Connective Formula}) \\ & \mid & \neg\ \textit{Formula} \end{array}$$

$$\textit{AtomicFormula} \Rightarrow \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots$$

$$\textit{Connective} \Rightarrow \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow$$

As S-expressions

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & \mid & (\mathit{Formula}\ \ \mathit{Connective}\ \ \mathit{Formula}) \\
 & \mid & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ \ldots \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \ \vee \ \mid \ \rightarrow \ \mid \ \leftrightarrow
\end{array}
$$

p

As S-expressions

$$
\begin{array}{lcl}
\textit{Formula} & \Rightarrow & \textit{AtomicFormula} \\
 & \mid & (\textit{Formula Connective Formula}) \\
 & \mid & \neg\ \textit{Formula} \\
\hline
\textit{AtomicFormula} & \Rightarrow & \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots \\
\hline
\textit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow
\end{array}
$$

p P

As S-expressions

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \ \mathit{Connective}\ \ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow
\end{array}
$$

`p`   `P`   `bradywillbeback`

As S-expressions

$$
\begin{array}{lcl}
\textit{Formula} & \Rightarrow & \textit{AtomicFormula} \\
 & | & (\textit{Formula}\ \textit{Connective}\ \textit{Formula}) \\
 & | & \neg\ \textit{Formula} \\
\hline
\textit{AtomicFormula} & \Rightarrow & \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots \\
\hline
\textit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow
\end{array}
$$

```
p   P   bradywillbeback  P26
```

As S-expressions

$$
\begin{array}{lcl}
\textit{Formula} & \Rightarrow & \textit{AtomicFormula} \\
 & | & (\textit{Formula}\ \ \textit{Connective}\ \ \textit{Formula}) \\
 & | & \neg\ \textit{Formula} \\
\hline
\textit{AtomicFormula} & \Rightarrow & \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots \\
\hline
\textit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow
\end{array}
$$

`p   P   bradywillbeback   P26   ...`

As S-expressions

$\mathit{Formula}$	$\Rightarrow$	$\mathit{AtomicFormula}$
	$\mid$	$(\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula})$
	$\mid$	$\neg\ \mathit{Formula}$
$\mathit{AtomicFormula}$	$\Rightarrow$	$\mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots$
$\mathit{Connective}$	$\Rightarrow$	$\wedge \mid \vee \mid \rightarrow \mid \leftrightarrow$

`p   P   bradywillbeback  P26   ...`

As S-expressions

Formula	$\Rightarrow$	*AtomicFormula*
	$\mid$	(*Formula Connective Formula*)
	$\mid$	$\neg$ *Formula*
AtomicFormula	$\Rightarrow$	$\mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots$
Connective	$\Rightarrow$	$\wedge \mid \vee \mid \rightarrow \mid \leftrightarrow$

`p   P   bradywillbeback   P26   ...`

`(not p)`

As S-expressions

$$
\begin{aligned}
\textit{Formula} \quad &\Rightarrow \quad \textit{AtomicFormula} \\
&\mid \quad (\textit{Formula}\ \ \textit{Connective}\ \ \textit{Formula}) \\
&\mid \quad \neg\ \textit{Formula}
\end{aligned}
$$

$$\textit{AtomicFormula} \quad \Rightarrow \quad \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots$$

$$\textit{Connective} \quad \Rightarrow \quad \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow$$

```
p   P   bradywillbeback  P26   ...
```

```
(not p)  (not P)
```

As S-expressions

$$
\begin{array}{lcl}
\textit{Formula} & \Rightarrow & \textit{AtomicFormula} \\
 & | & (\textit{Formula}\ \textit{Connective}\ \textit{Formula}) \\
 & | & \neg\ \textit{Formula} \\
\hline
\textit{AtomicFormula} & \Rightarrow & \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots \\
\hline
\textit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow
\end{array}
$$

```
p   P   bradywillbeback   P26   ...

(not p)   (not P)   (not P26)
```

As S-expressions

$$
\begin{array}{lcl}
\textit{Formula} & \Rightarrow & \textit{AtomicFormula} \\
 & | & (\textit{Formula Connective Formula}) \\
 & | & \neg\ \textit{Formula} \\
\hline
\textit{AtomicFormula} & \Rightarrow & \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots \\
\hline
\textit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow
\end{array}
$$

```
p   P   bradywillbeback  P26   ...

(not p)  (not P)  (not P26)    ...
```

As S-expressions

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula}
\end{array}
$$

$$\mathit{AtomicFormula} \Rightarrow \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots$$

$$\mathit{Connective} \Rightarrow \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow$$

```
p   P   bradywillbeback  P26   ...

(not p)  (not P)  (not P26)    ...
```

As S-expressions

$$
\begin{aligned}
\mathit{Formula} \quad &\Rightarrow \quad \mathit{AtomicFormula} \\
&\mid \quad (\mathit{Formula}\ \ \mathit{Connective}\ \ \mathit{Formula}) \\
&\mid \quad \neg\ \mathit{Formula}
\end{aligned}
$$

$$\mathit{AtomicFormula} \quad \Rightarrow \quad \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots$$

$$\mathit{Connective} \quad \Rightarrow \quad \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow$$

```
p   P   bradywillbeback  P26   ...

(not p)  (not P)  (not P26)    ...

(and P Q)
```

As S-expressions

$$
\begin{array}{lcl}
\textit{Formula} & \Rightarrow & \textit{AtomicFormula} \\
 & | & (\textit{Formula}\ \ \textit{Connective}\ \ \textit{Formula}) \\
 & | & \neg\ \textit{Formula} \\
\hline
\textit{AtomicFormula} & \Rightarrow & \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots \\
\hline
\textit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow
\end{array}
$$

`p   P   bradywillbeback  P26   ...`

`(not p)  (not P)  (not P26)   ...`

`(and P Q)  (or P Q)`

As S-expressions

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \ \mathit{Connective}\ \ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & \mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow
\end{array}
$$

```
p   P   bradywillbeback  P26   ...

(not p)  (not P)  (not P26)    ...

(and P Q)  (or P Q) (if P Q)
```

As S-expressions

Formula	$\Rightarrow$	*AtomicFormula*
	$\mid$	(*Formula Connective Formula*)
	$\mid$	$\neg$ *Formula*
AtomicFormula	$\Rightarrow$	$\mathtt{P}_1 \mid \mathtt{P}_2 \mid \mathtt{P}_3 \mid \ldots$
Connective	$\Rightarrow$	$\wedge \mid \vee \mid \rightarrow \mid \leftrightarrow$

p P bradywillbeback P26 ...

(not p) (not P) (not P26) ...

(and P Q) (or P Q) (if P Q) (iff P Q)

More *Expressive* Formal Language: Pure Predicate Calculus …

$$\mathscr{L}_0$$

Better Formal Language: Pure Predicate Calculus (presented via formal grammar)

$$
\begin{array}{lcl}
\textit{Formula} & \Rightarrow & \textit{AtomicFormula} \\
 & | & (\textit{Formula}\ \ \textit{Connective}\ \ \textit{Formula}) \\
 & | & \neg\ \textit{Formula}
\end{array}
$$

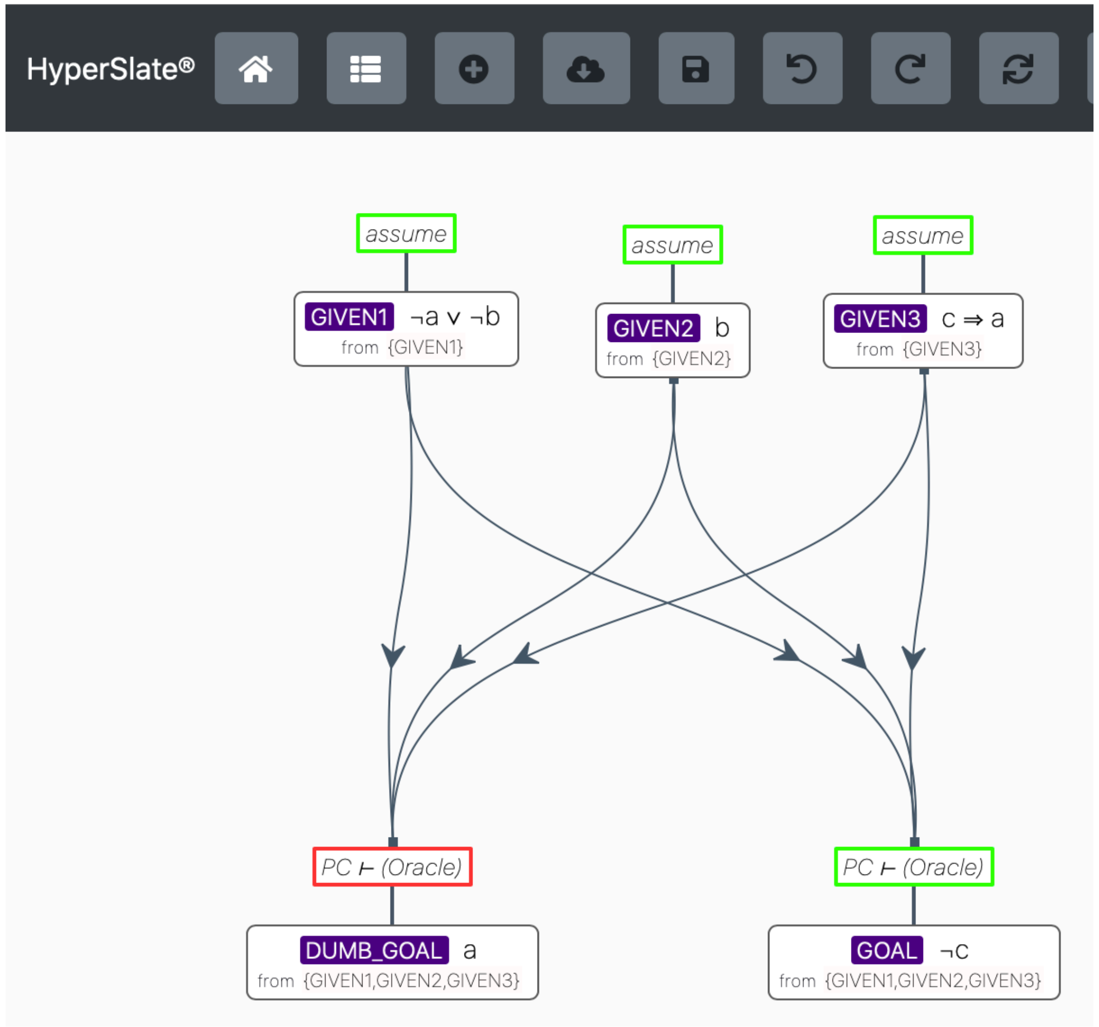
$$
\textit{AtomicFormula} \Rightarrow (\textit{Predicate}\ \ \textit{Term}_1 \ldots \textit{Term}_k)
$$

$$
\begin{array}{lcl}
\textit{Term} & \Rightarrow & (\textit{Function}\ \ \textit{Term}_1\ \ \ldots\ \ \textit{Term}_k) \\
 & | & \textit{Constant} \\
 & | & \textit{Variable}
\end{array}
$$

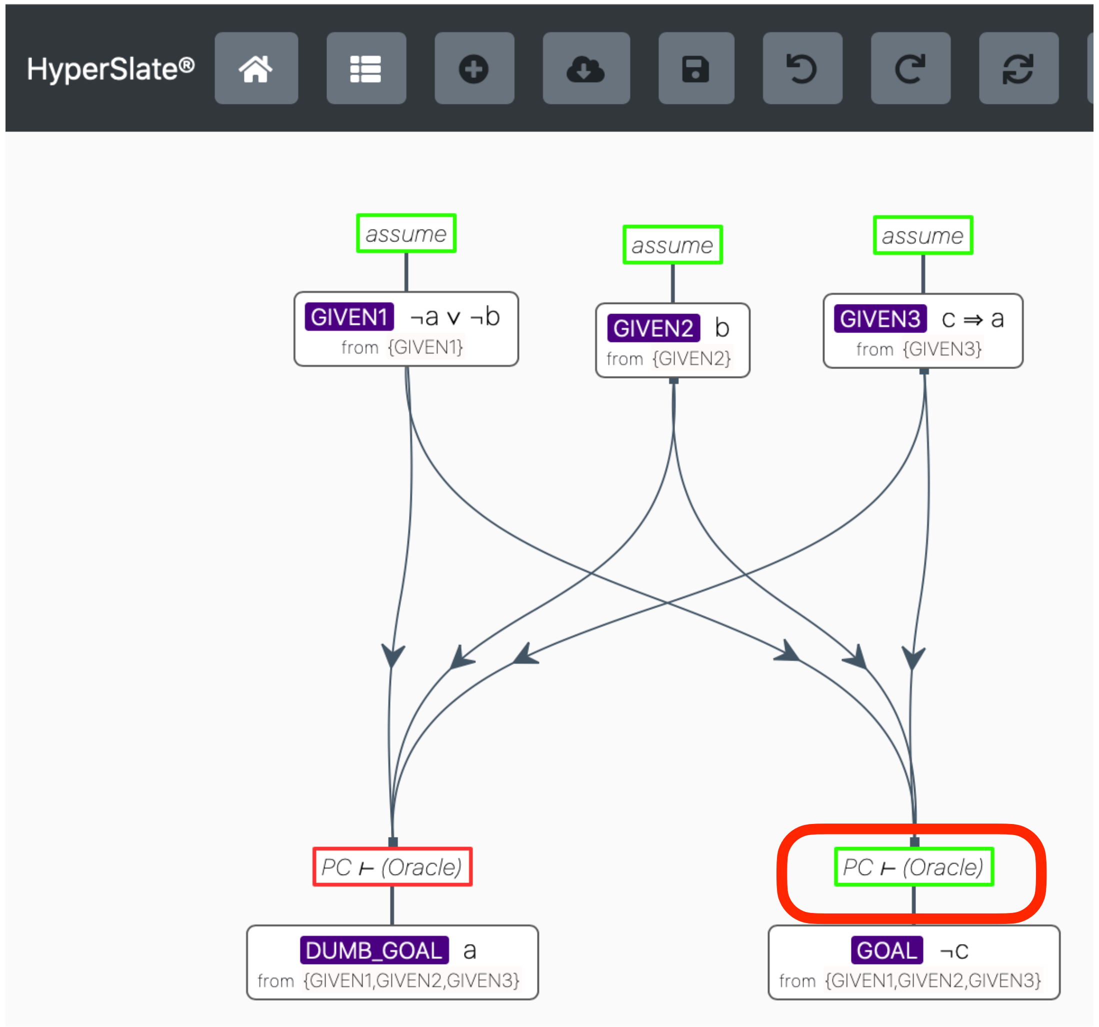
$$
\textit{Connective} \Rightarrow \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow
$$

$$
\begin{array}{lcl}
\textit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\textit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\textit{Variable} & \Rightarrow & v_1 \mid v_2 \mid v_3\ \ldots \\
\textit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

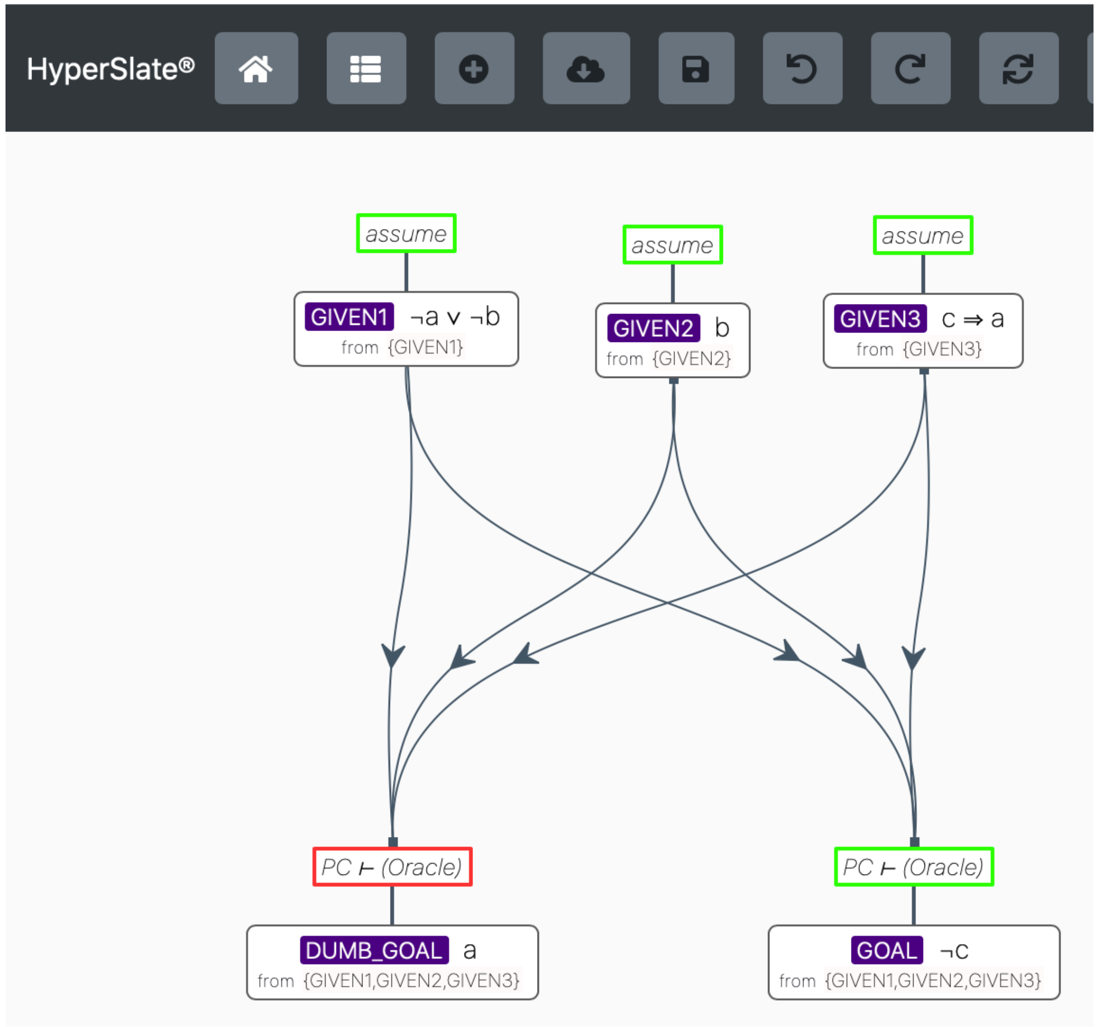
The PC (Provability) Oracle



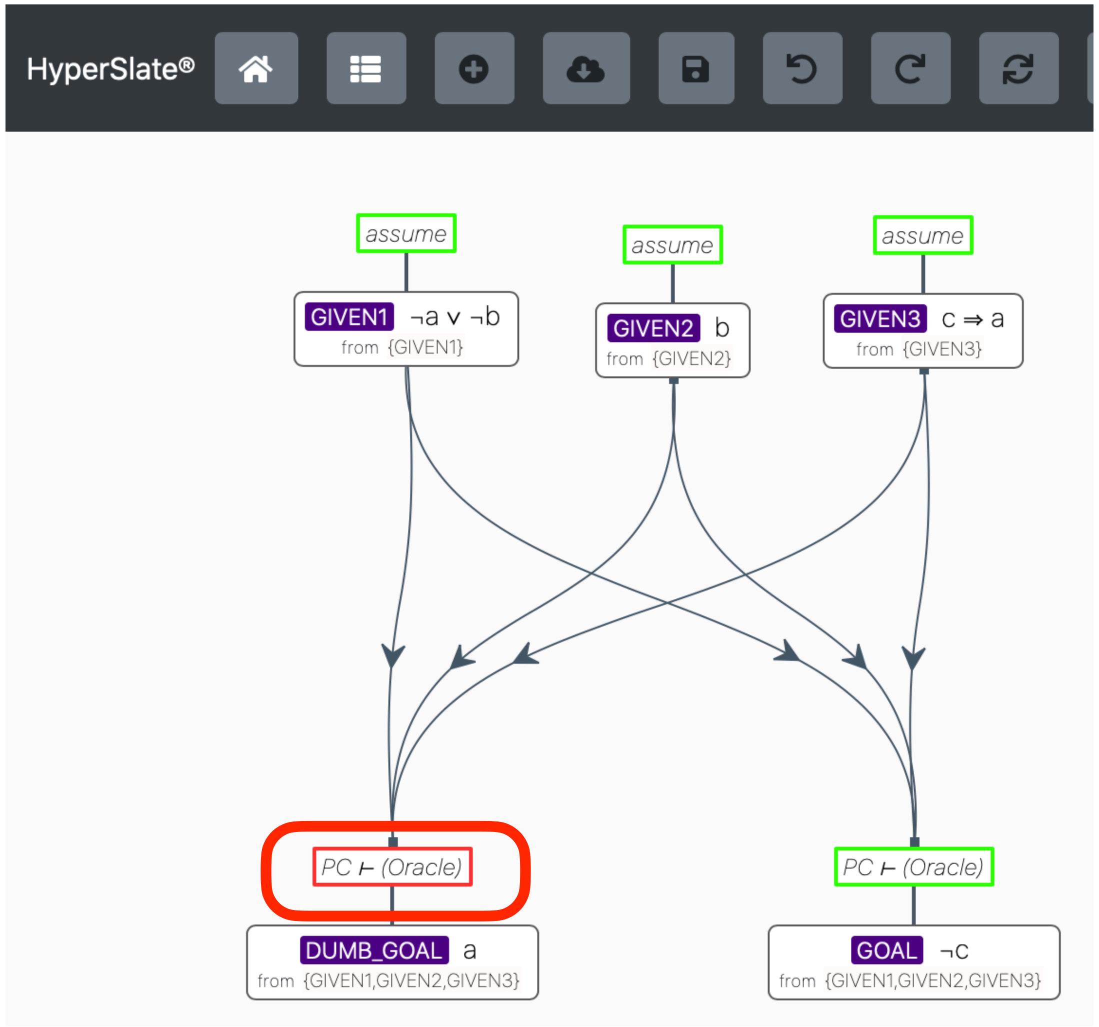
The PC (Provability) Oracle



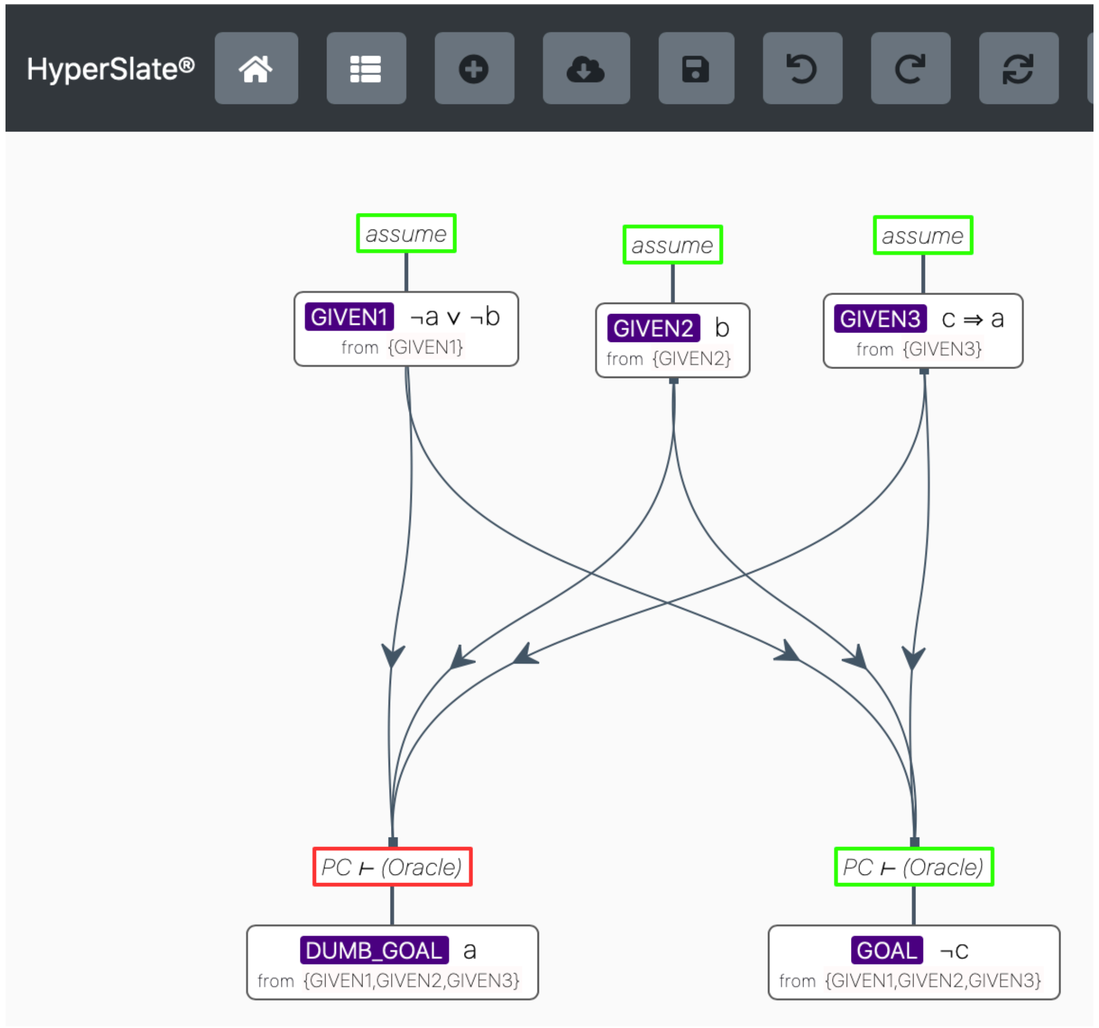
The PC (Provability) Oracle



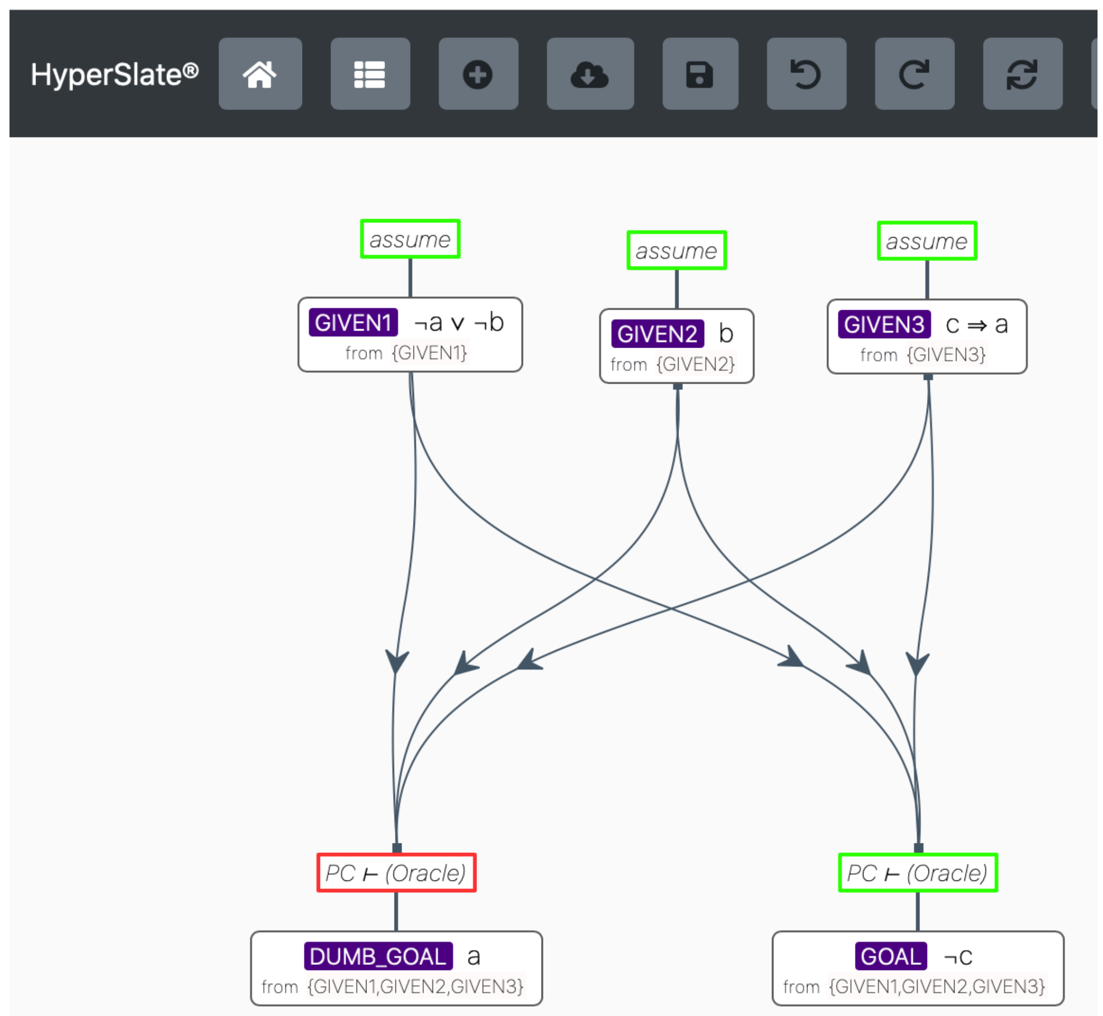
The PC (Provability) Oracle



The PC (Provability) Oracle



The PC (Provability) Oracle



The Infamous "NYS 3"

Given the statements

$\neg\neg c$

$c \rightarrow a$

$\neg a \vee b$

$b \rightarrow d$

$\neg(d \vee e)$

which one of the following statements are provable?

$\neg c$

e

h

$\neg a$

all of the above

The Infamous "NYS 3"

Given the statements

$\neg\neg c$

$c \rightarrow a$

$\neg a \vee b$

$b \rightarrow d$

$\neg(d \vee e)$

which one of the following statements are provable?

$\neg c$

e

h

$\neg a$

all of the above

The Infamous "NYS 3"

Given the statements

$\neg\neg c$

$c \rightarrow a$

$\neg a \vee b$

$b \rightarrow d$

$\neg(d \vee e)$

Show in HyperSlate® that each of the first four options can be proved using the PC provability oracle.

which one of the following statements are provable?

$\neg c$

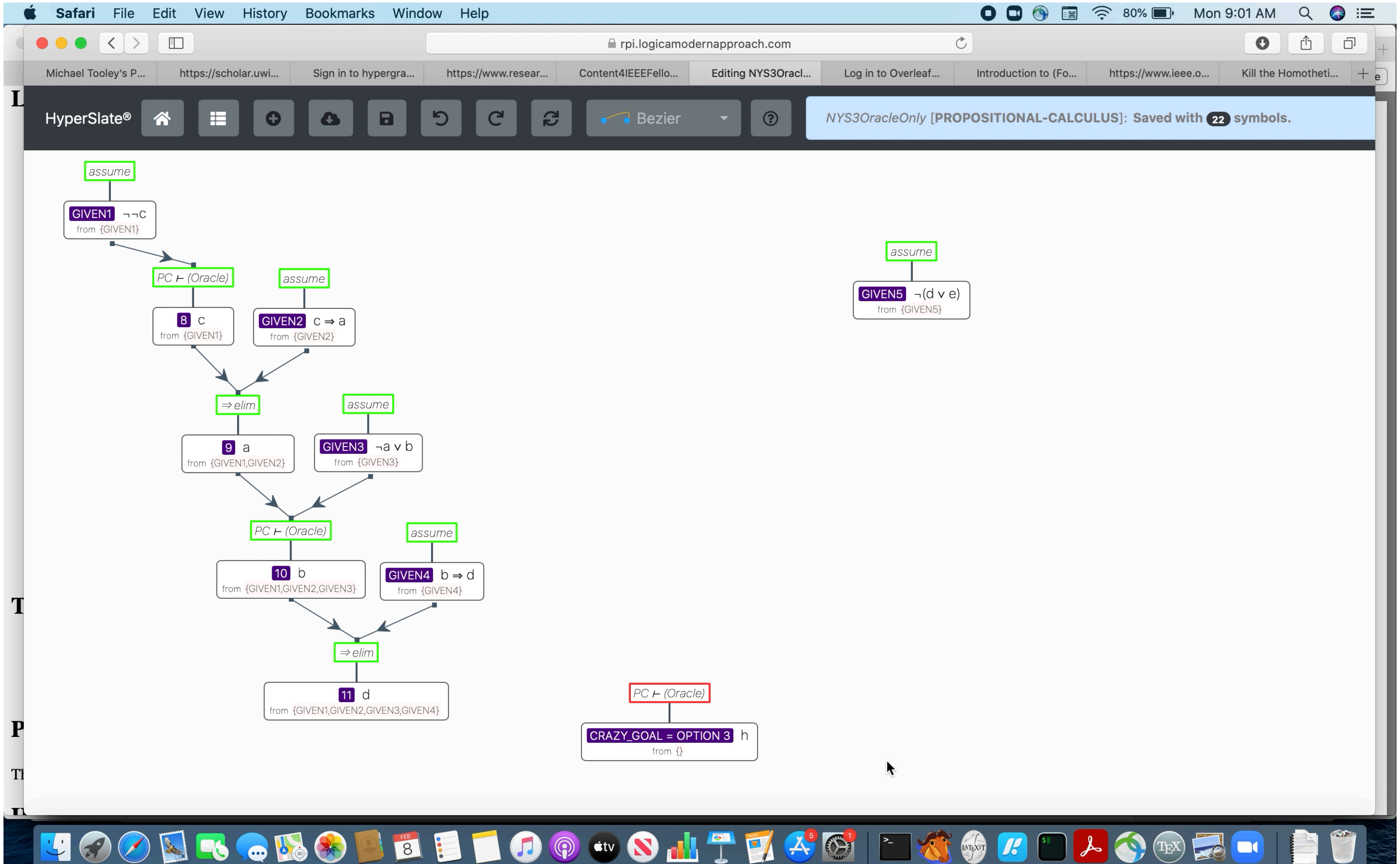
e

h

$\neg a$

all of the above

The Infamous “NYS 3”



all of the above

Logistics for Registration etc …

The Starting Code to Purchase in Bookstore

Your code for starting the registration process is:

To access HyperGrader, HyperSlate, the license agreement, and to obtain the textbook LAMA-BDLA, go to::

https://rpi.logicamodernapproach.com

The Starting Code to Purchase in Bookstore

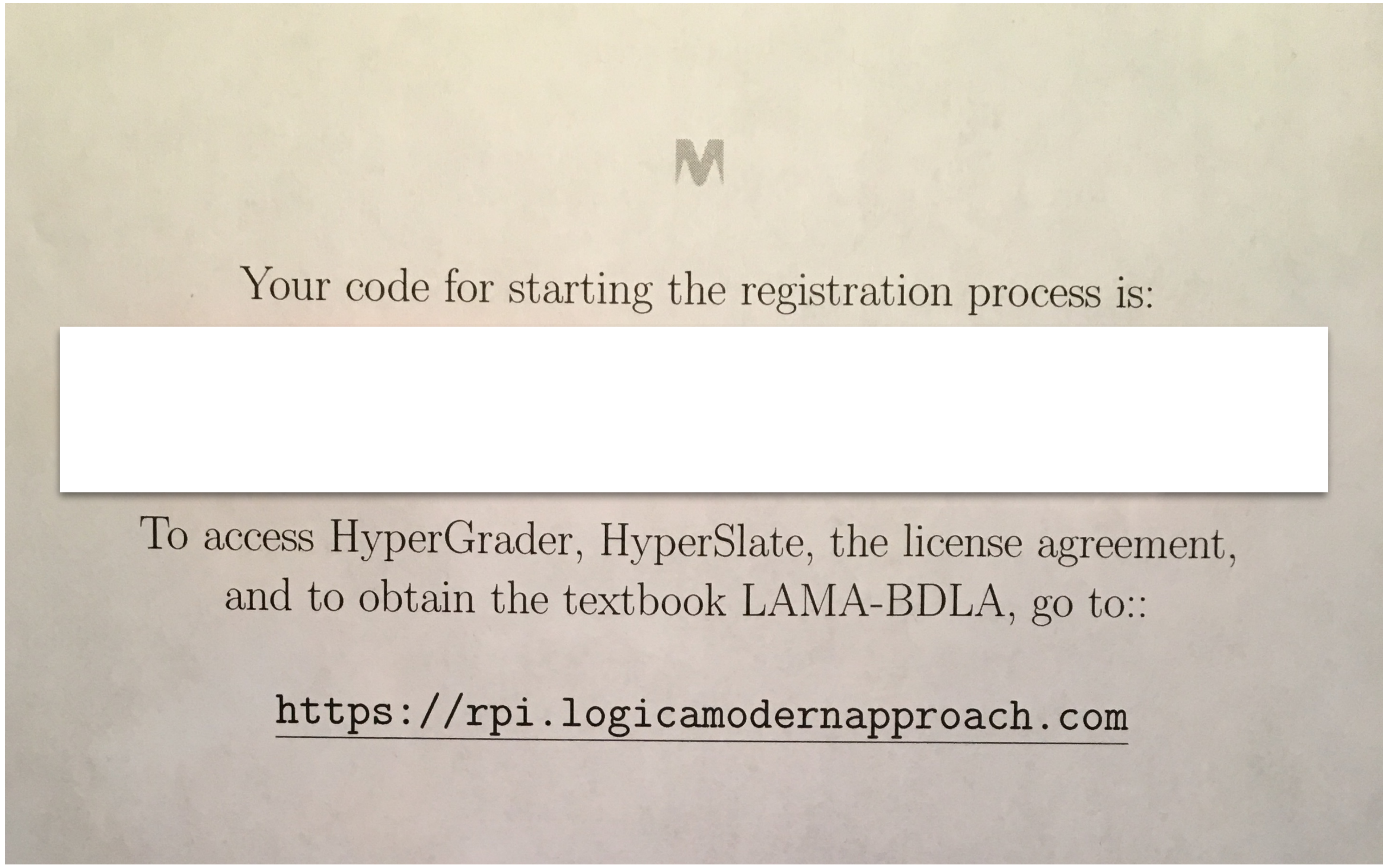
Your code for starting the registration process is:

To access HyperGrader, HyperSlate, the license agreement, and to obtain the textbook LAMA-BDLA, go to::

https://rpi.logicamodernapproach.com

Once seal broken on envelope, no return. Remember from first class, any reservations, opt out of ILBAI!

The Starting Code to Purchase in Bookstore


Your code for starting the registration process is:

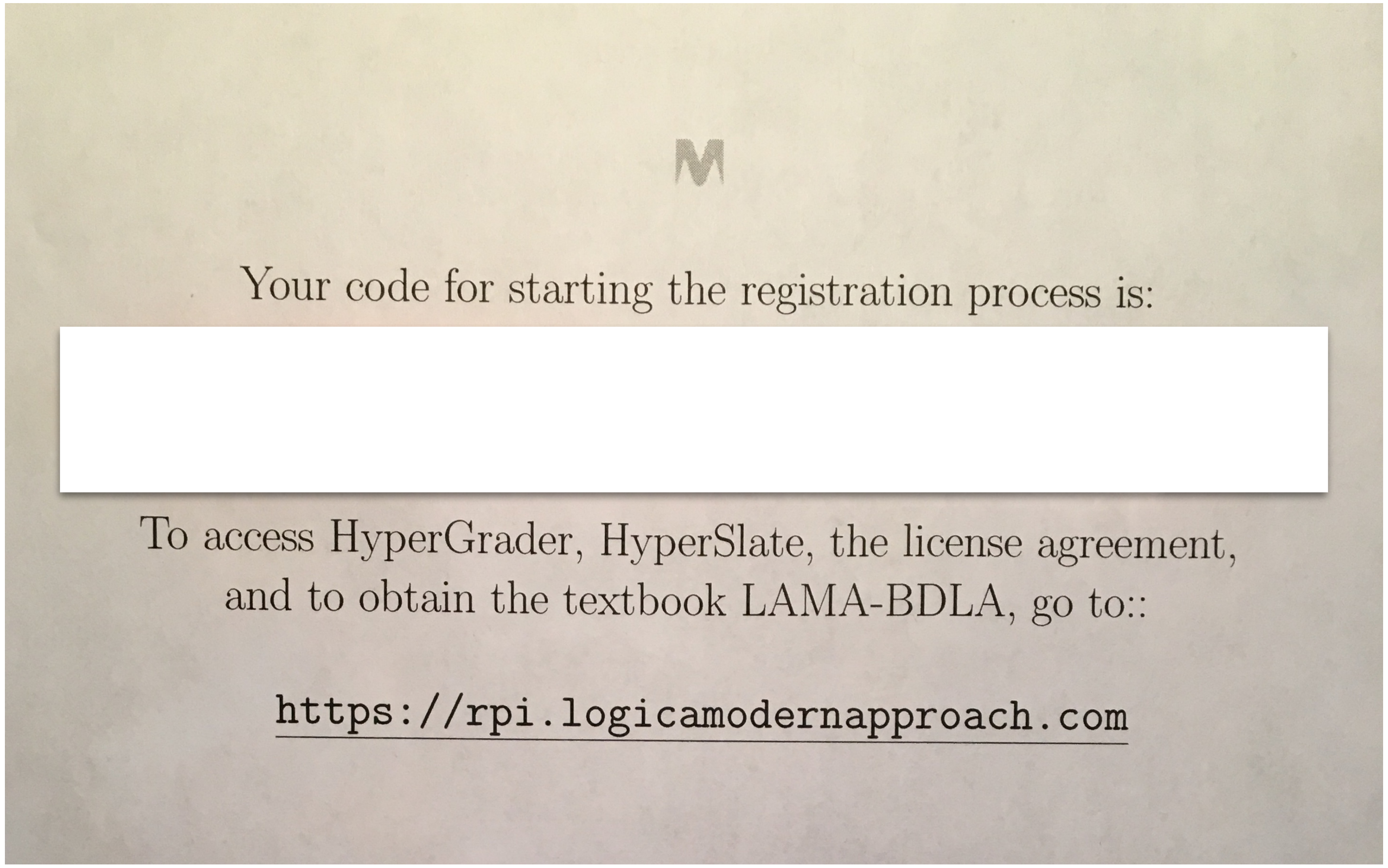
To access HyperGrader, HyperSlate, the license agreement, and to obtain the textbook LAMA-BDLA, go to::

https://rpi.logicamodernapproach.com

Once seal broken on envelope, no return. Remember from first class, any reservations, opt out of ILBAI!

The email address you enter is case-sensitive!

The Starting Code to Purchase in Bookstore


Your code for starting the registration process is:

To access HyperGrader, HyperSlate, the license agreement, and to obtain the textbook LAMA-BDLA, go to::

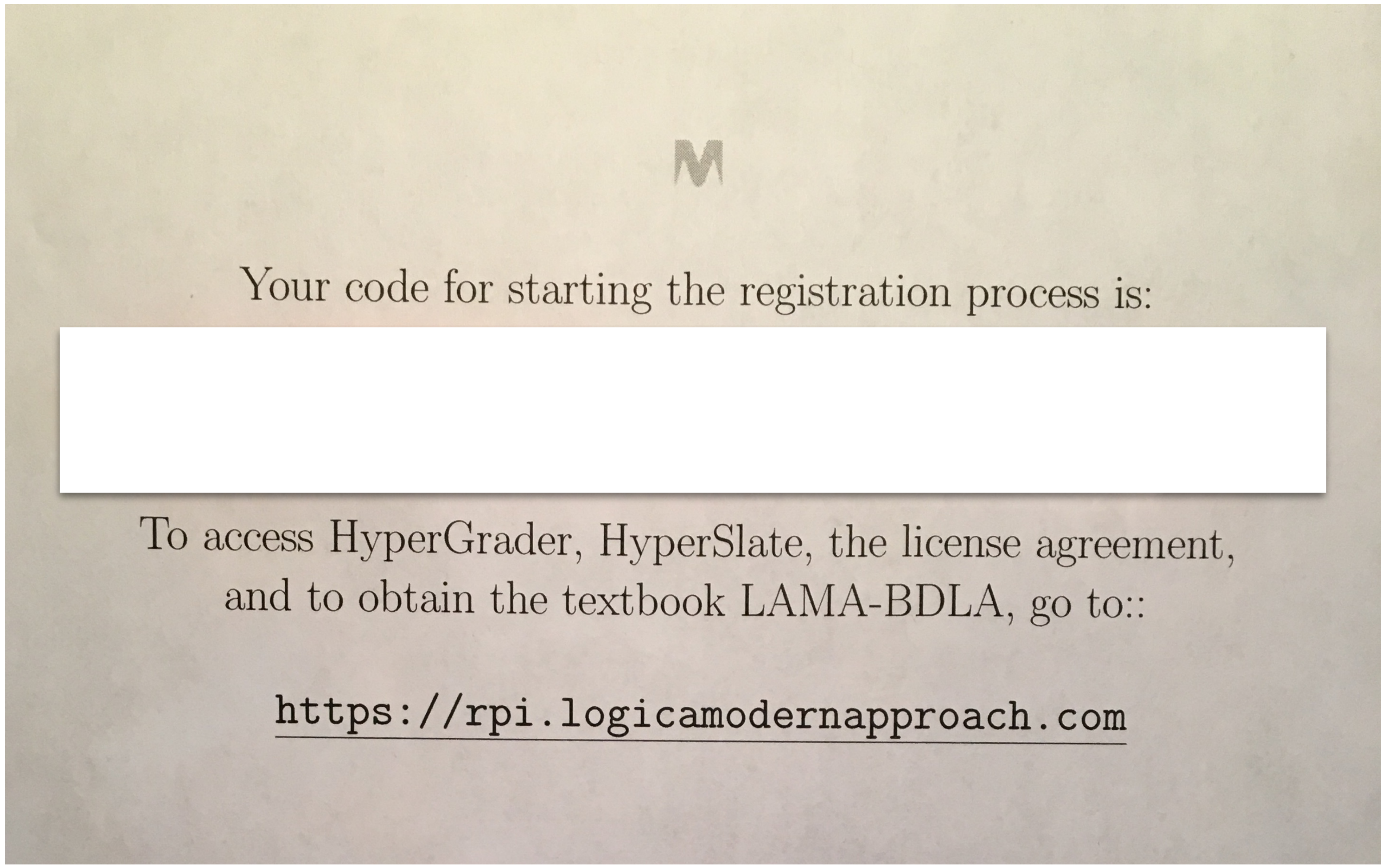
https://rpi.logicamodernapproach.com

Once seal broken on envelope, no return. Remember from first class, any reservations, opt out of ILBAI!

The email address you enter is case-sensitive!

Your OS and browser must be fully up-to-date; Chrome is the best choice, browser-wise (though I use Safari :)).

The Starting Code to Purchase in Bookstore



Your code for starting the registration process is:

To access HyperGrader, HyperSlate, the license agreement, and to obtain the textbook LAMA-BDLA, go to::

https://rpi.logicamodernapproach.com

Once seal broken on envelope, no return. Remember from first class, any reservations, opt out of ILBAI!

The email address you enter is case-sensitive!

Your OS and browser must be fully up-to-date; Chrome is the best choice, browser-wise (though I use Safari :)).

Watch that the link emailed to you doesn't end up being classified as spam.

Now for live tutorial etc on this …

Kunstig intelligens bør være logisk!