Rigorously & “Brainishly” Speaking, What are We?

Bringsjord v. Granger

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 (ILBAI)
11/11/2024

version Nov 11 2024 for ILBAI

Topics Before Granger Madness

Topics Before Granger Madness

- Wrap-up of AI & Idolatry.

Topics Before Granger Madness

- Wrap-up of AI & Idolatry.
- DFT+ via `o1 preview`.

Topics Before Granger Madness

- Wrap-up of AI & Idolatry.
- DFT+ via `o1 preview`.
- Try a new `Live Logic` problem?

Topics Before Granger Madness

- Wrap-up of AI & Idolatry.
- DFT+ via `o1 preview`.
- Try a new `Live Logic` problem?
- Back to the ladder & SOL in HyperSlate®.

Topics Before Granger Madness

- Wrap-up of AI & Idolatry.
- DFT+ via `o1 preview`.
- Try a new `Live Logic` problem?
- Back to the ladder & SOL in HyperSlate®.
- `DoubleMindedMan` explored/solved.

Topics Before Granger Madness

- Wrap-up of AI & Idolatry.
- DFT+ via `o1 preview`.
- Try a new `Live Logic` problem?
- Back to the ladder & SOL in HyperSlate®.
- `DoubleMindedMan` explored/solved.
- Can we simulate `o1 preview` on **S**OL problems?

Topics Before Granger Madness

- Wrap-up of AI & Idolatry.
- DFT+ via `o1 preview`.
- Try a new `Live Logic` problem?
- Back to the ladder & SOL in HyperSlate®.
- `DoubleMindedMan` explored/solved.
- Can we simulate `o1 preview` on **S**OL problems?
- Now to Bringsjord vs. Granger …

Exemplar in HG®

Exemplar in HG®

```
(forall (n) (= (func n 1) (inc 1)))
```

Exemplar in HG®

```
(forall (n) (= (func n 1) (inc 1)))
```

```
(forall (x) (= (func 1 (inc x)) (inc (inc (func 0 x)))))
```

Exemplar in HG®

```
(forall (n) (= (func n 1) (inc 1)))

(forall (x) (= (func 1 (inc x)) (inc (inc (func 0 x)))))

(forall (n x) (= (func (inc n) (inc x)) (func n (func (inc n) x))))
```

Exemplar in HG®

```
(forall (n) (= (func n 1) (inc 1)))

(forall (x) (= (func 1 (inc x)) (inc (inc (func 0 x)))))

(forall (n x) (= (func (inc n) (inc x) (func n (func (inc n) x))))

(NatNum 1)
```

Exemplar in HG®

```
(forall (n) (= (func n 1) (inc 1)))

(forall (x) (= (func 1 (inc x)) (inc (inc (func 0 x)))))

(forall (n x) (= (func (inc n) (inc x) (func n (func (inc n) x))))

(NatNum 1)

(forall (n) (if (NatNum n) (NatNum (inc n))))
```

Exemplar in HG®

```
(forall (n) (= (func n 1) (inc 1)))

(forall (x) (= (func 1 (inc x)) (inc (inc (func 0 x)))))

(forall (n x) (= (func (inc n) (inc x) (func n (func (inc n) x))))

(NatNum 1)

(forall (n) (if (NatNum n) (NatNum (inc n))))

(NatNum (func 5 5))?????
```

Possible Paper

(Review k-order ladder)

Possible Paper

(Review k-order ladder)

Does $\mathscr{L}_3 = \text{TOL}$ work in HyperSlate? Partially? Not at all? What's possible and what's not? What exactly is needed inference-rule-wise for a full natural-deduction system for TOL. Can a chatbot like GPT-4 o1 preview etc. handle TOL reasoning challenges expressed in English? What specimens do you have for your answer?

Some Roots of the Debate

Theoretical Computer Science 633 (2016) 100–111

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

The grammar of mammalian brain capacity

A. Rodriguez, R. Granger *

6207 Moore Hall, Dartmouth College, Hanover, NH 03755, United States

A R T I C L E I N F O

Article history:
Received 8 November 2015
Received in revised form 21 December 2015
Accepted 16 March 2016
Available online 21 March 2016

Keywords:
Brain allometry
Grammars
High-order pushdown automata
Thalamocortical circuits

A B S T R A C T

Uniquely human abilities may arise from special-purpose brain circuitry, or from concerted general capacity increases due to our outsized brains. We forward a novel hypothesis of the relation between computational capacity and brain size, linking mathematical formalisms of grammars with the allometric increases in cortical–subcortical ratios that arise in large brains. In sum, i) thalamocortical loops compute formal grammars; ii) successive cortical regions describe grammar rewrite rules of increasing size; iii) cortical–subcortical ratios determine the quantity of stacks in single-stack pushdown grammars; iv) quantitative increase of stacks yields grammars with qualitatively increased computational power. We arrive at the specific conjecture that human brain capacity is equivalent to that of indexed grammars – far short of full Turing-computable (recursively enumerable) systems. The work provides a candidate explanatory account of a range of existing human and animal data, addressing longstanding questions of how repeated similar brain algorithms can be successfully applied to apparently dissimilar computational tasks (e.g., perceptual versus cognitive, phonological versus syntactic); and how quantitative increases to brains can confer qualitative changes to their computational repertoire.

1. Brain growth shows surprisingly few signs of evolutionary pressure

Different animals exhibit different mental and behavioral abilities, but it is not known which abilities arise from specializations in the brain, i.e., circuitry to specifically support or enable particular capacities. Evolutionary constraints on brain construction severely narrow the search for candidate specializations. Although mammalian brain sizes span four orders of magnitude [1], the range of structural variation differentiating those brains is extraordinarily limited.

An animal's brain size can be roughly calculated from its body size [2], but much more telling is the relationship between the sizes of brains and of their constituent parts: the size of almost every component brain circuit can be computed with remarkable accuracy just from the overall size of that brain [1,3–5], and thus the ratios among brain parts (e.g. cortical to subcortical size ratios) increase in a strictly predictable allometric fashion as overall brain size increases [6,7] (Fig. 1).

These allometric regularities obtain even at the level of individual brain structures (e.g., hippocampus, basal ganglia, cortical areas). There are a few specific exceptions to the well-documented allometric rule (such as the primate olfactory system [8]), clearly demonstrating that at least some brain structure sizes *can* be differentially regulated in evolution, yet despite this capability, it is extremely rare for telencephalic structures ever to diverge from the allometric rule [4,6,7,9]. Area 10, the frontal pole, is the most disproportionately expanded structure in the human brain, and has sometimes been argued to be *selected* for differential expansion, yet the evidence has strongly indicated that area 10 (and the rest of anterior cortex) are nonetheless precisely the size that is predicted allometrically [6,7,10,11].

* Corresponding author.
E-mail address: Richard.Granger@gmail.com (R. Granger).

http://dx.doi.org/10.1016/j.tcs.2016.03.021
0304-3975/© 2016 Published by Elsevier B.V.

Available online at www.sciencedirect.com

SCIENCE DIRECT

Theoretical Computer Science 317 (2004) 167–190

Theoretical Computer Science

www.elsevier.com/locate/tcs

The modal argument for hypercomputing minds

Selmer Bringsjord*, Konstantine Arkoudas

Department of Computer Science, Department of Cognitive Science, Rensselaer AI & Reasoning Laboratory, Rensselaer Polytechnic Institute (RPI), Troy, NY 12180, USA

Received 14 July 2003; received in revised form 21 October 2003

Abstract

We now know both that hypercomputation (or super-recursive computation) is mathematically well-understood, and that it provides a theory that according to some accounts for some real-life computation (e.g., operating systems that, unlike Turing machines, never simply output an answer and halt) better than the standard theory of computation at and below the "Turing Limit." But one of the things we do not know is whether the human mind hypercomputes, or merely computes—this despite informal arguments from Gödel, Lucas, Penrose and others for the view that, in light of incompleteness theorems, the human mind has powers exceeding those of TMs and their equivalents. All these arguments fail; their fatal flaws have been repeatedly exposed in the literature. However, we give herein a novel, formal *modal* argument showing that since it's mathematically *possible* that human minds are hypercomputers, such minds *are* in fact hypercomputers. We take considerable pains to anticipate and rebut objections to this argument.

Keywords: Computationalism; Hypercomputation; Incompleteness theorems

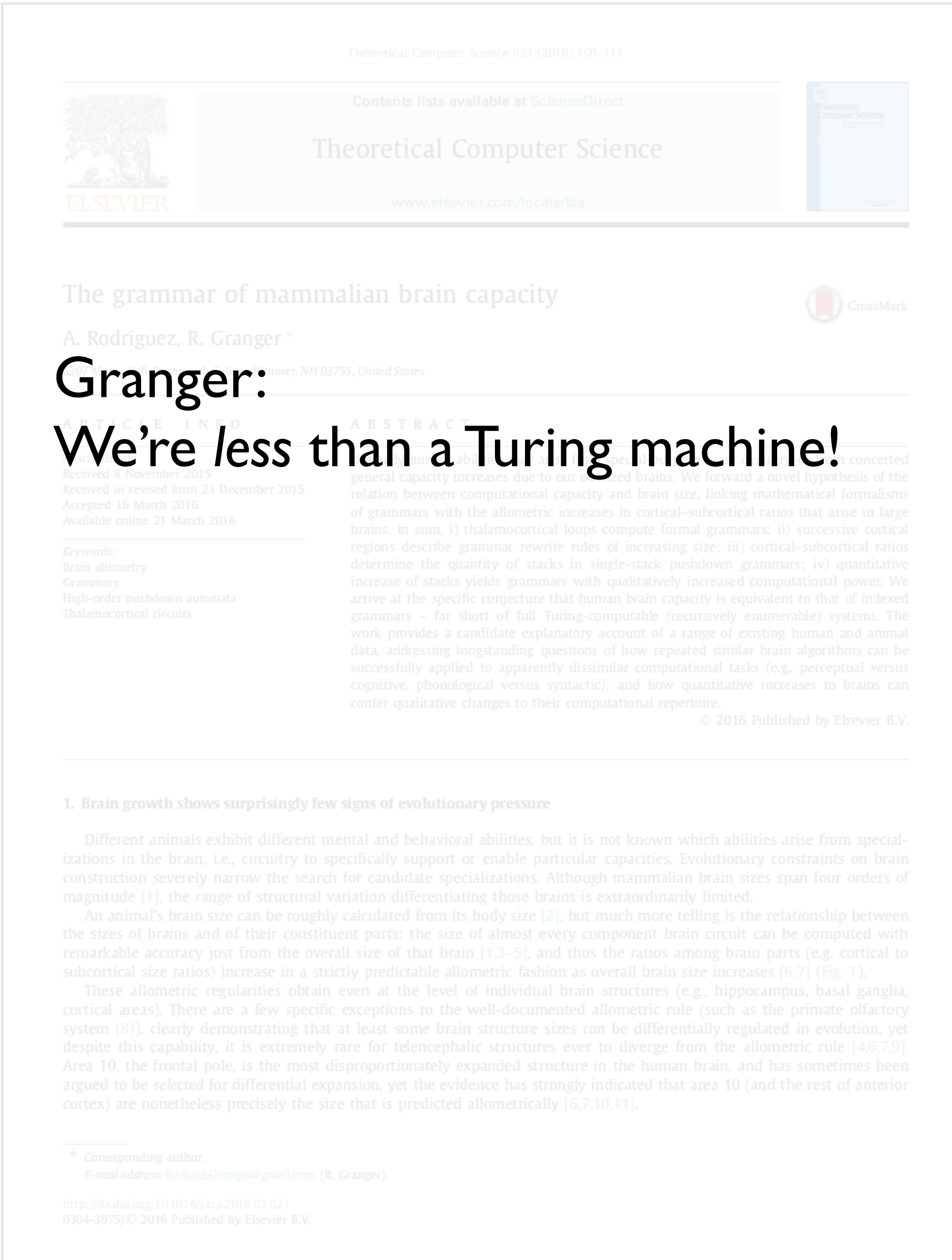
1. Introduction

Four decades ago, Lucas [50] expressed supreme confidence that Gödel's first incompleteness theorem (= Gödel I) entails the falsity of computationalism, the view that human persons are computing machines (e.g., Turing machines). Put barbarically, Lucas' basic idea is that minds are more powerful than Turing machines. Today, given our understanding of hypercomputation in theoretical computer science, and given the absolute consensus reigning in cognitive science that the human mind is, at least in large part, *some* sort of information-processing device, we know enough to infer that if Lucas is right, the mind is a hypercomputer. However, Lucas' arguments have

* Corresponding author.
E-mail addresses: selmer@rpi.edu (S. Bringsjord), koud@ai.mit.edu (K. Arkoudas).

0304-3975/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2003.12.010

Some Roots of the Debate



Theoretical Computer Science 631 (2016) 100–111

Contents lists available at ScienceDirect

Theoretical Computer Science

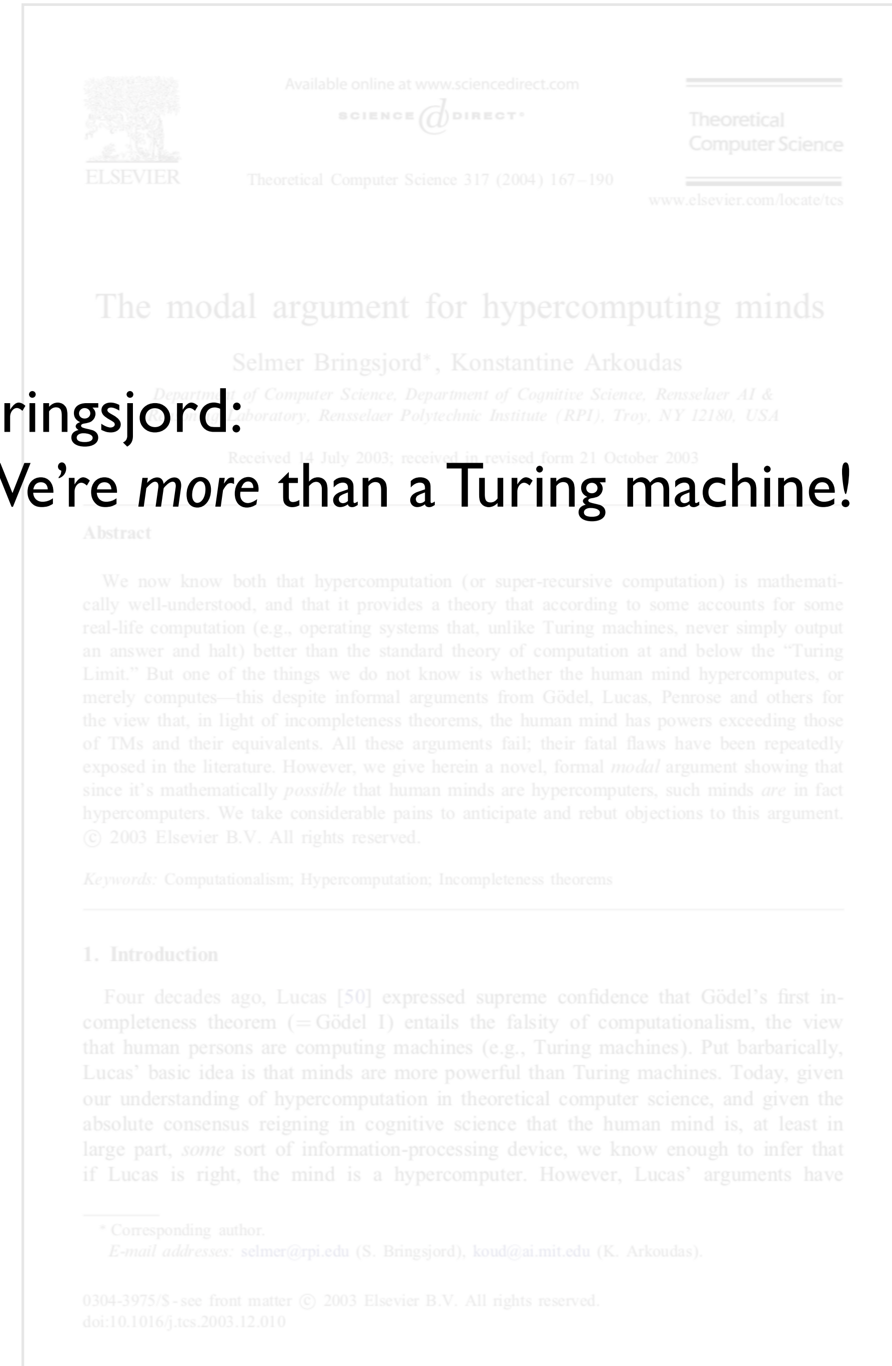
www.elsevier.com/locate/tcs

The grammar of mammalian brain capacity

A. Rodriguez, R. Granger

1. Brain growth shows surprisingly few signs of evolutionary pressure

Granger:
We're *less* than a Turing machine!



Available online at www.sciencedirect.com

Theoretical Computer Science 317 (2004) 167–190

Theoretical Computer Science

www.elsevier.com/locate/tcs

The modal argument for hypercomputing minds

Selmer Bringsjord, Konstantine Arkoudas

Received 11 July 2003; received in revised form 21 October 2003

Abstract

1. Introduction

Bringsjord:
We're *more* than a Turing machine!

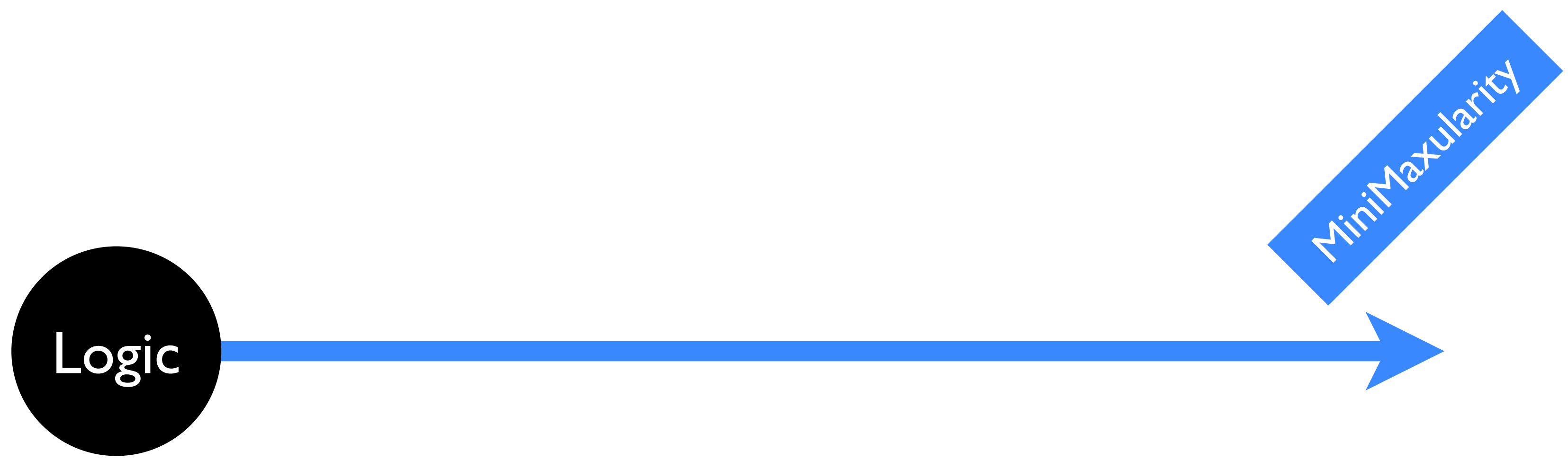
Fatal Objection?

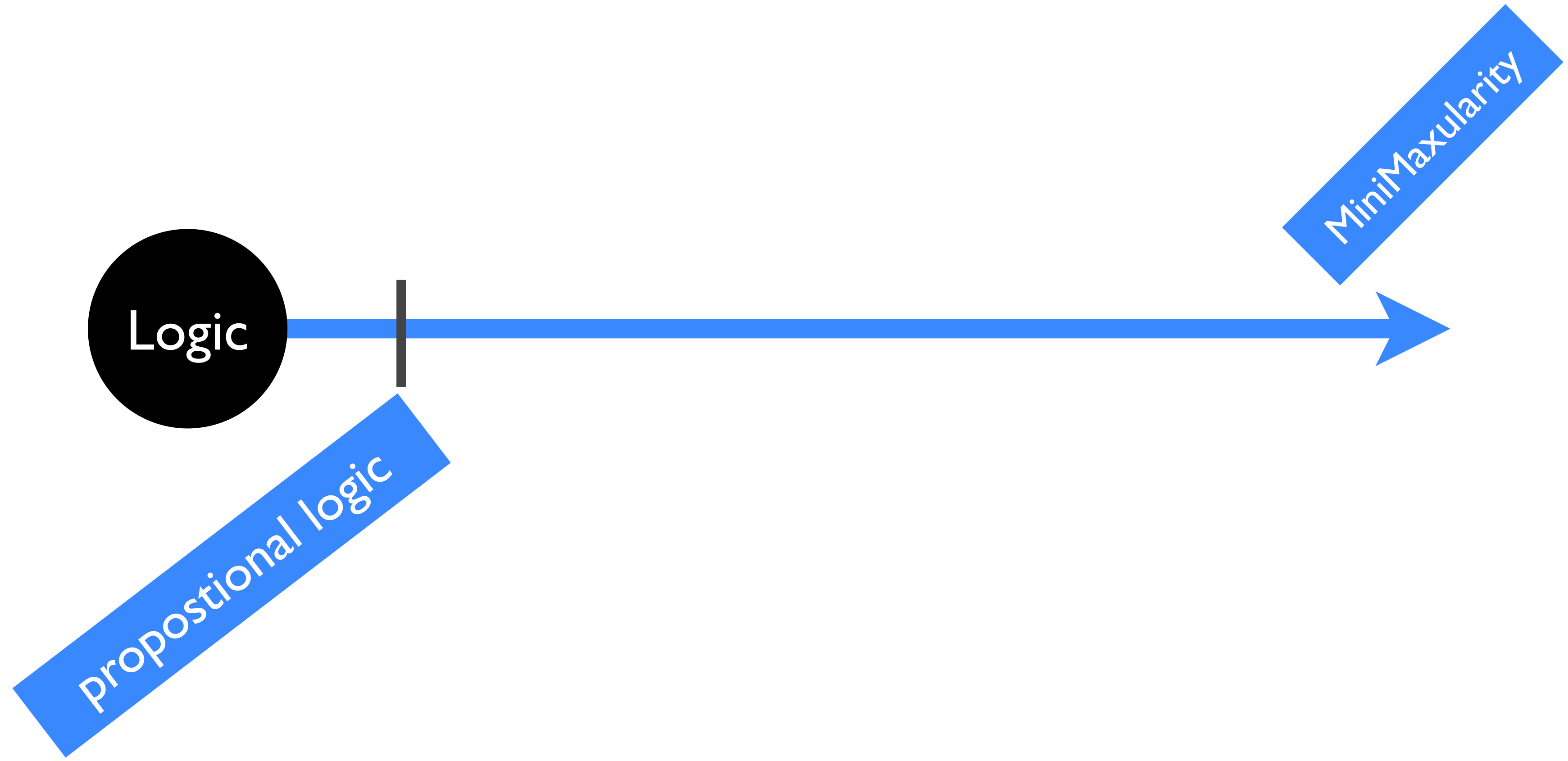
I can simulate a logic machine that's at least a Turing machine … e.g. by simulating some of what `o1 preview` provides me. Let's explore for SOL …

“‘I don’t yet know how to handle ‘non-linearity' in all of this, precisely. Maybe you can help. Here are some pointers, thoughts, initial constraints/structures …’’

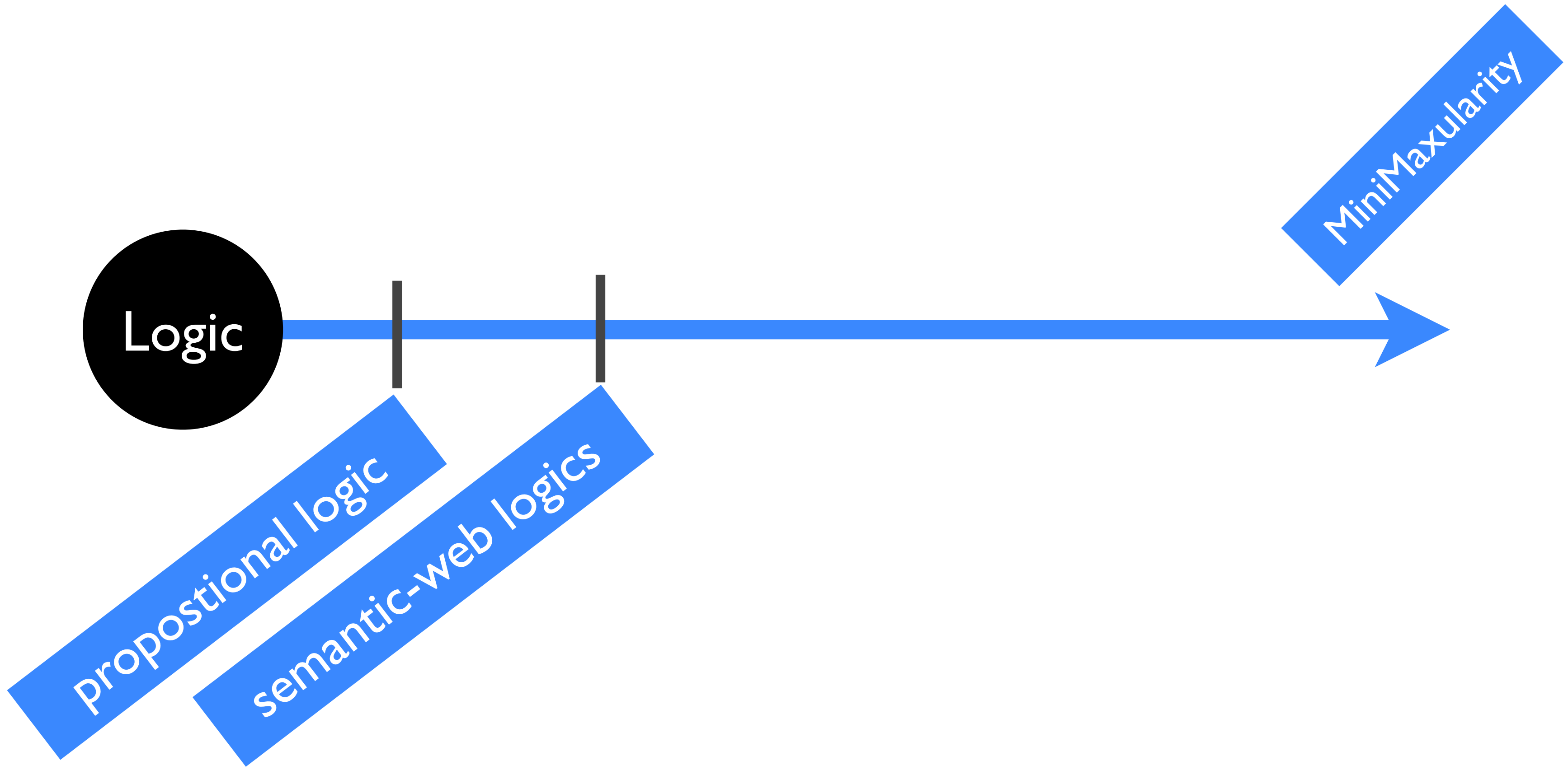
Logic

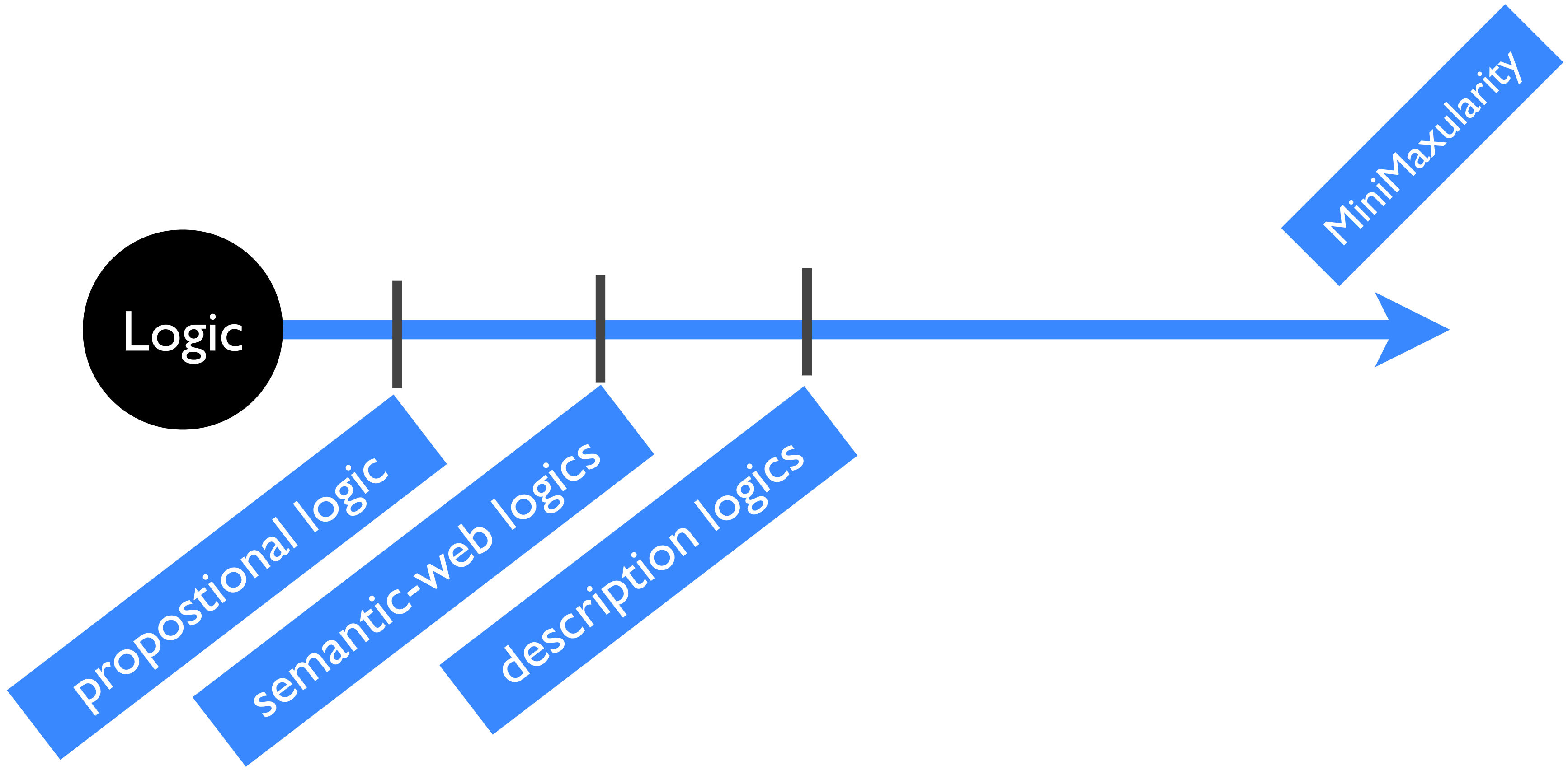

Logic

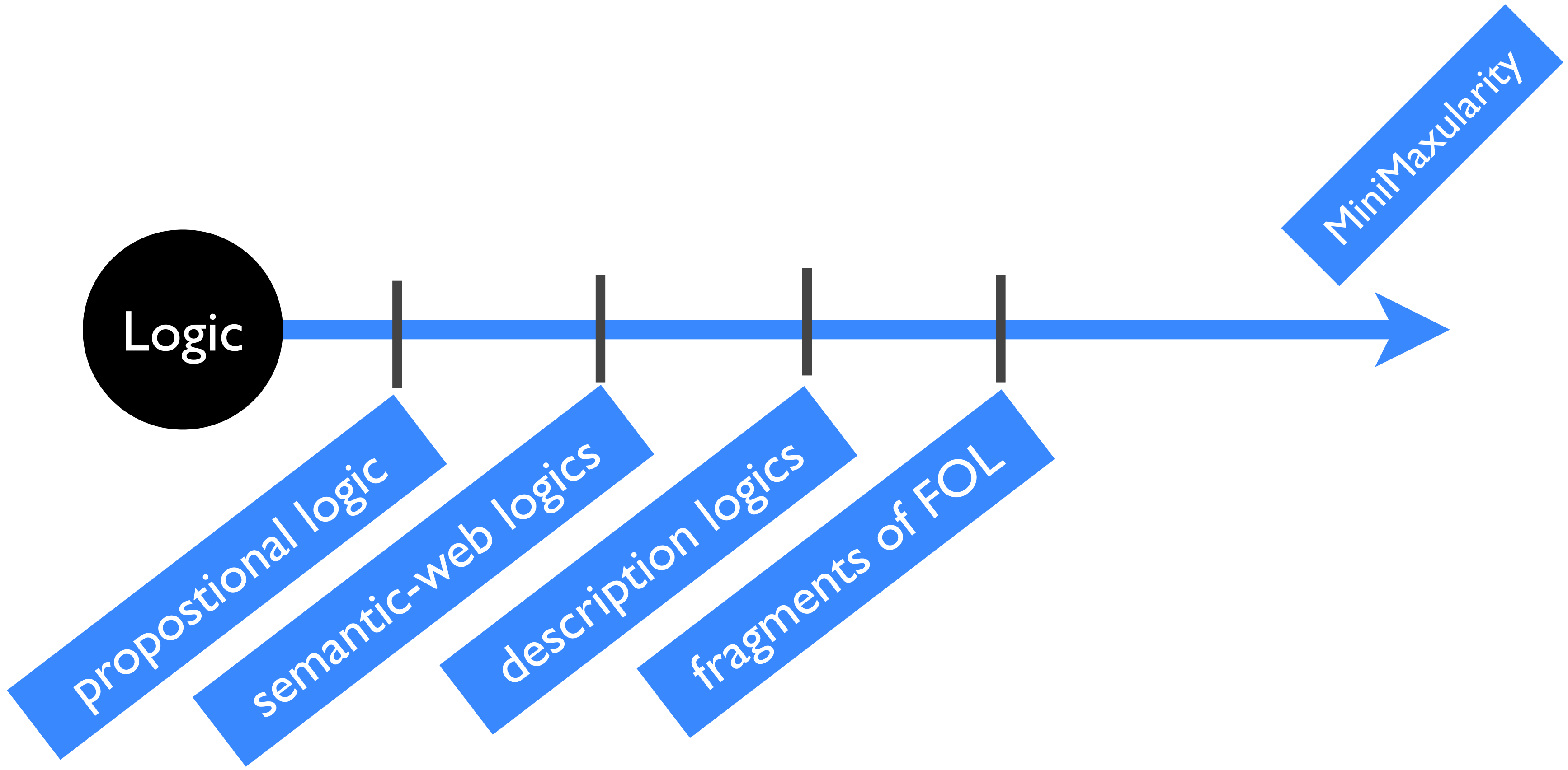

Logic
MiniMaxularity

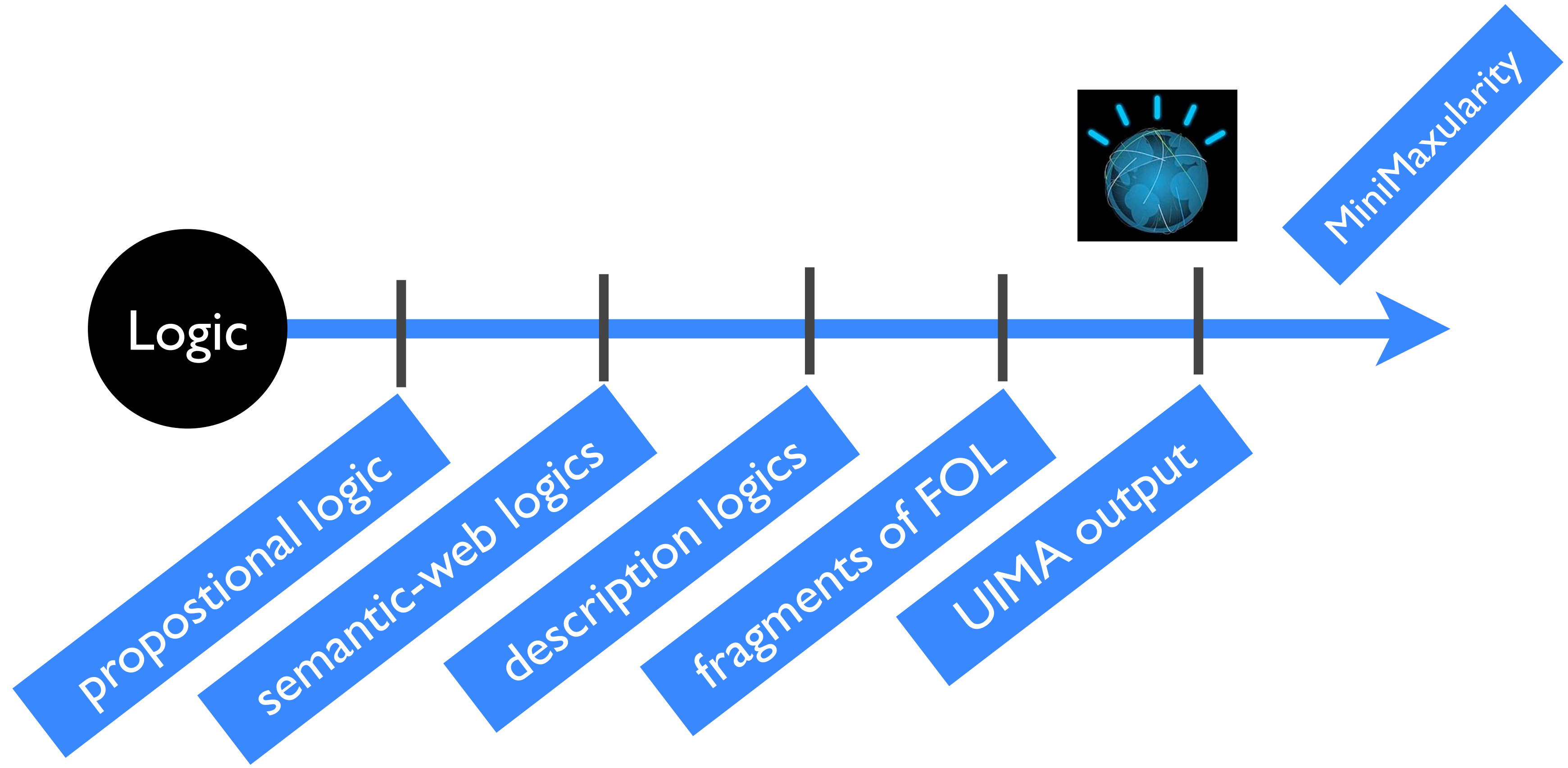


Logic
propostional logic
MiniMaxularity

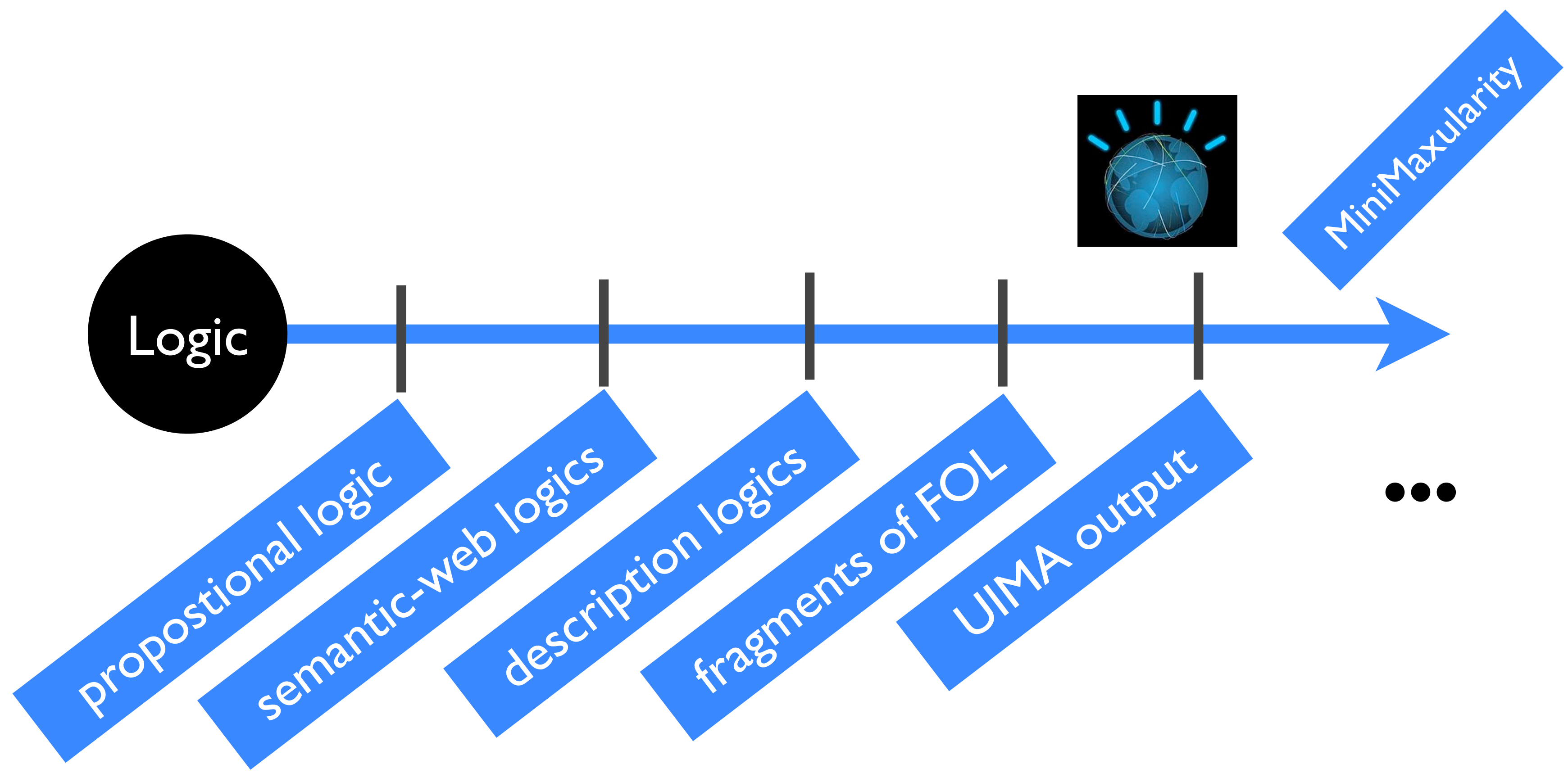

Logic
propostional logic
semantic-web logics
MiniMaxularity

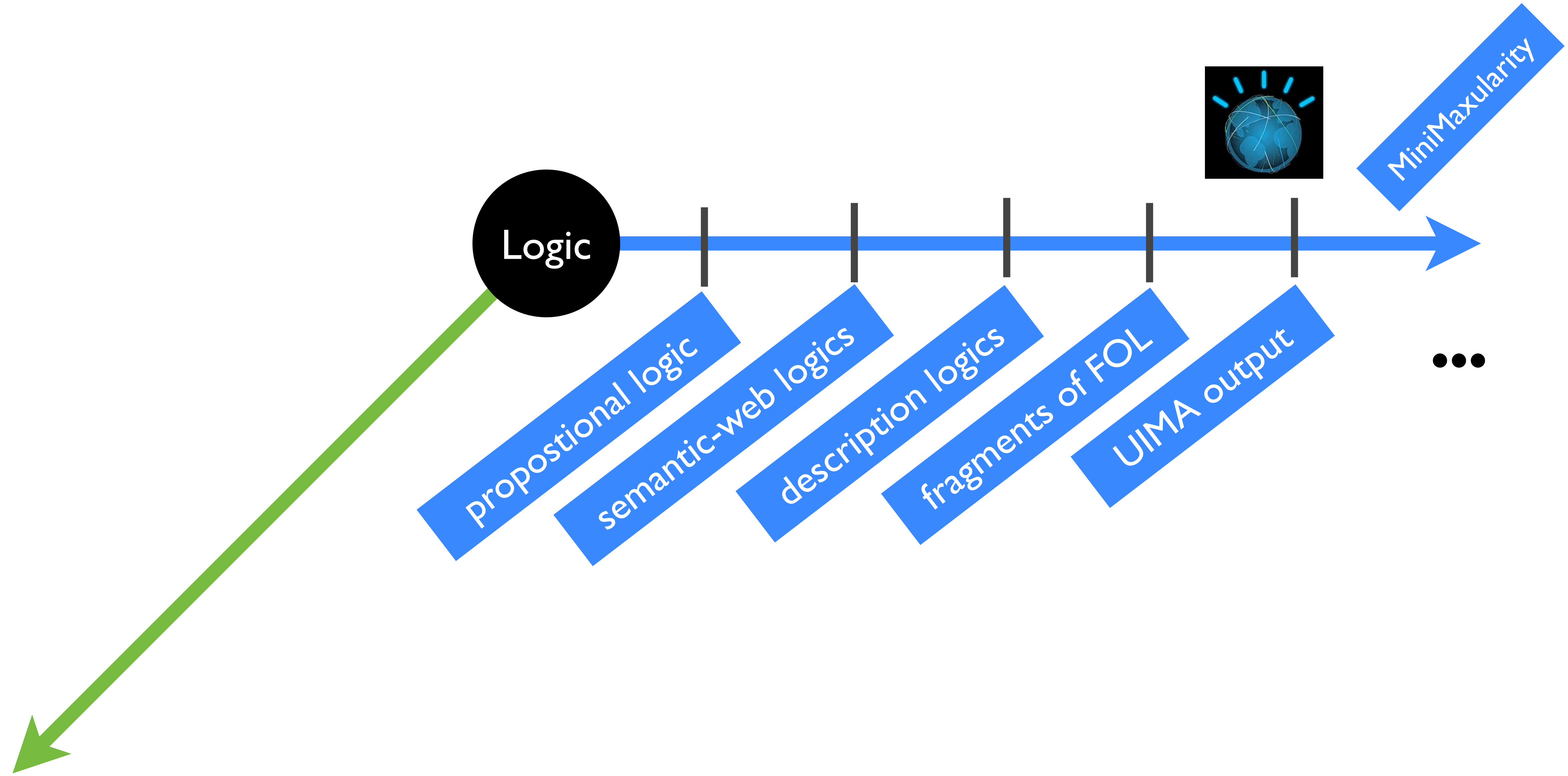

Logic
propostional logic
semantic-web logics
description logics
MiniMaxularity

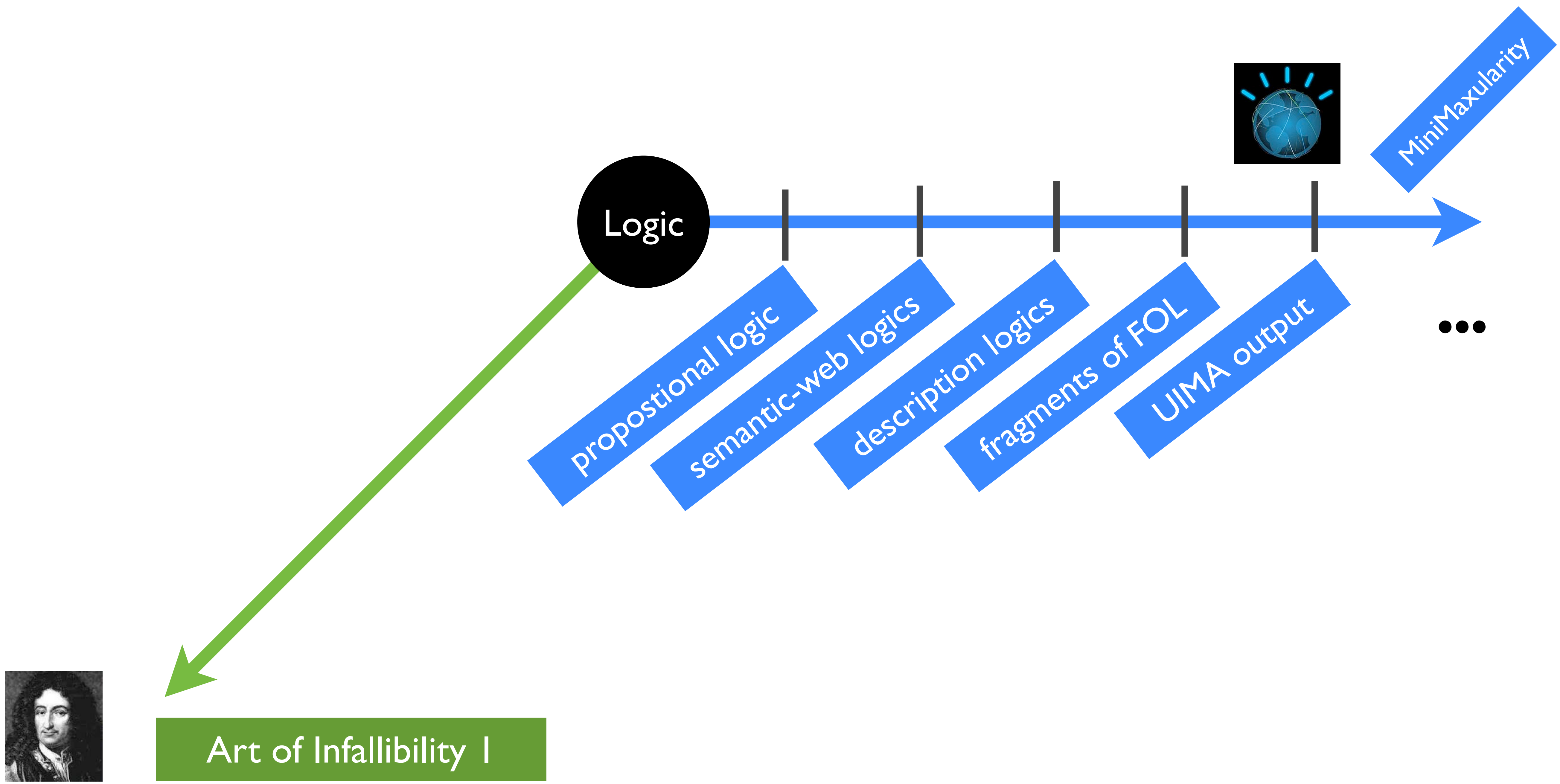

Logic
propostional logic
semantic-web logics
description logics
fragments of FOL
MiniMaxularity

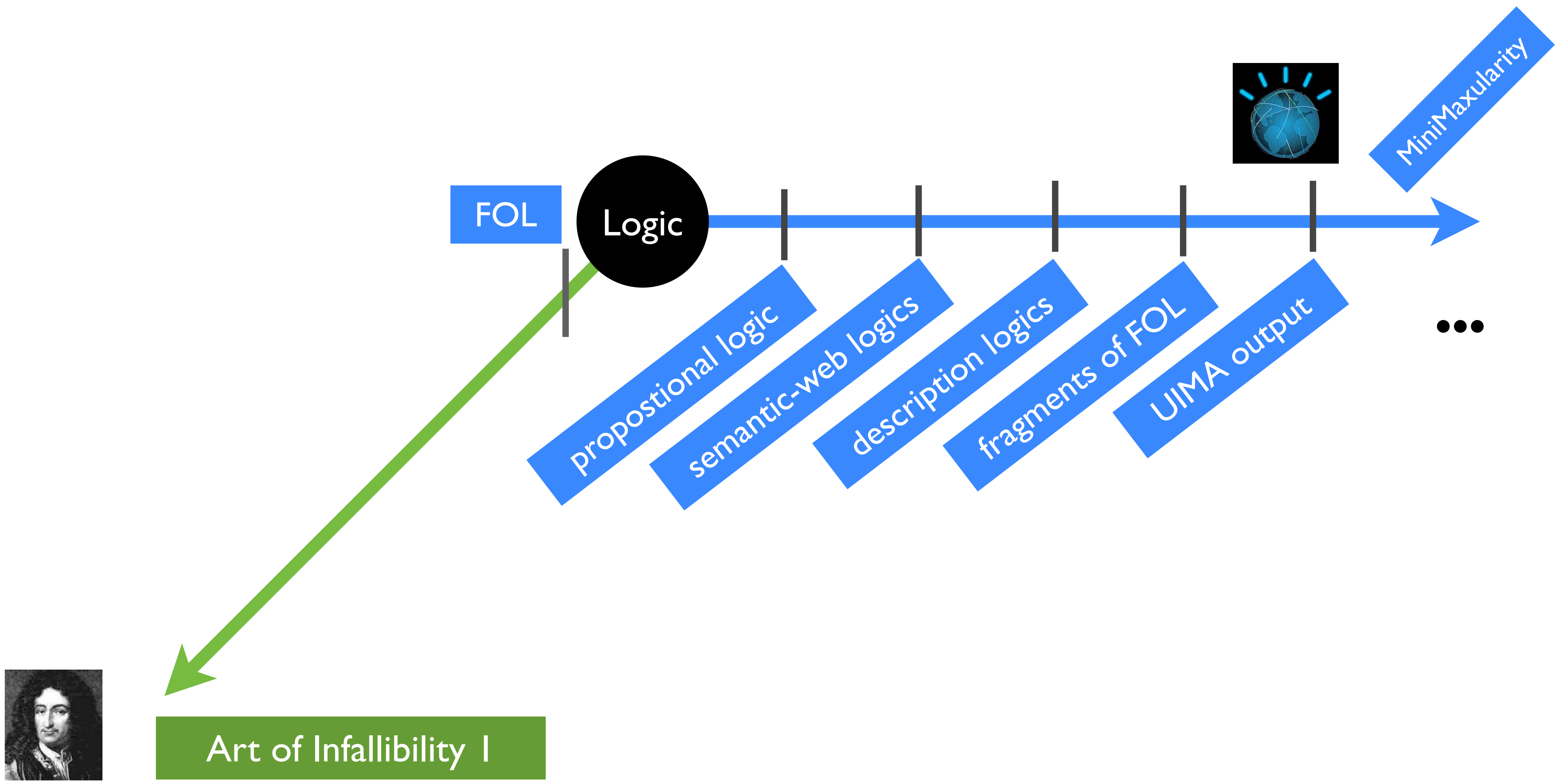


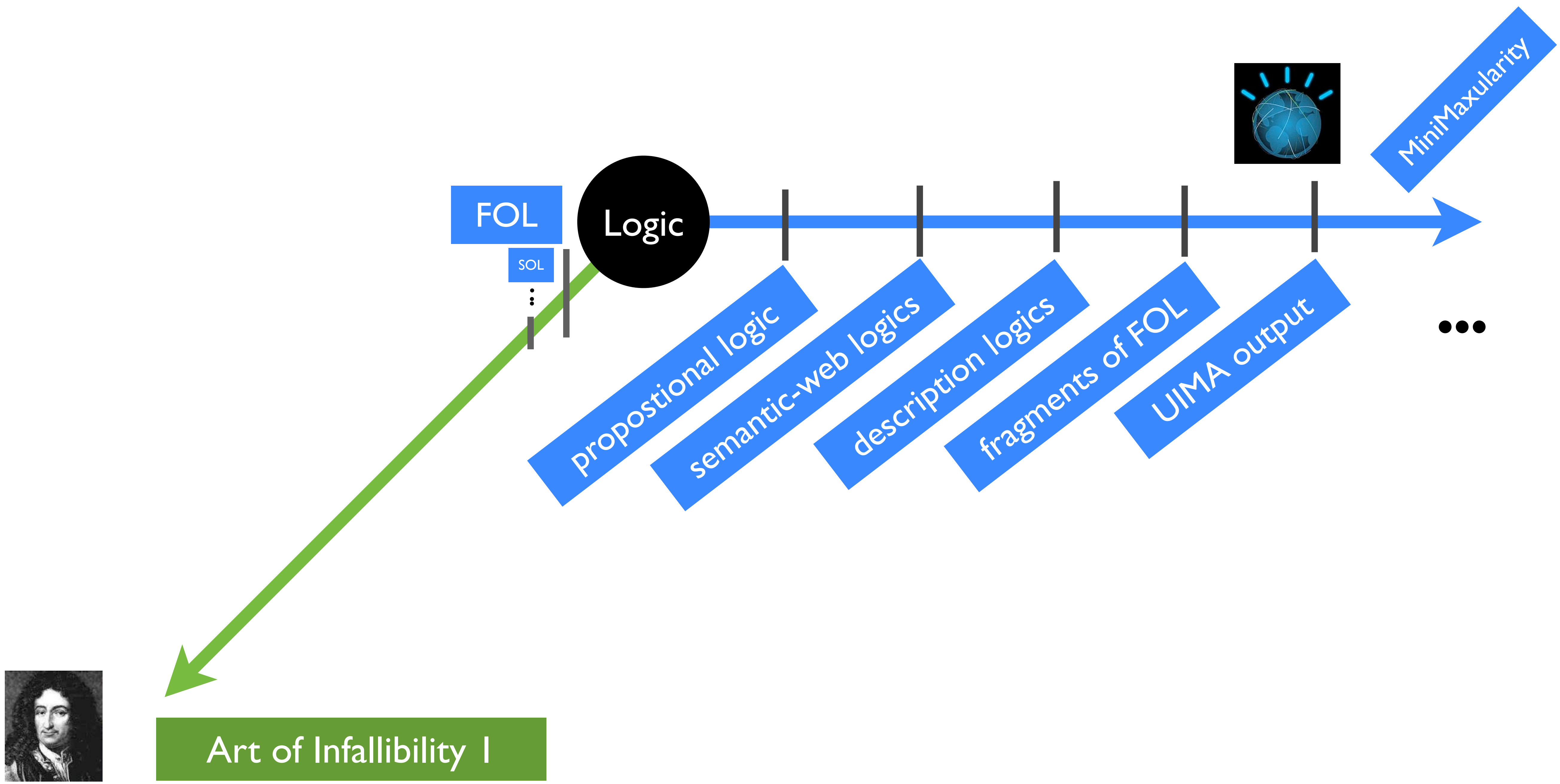
Logic
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity


Logic
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity
...

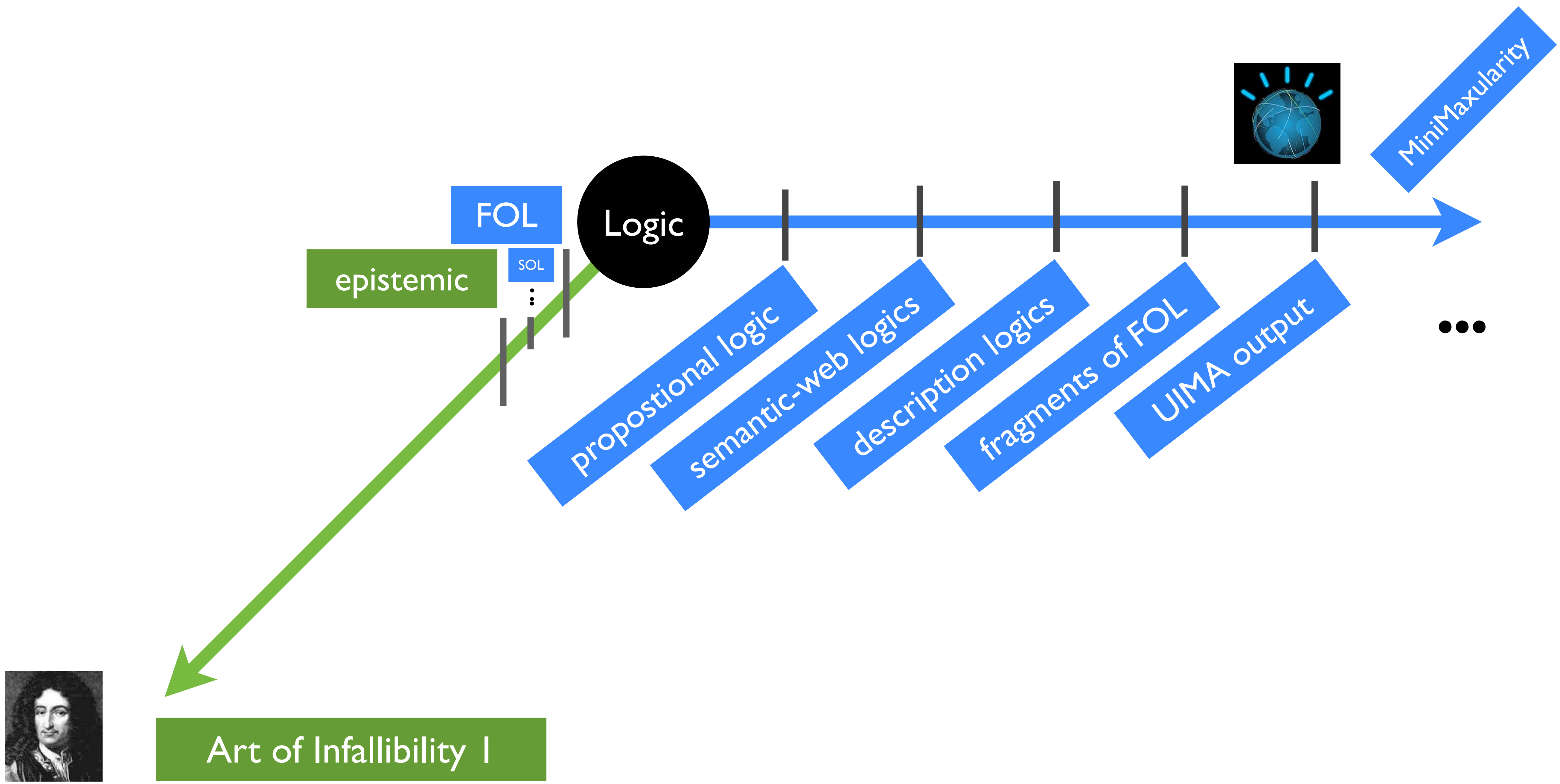

Logic
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity
...


Logic
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity
...
Art of Infallibility I

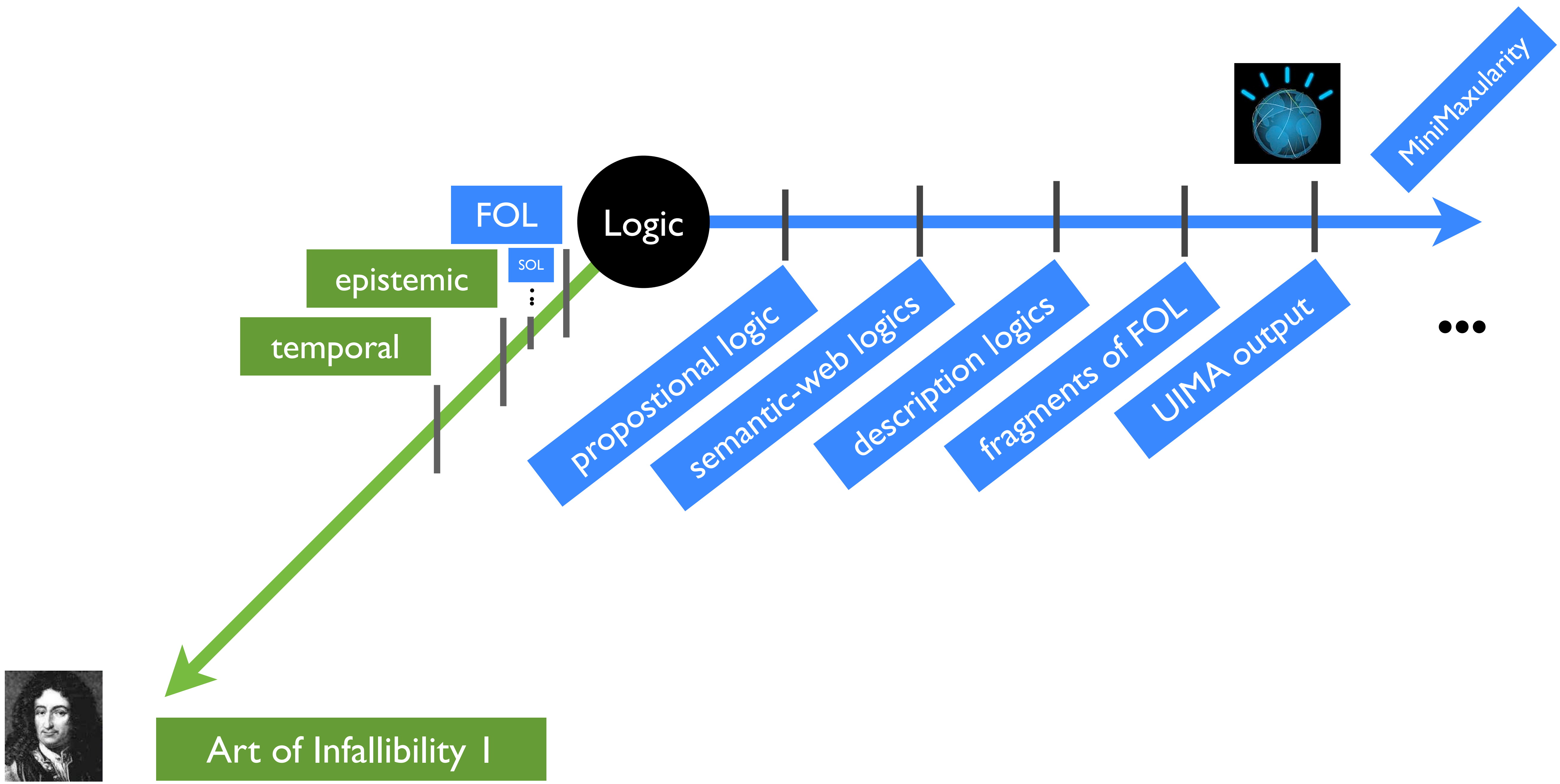

FOL
Logic
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity
...
Art of Infallibility I

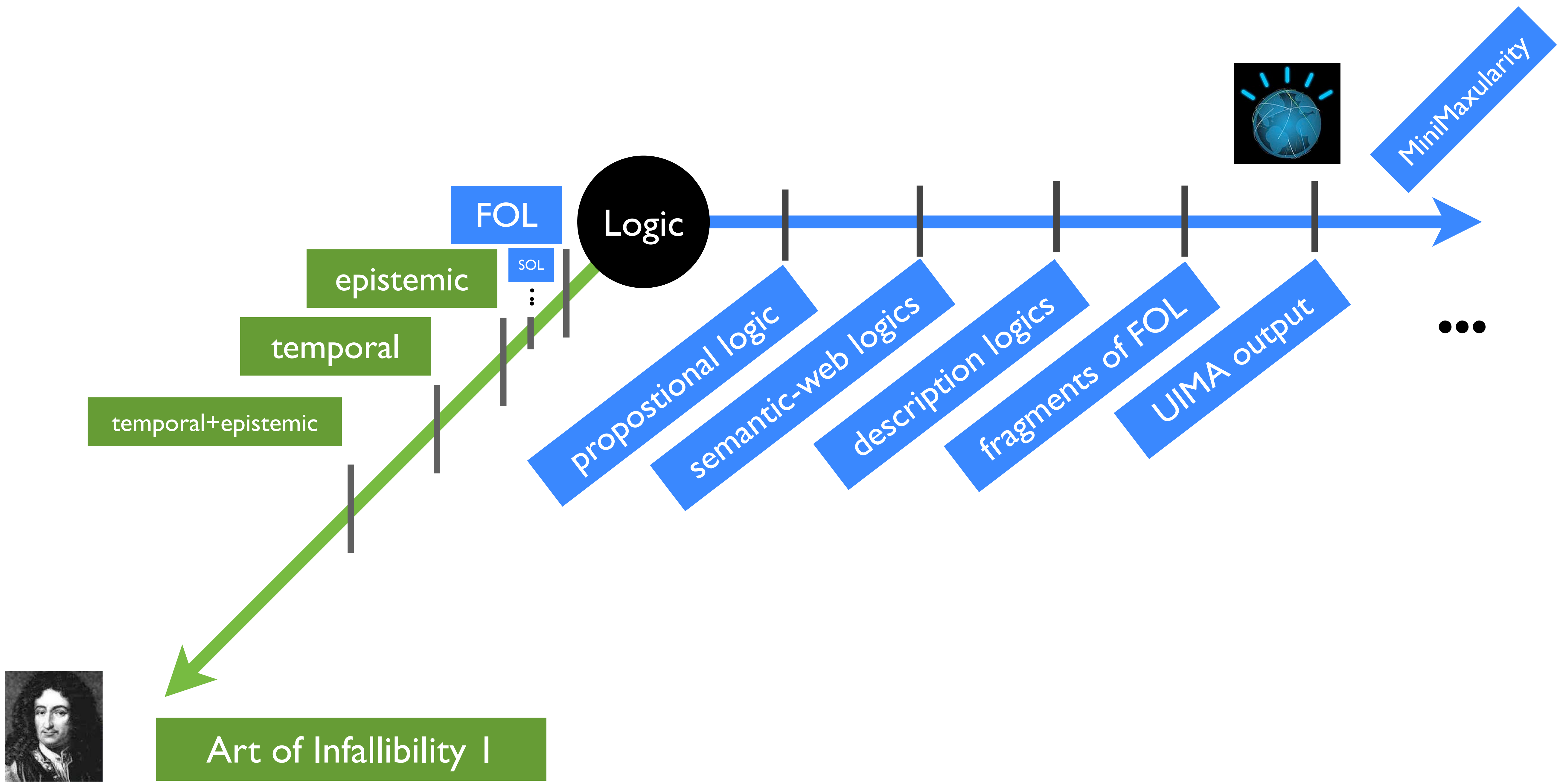


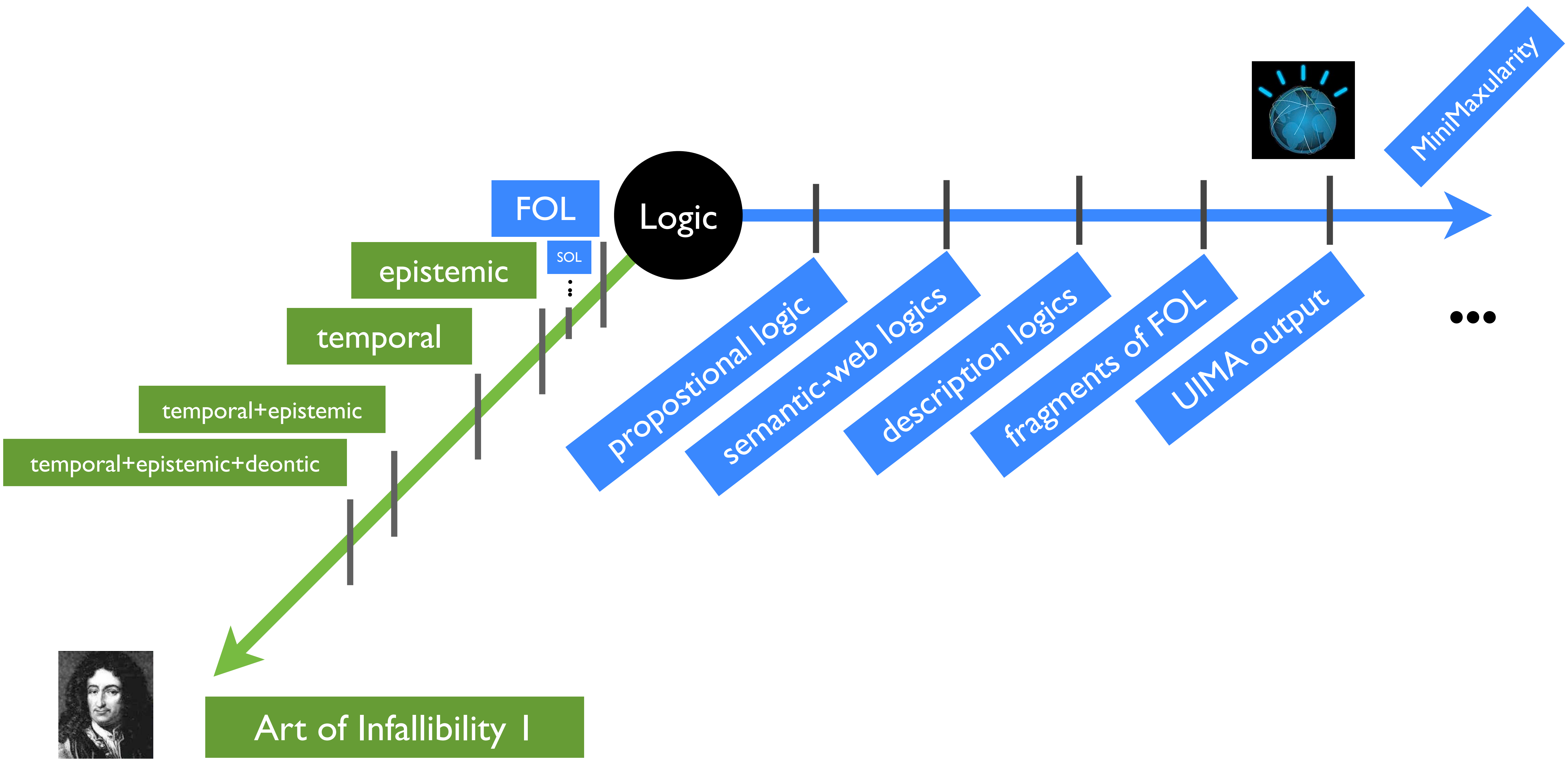
FOL
SOL
Logic
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity
...
Art of Infallibility I



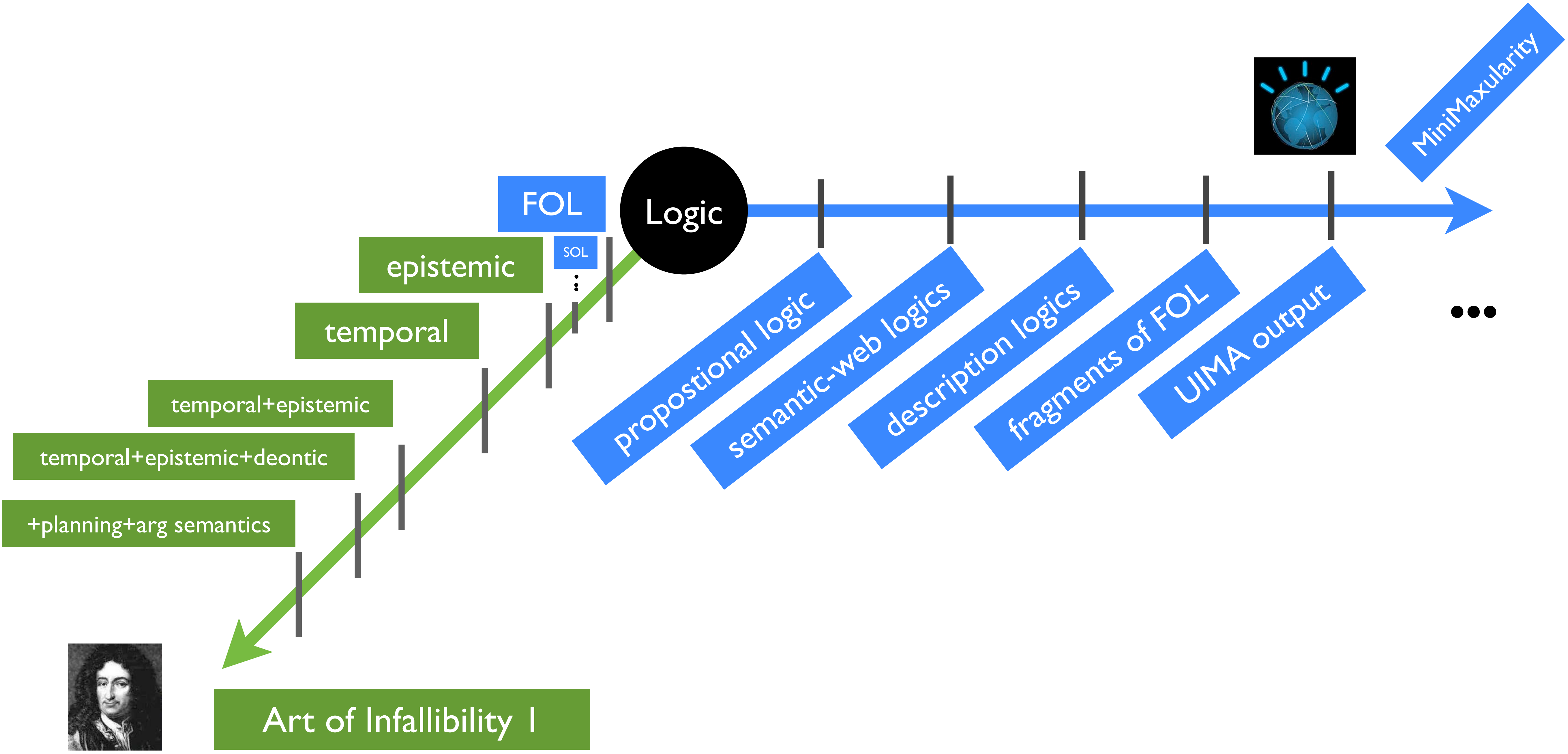
FOL
Logic
epistemic
SOL
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity
...
Art of Infallibility I

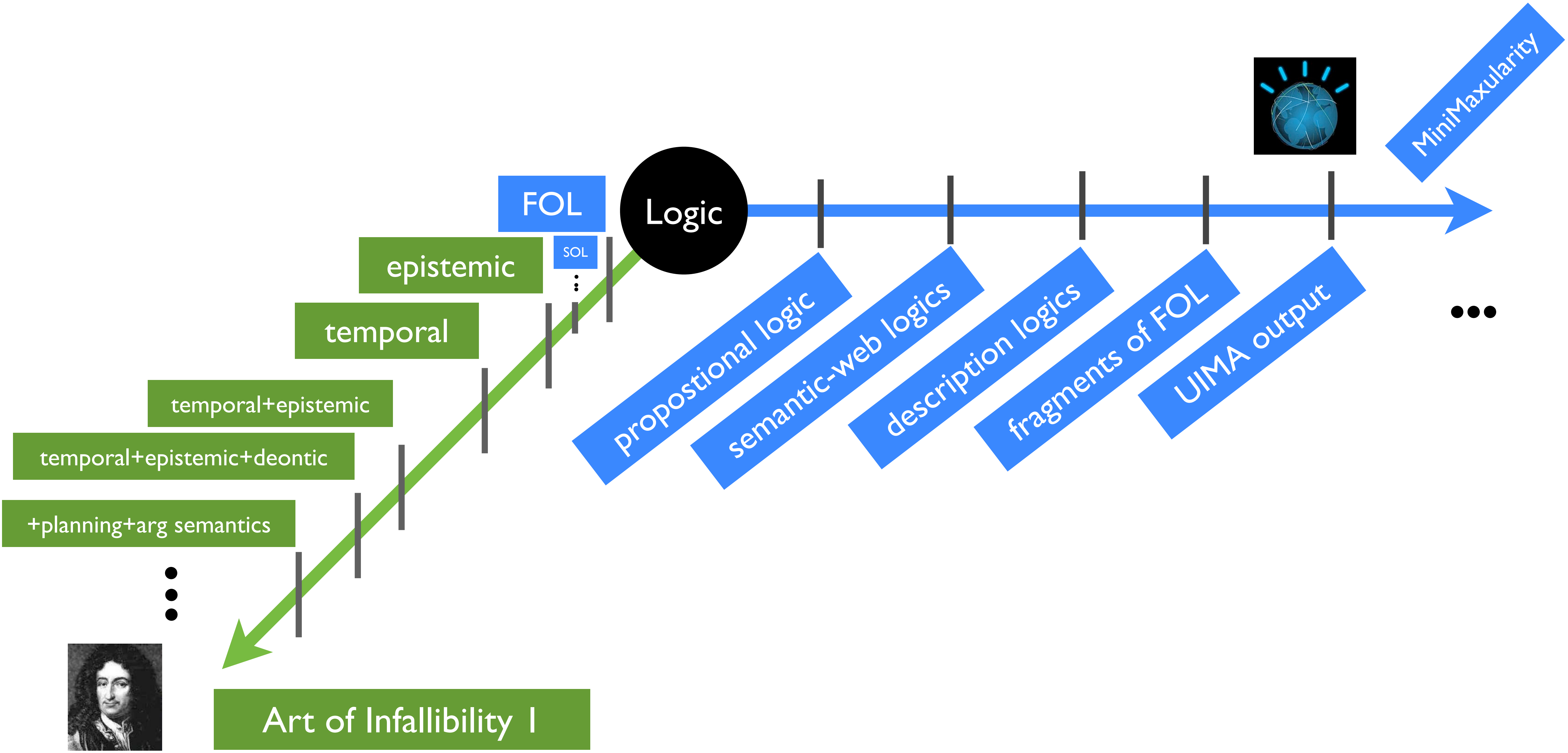

FOL
Logic
SOL
epistemic
temporal
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity
...
Art of Infallibility I

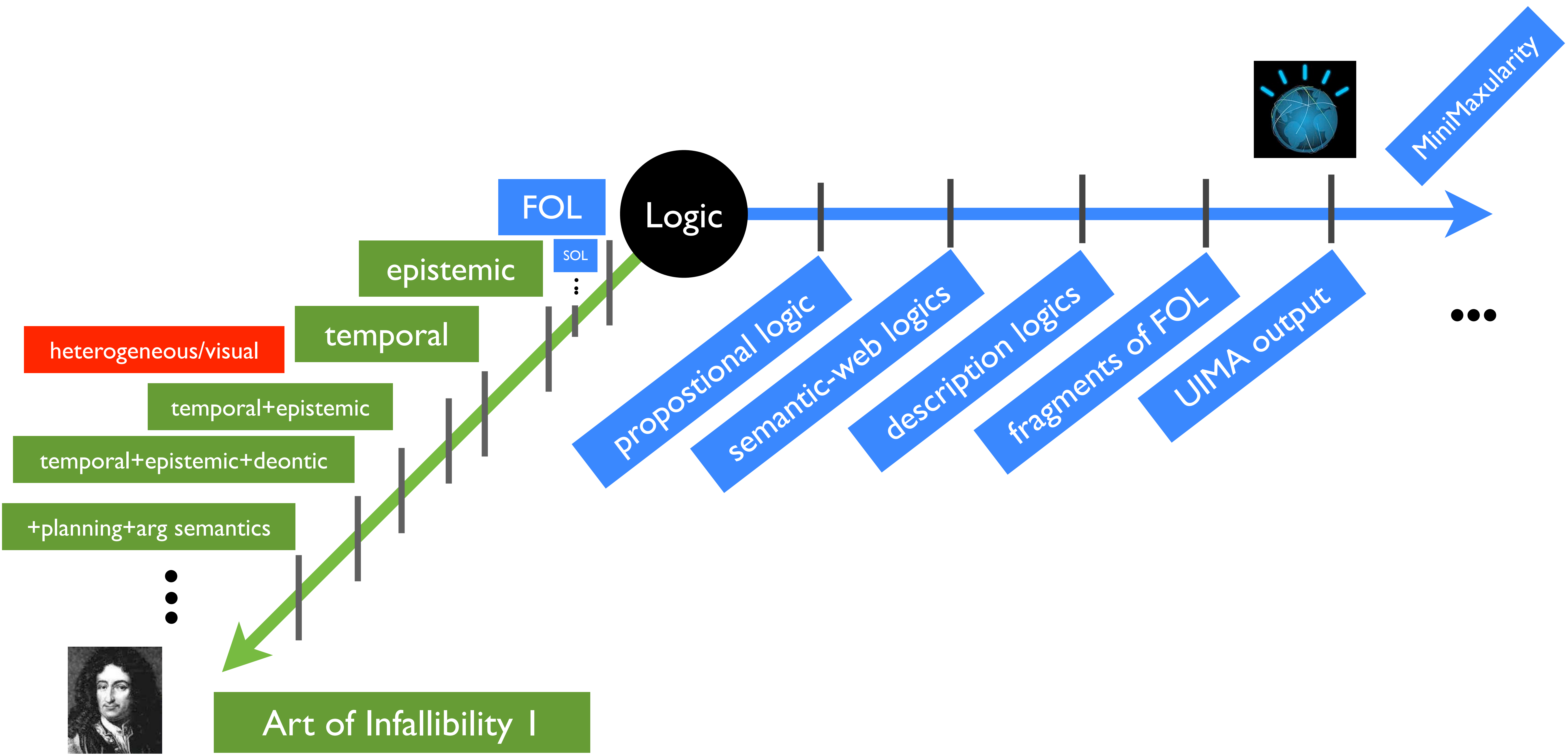

FOL
Logic
SOL
epistemic
temporal
temporal+epistemic
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity
•••
Art of Infallibility I



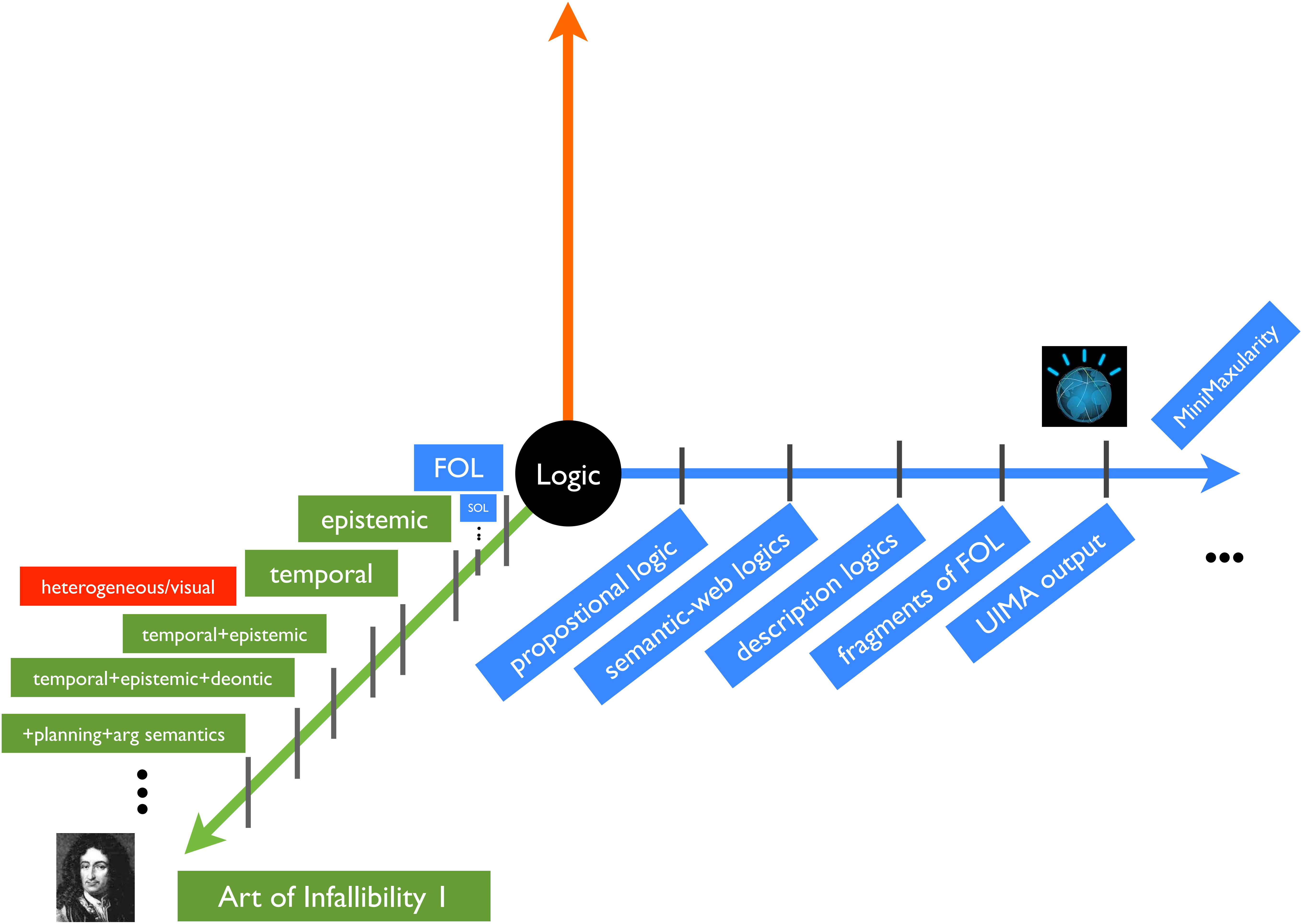
FOL
Logic
SOL
epistemic
temporal
temporal+epistemic
temporal+epistemic+deontic
Art of Infallibility I
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity
•••


FOL
Logic
SOL
epistemic
temporal
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
Art of Infallibility I
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity
•••

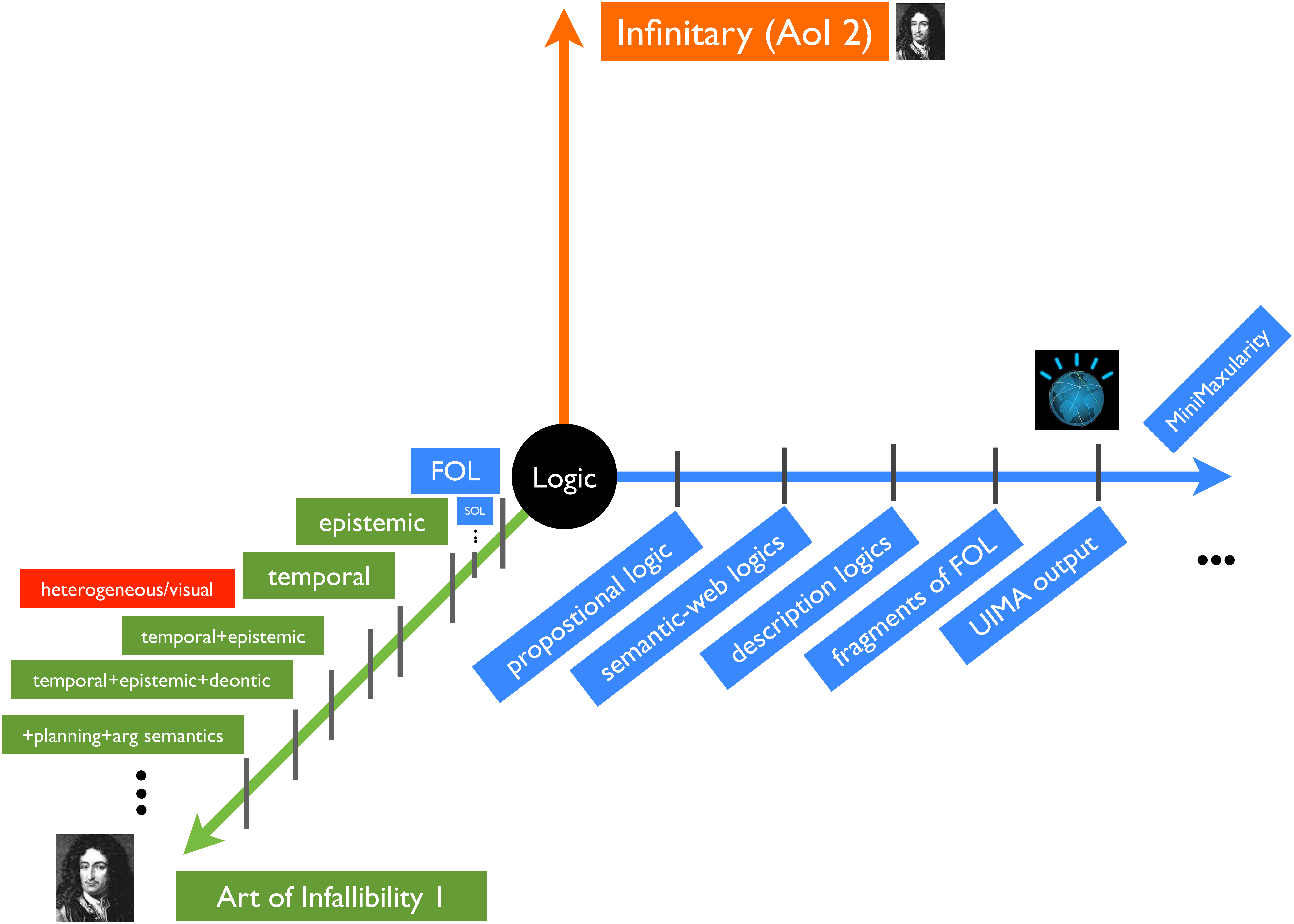

FOL
Logic
SOL
epistemic
temporal
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
Art of Infallibility I
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity

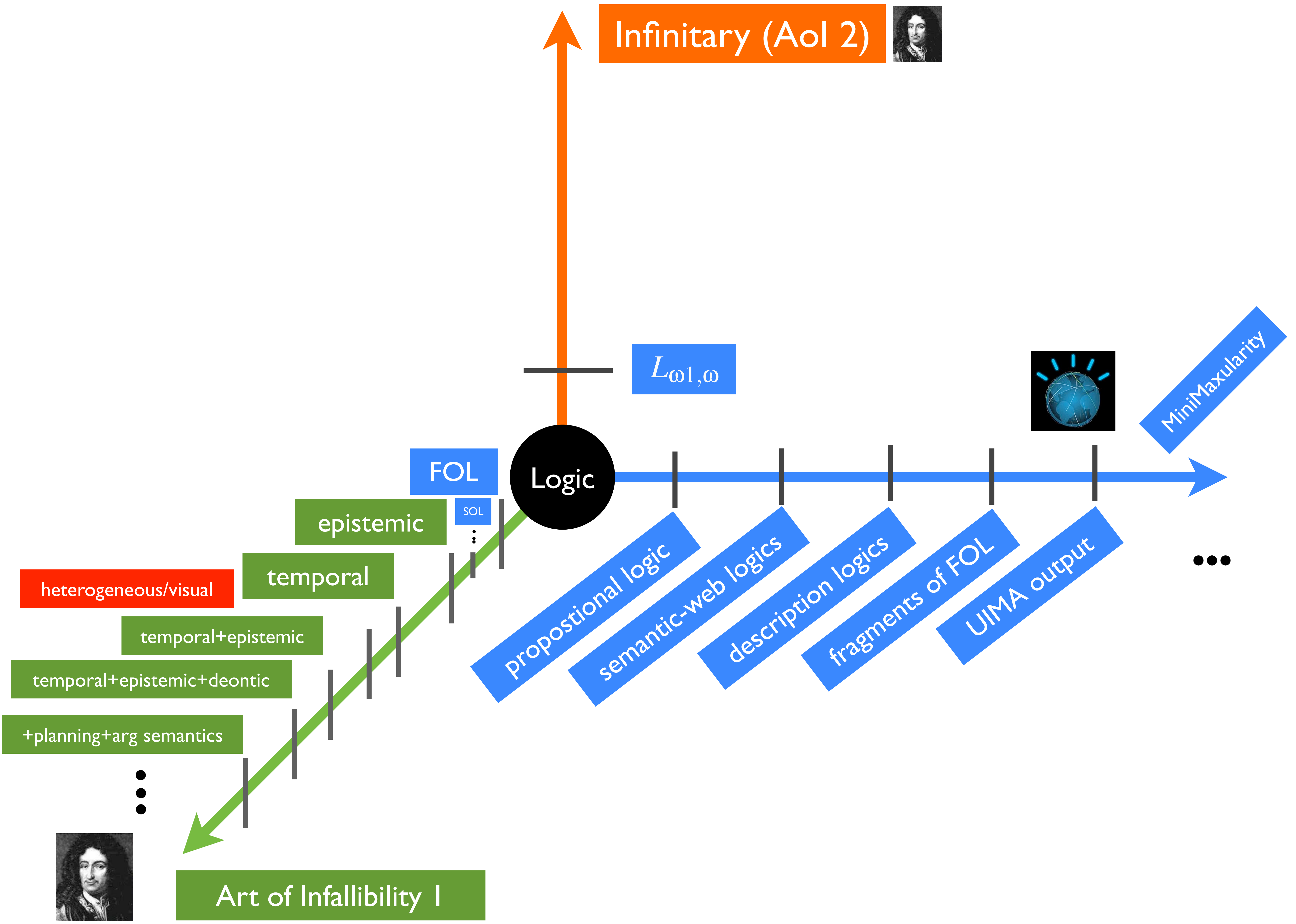


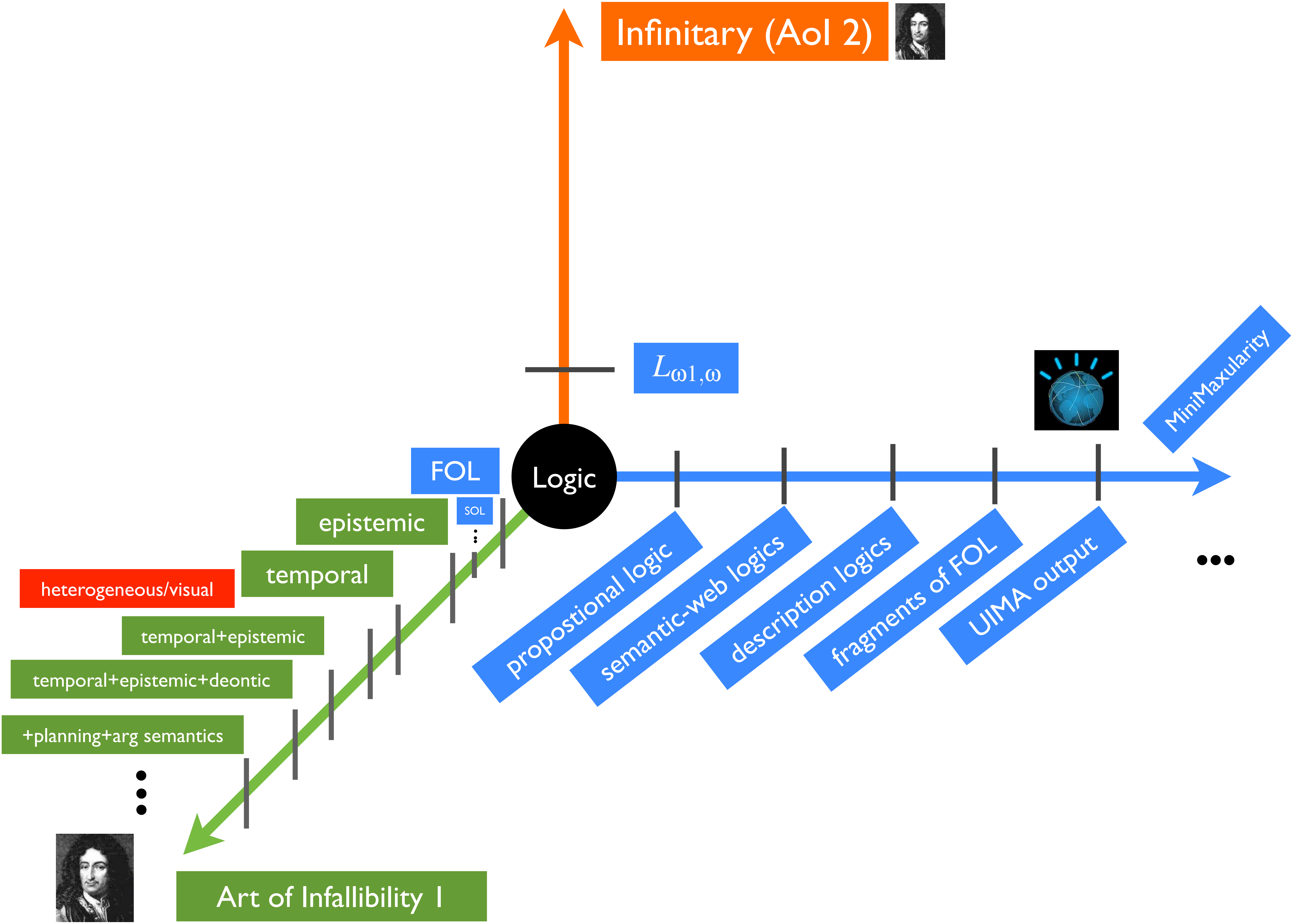
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
Art of Infallibility I
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity



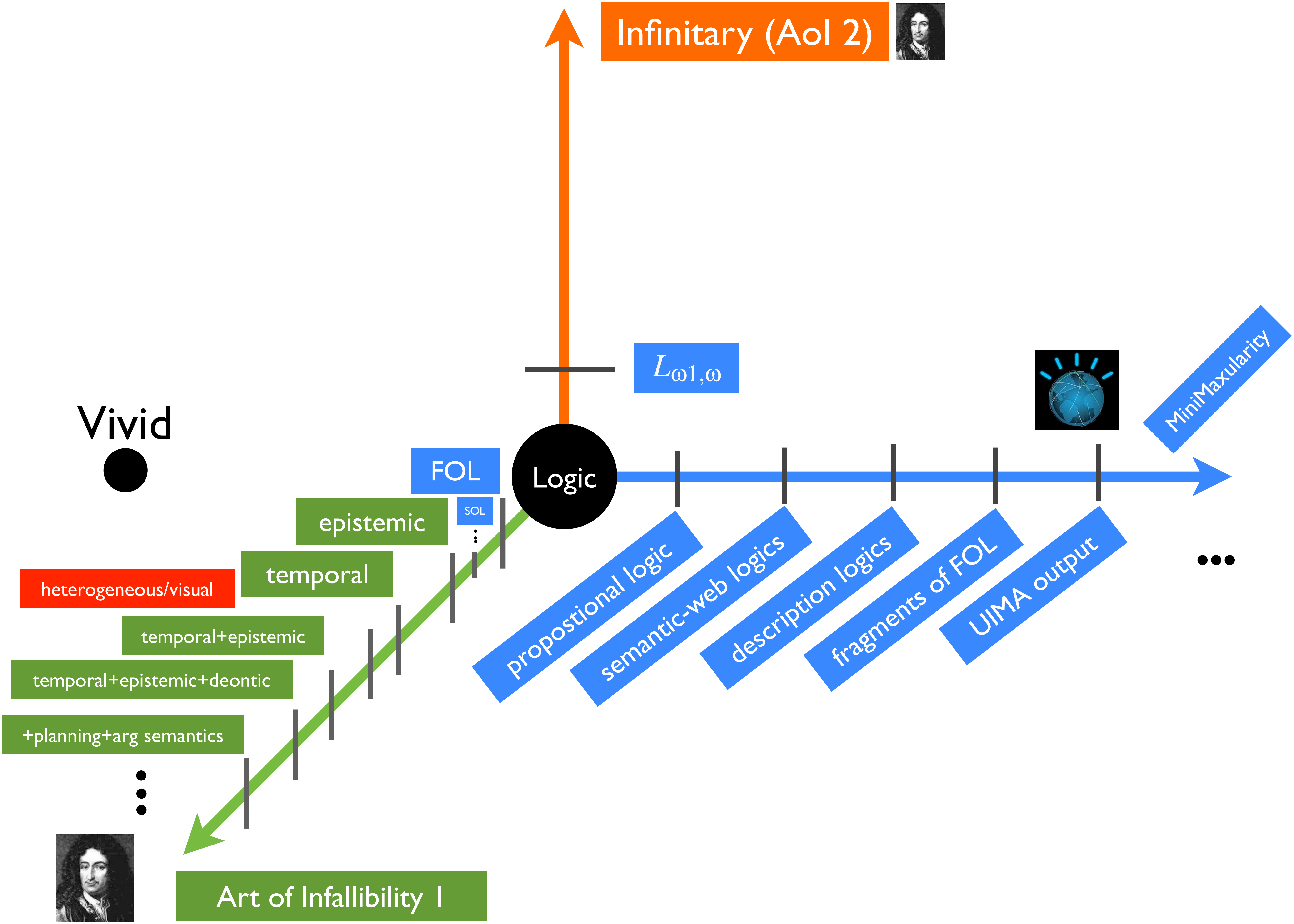
Logic
FOL
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
Art of Infallibility I
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity

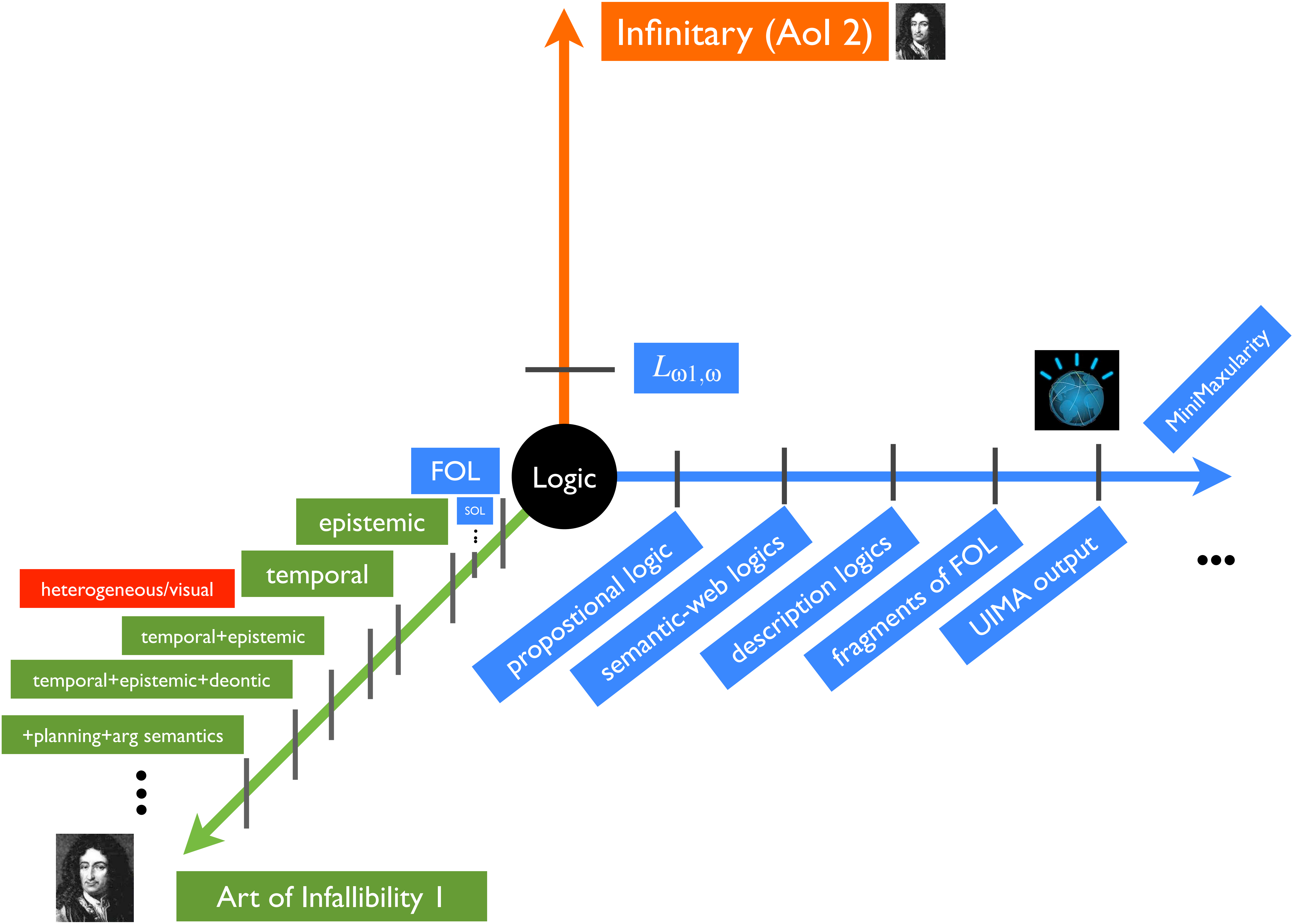

Infinitary (AoI 2)
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I

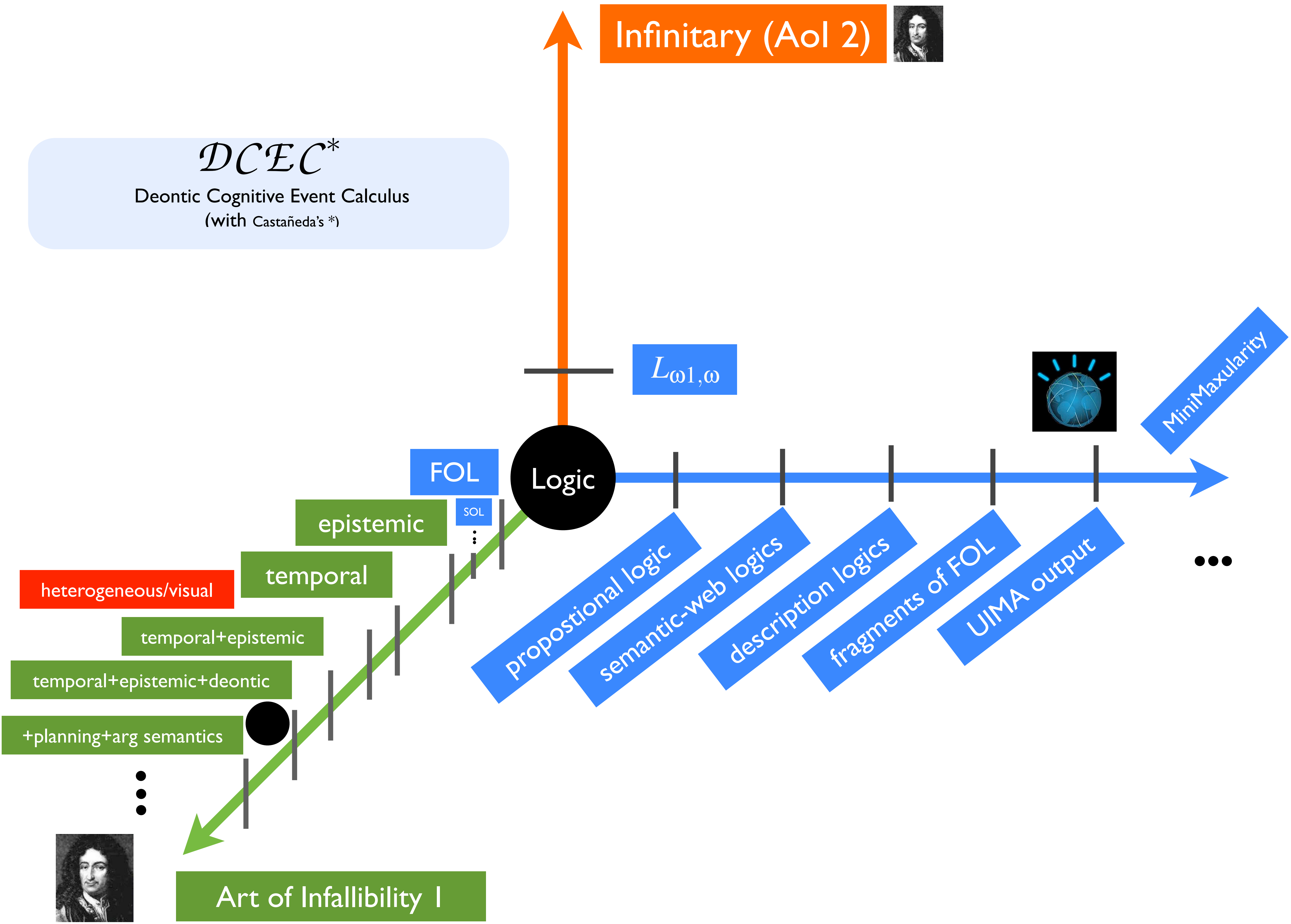

Infinitary (AoI 2)
$L_{\omega 1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I



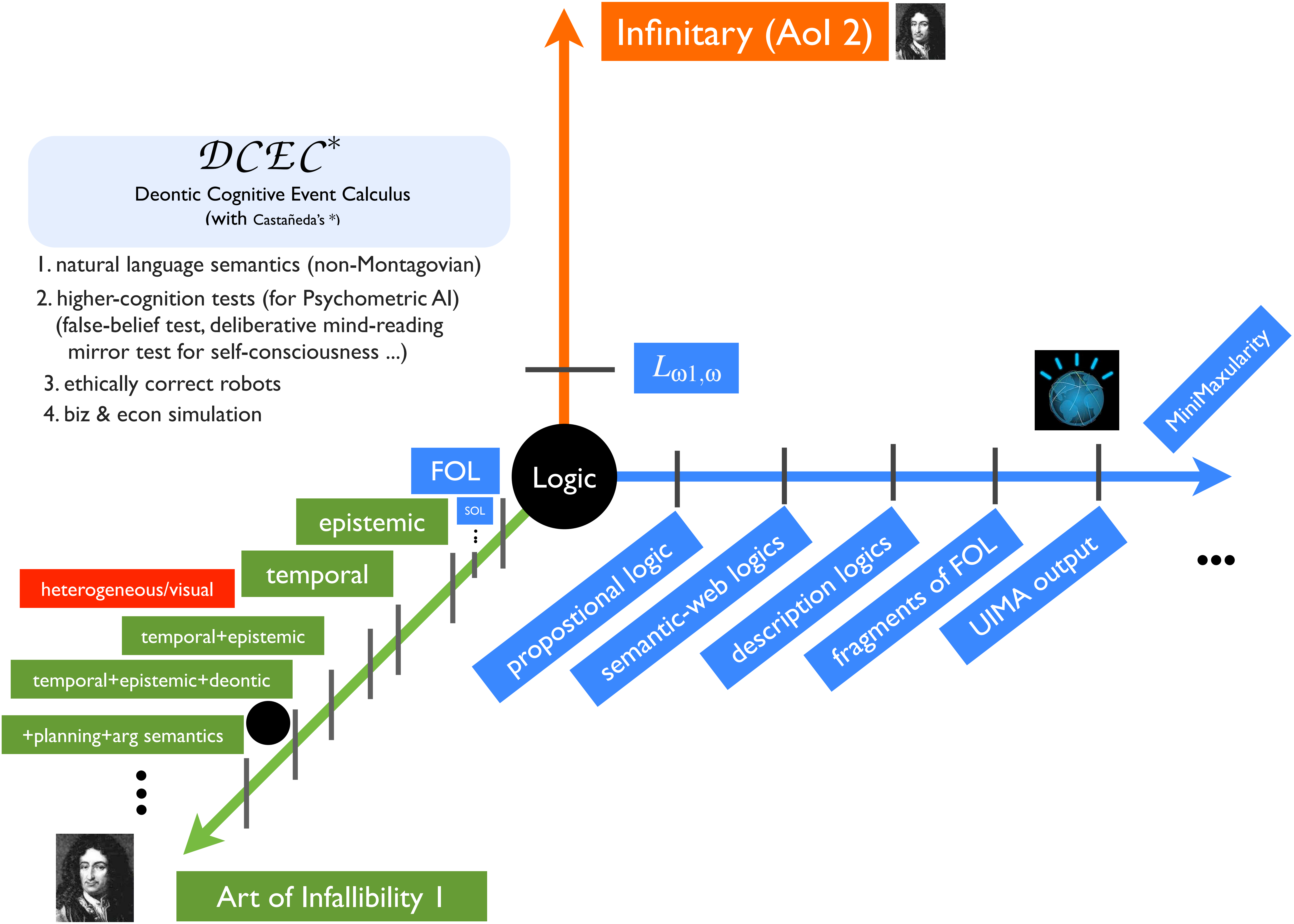
Infinitary (AoI 2)
$L_{\omega 1,\omega}$
Logic
FOL
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
Art of Infallibility I
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity

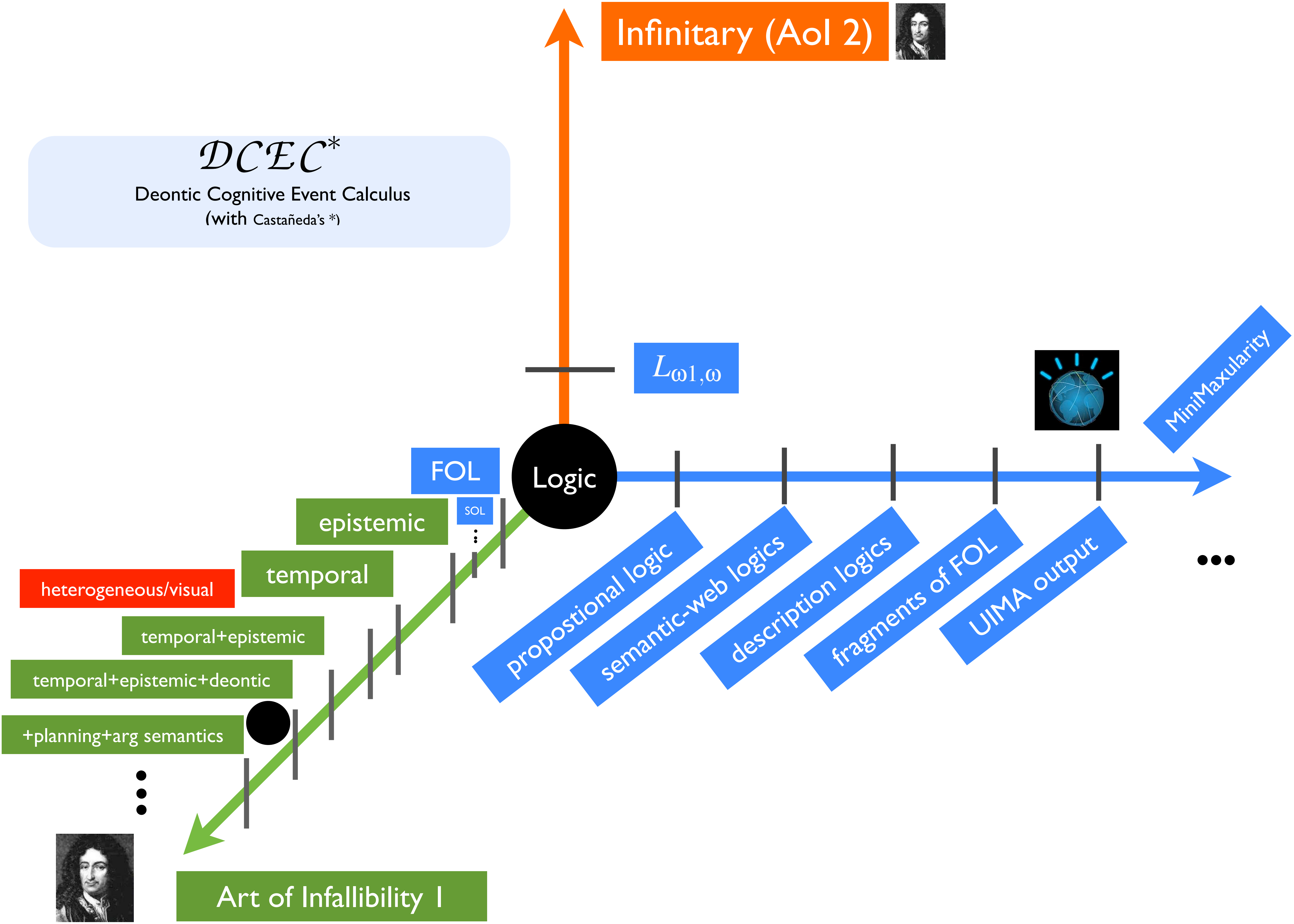

Infinitary (AoI 2)
$L_{\omega 1, \omega}$
Vivid
FOL
SOL
Logic
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
Art of Infallibility I
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
MiniMaxularity
...

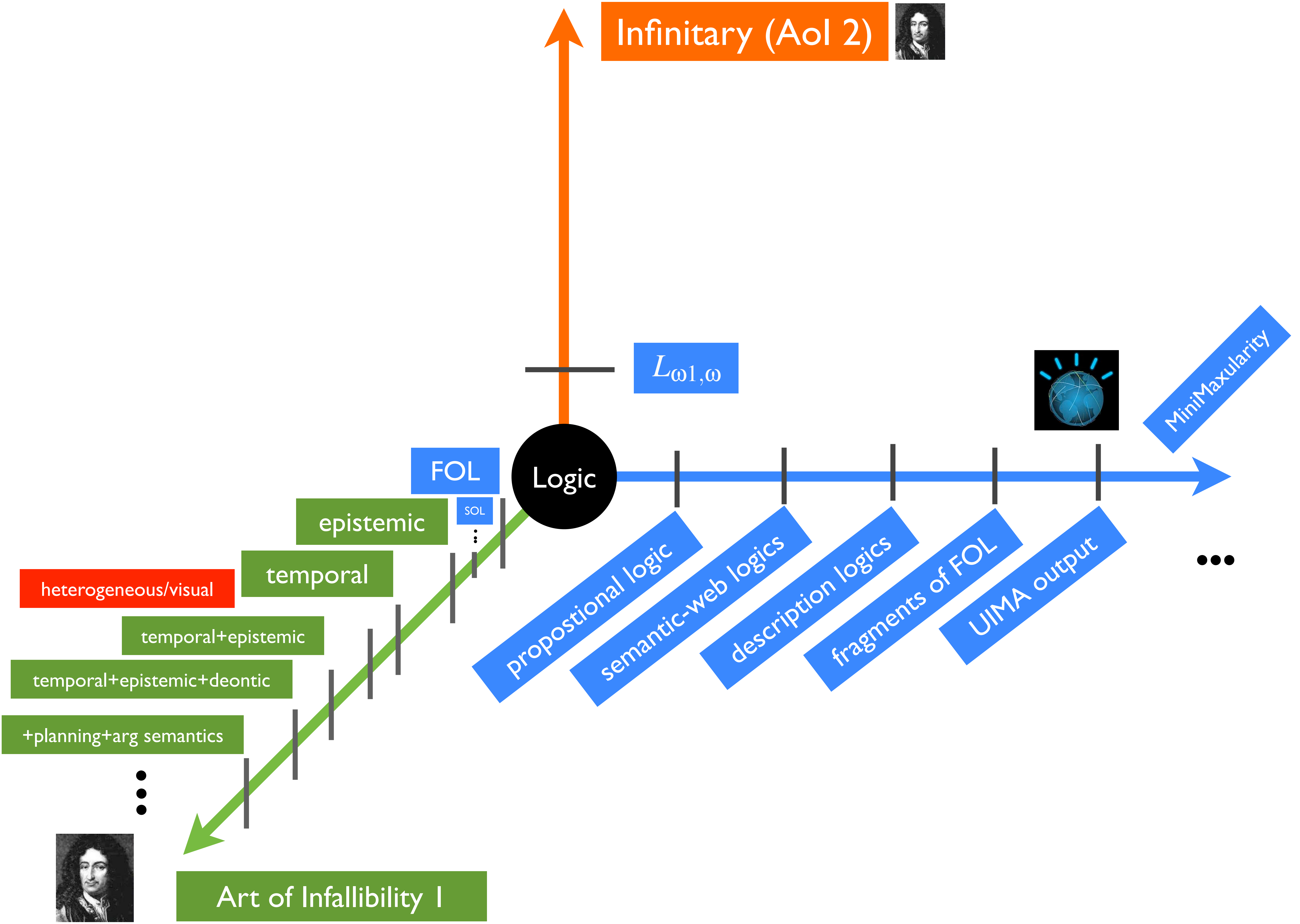

Infinitary (AoI 2)
$L_{\omega 1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I



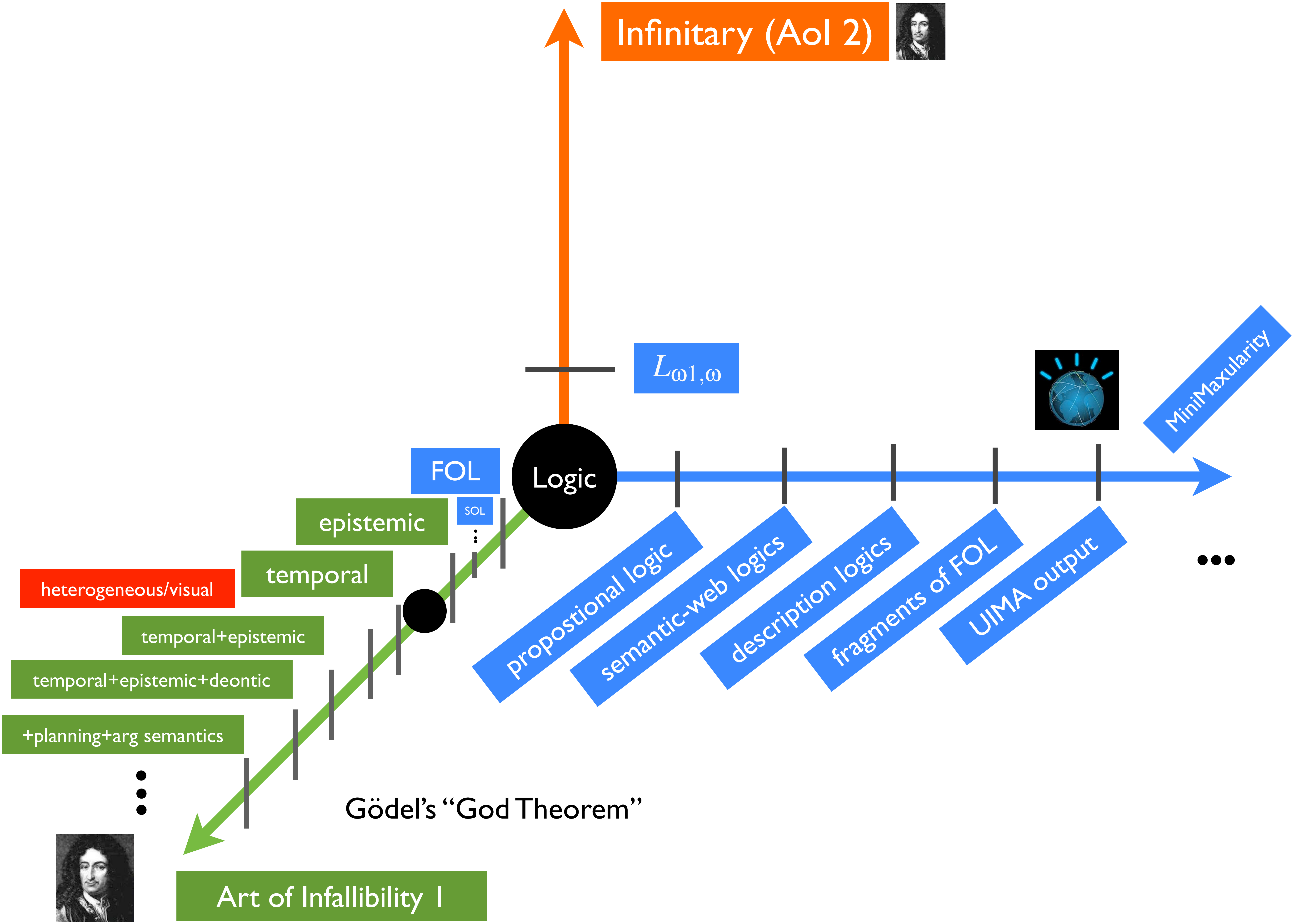
Infinitary (AoI 2)
$\mathcal{DCEC}^*$
Deontic Cognitive Event Calculus
(with Castañeda's *)
$L_{\omega 1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I


Infinitary (AoI 2)
$\mathcal{DCEC}^*$
Deontic Cognitive Event Calculus
(with Castañeda's *)
1. natural language semantics (non-Montagovian)
2. higher-cognition tests (for Psychometric AI)
(false-belief test, deliberative mind-reading
mirror test for self-consciousness ...)
3. ethically correct robots
4. biz & econ simulation
$L_{\omega1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
...
Art of Infallibility I

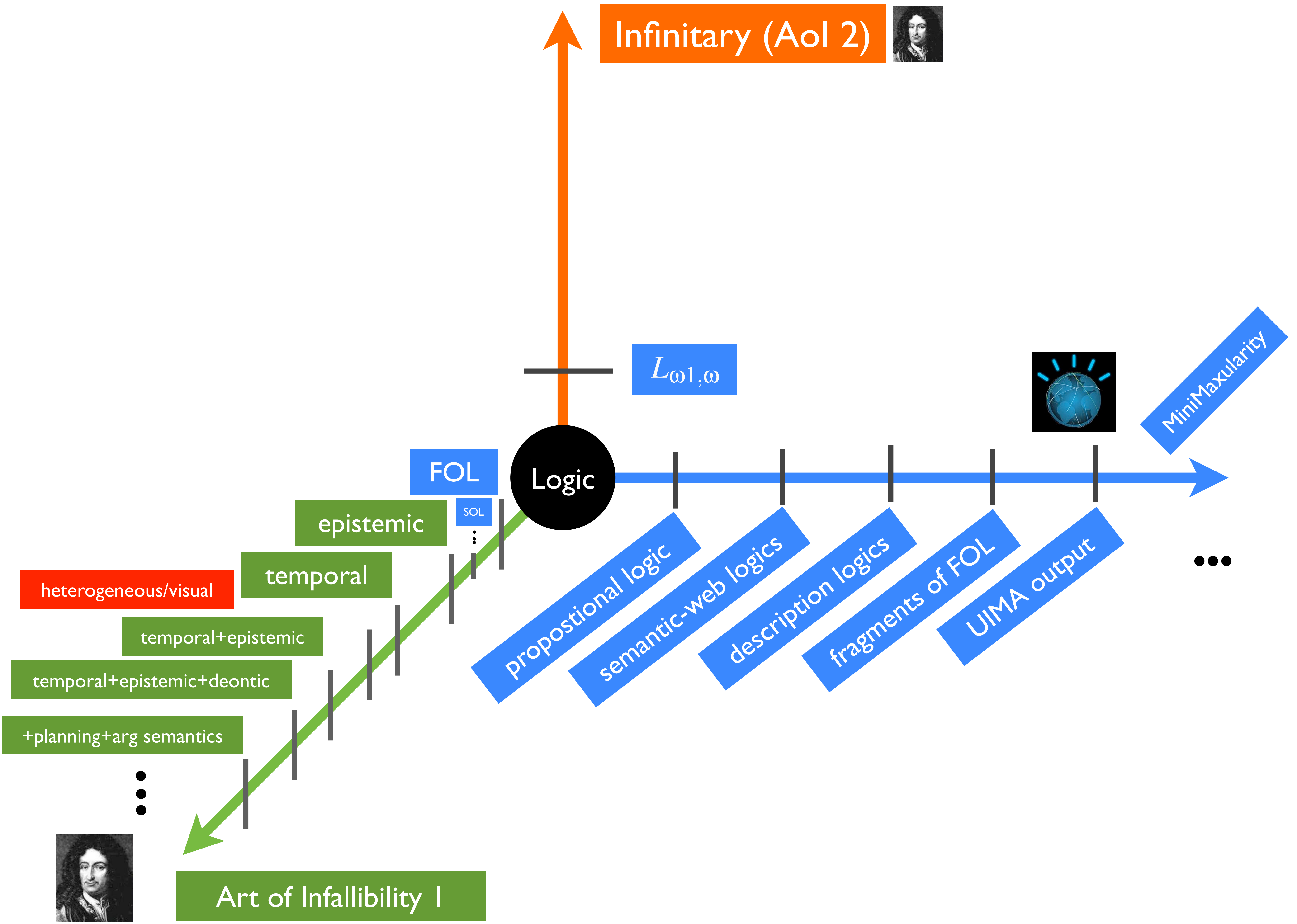

Infinitary (AoI 2)
$\mathcal{DCEC}^*$
Deontic Cognitive Event Calculus
(with Castañeda's *)
$L_{\omega 1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I

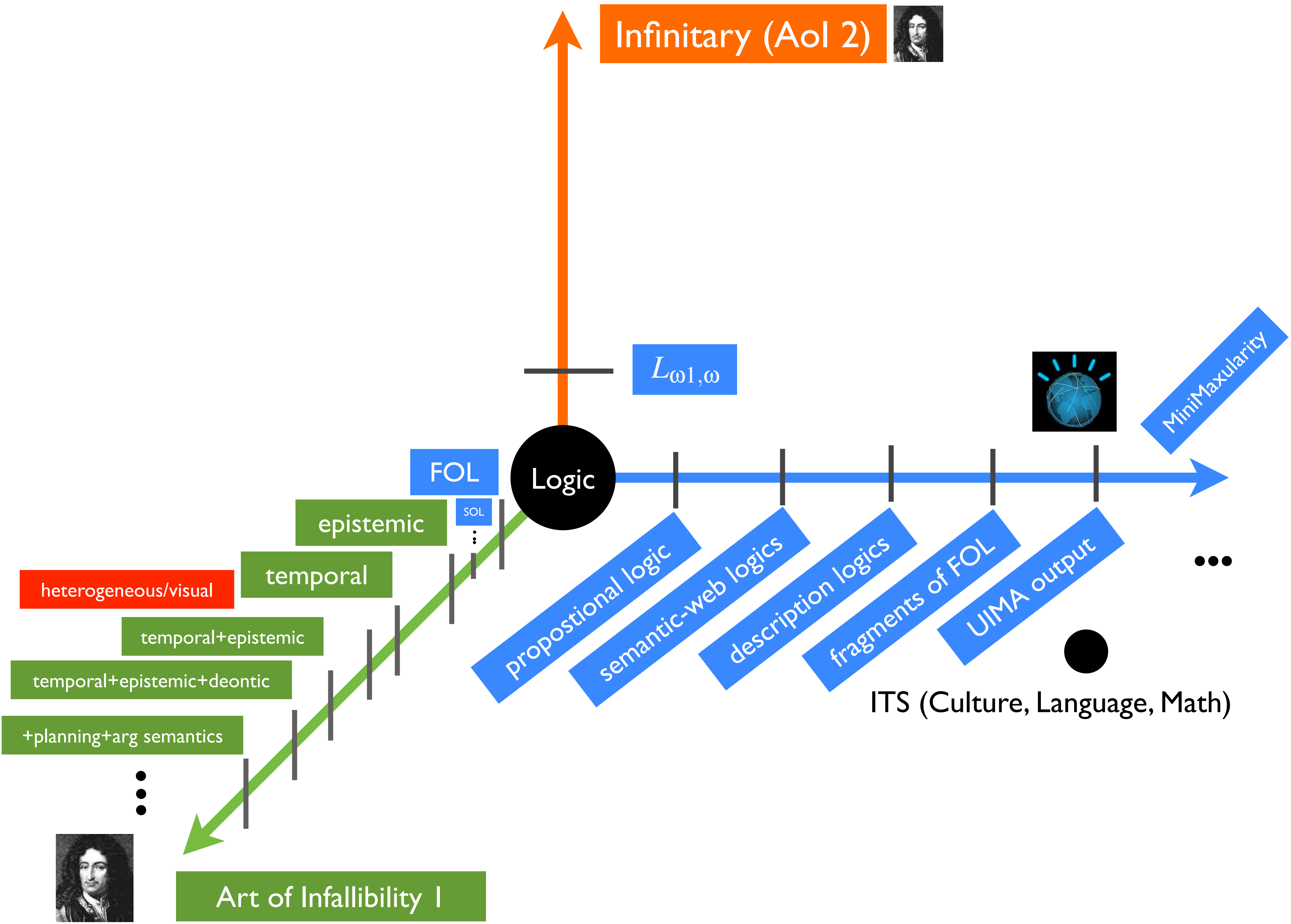


Infinitary (AoI 2)
$L_{\omega 1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I

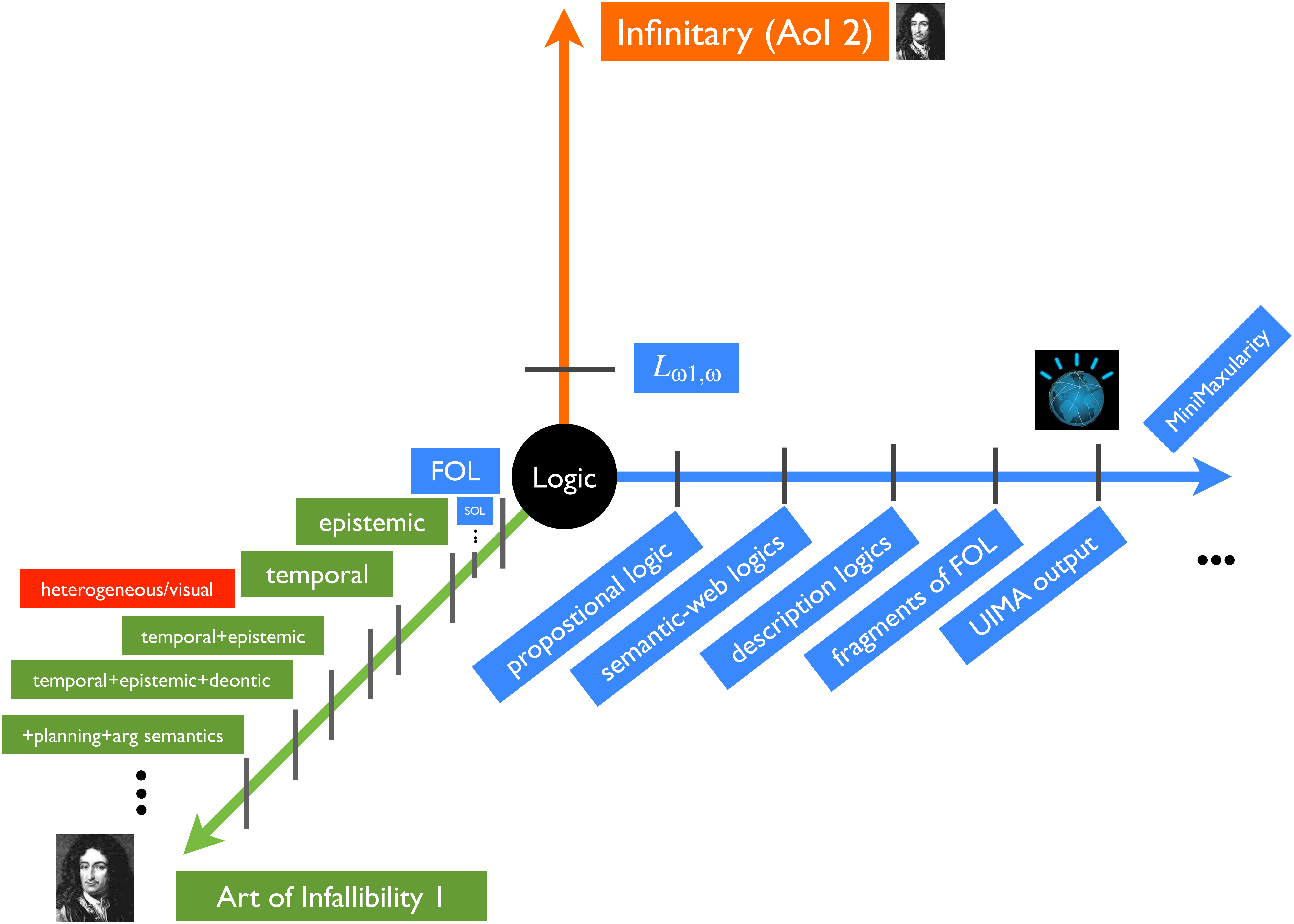


Infinitary (AoI 2)
$L_{\omega 1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Gödel's "God Theorem"
Art of Infallibility I

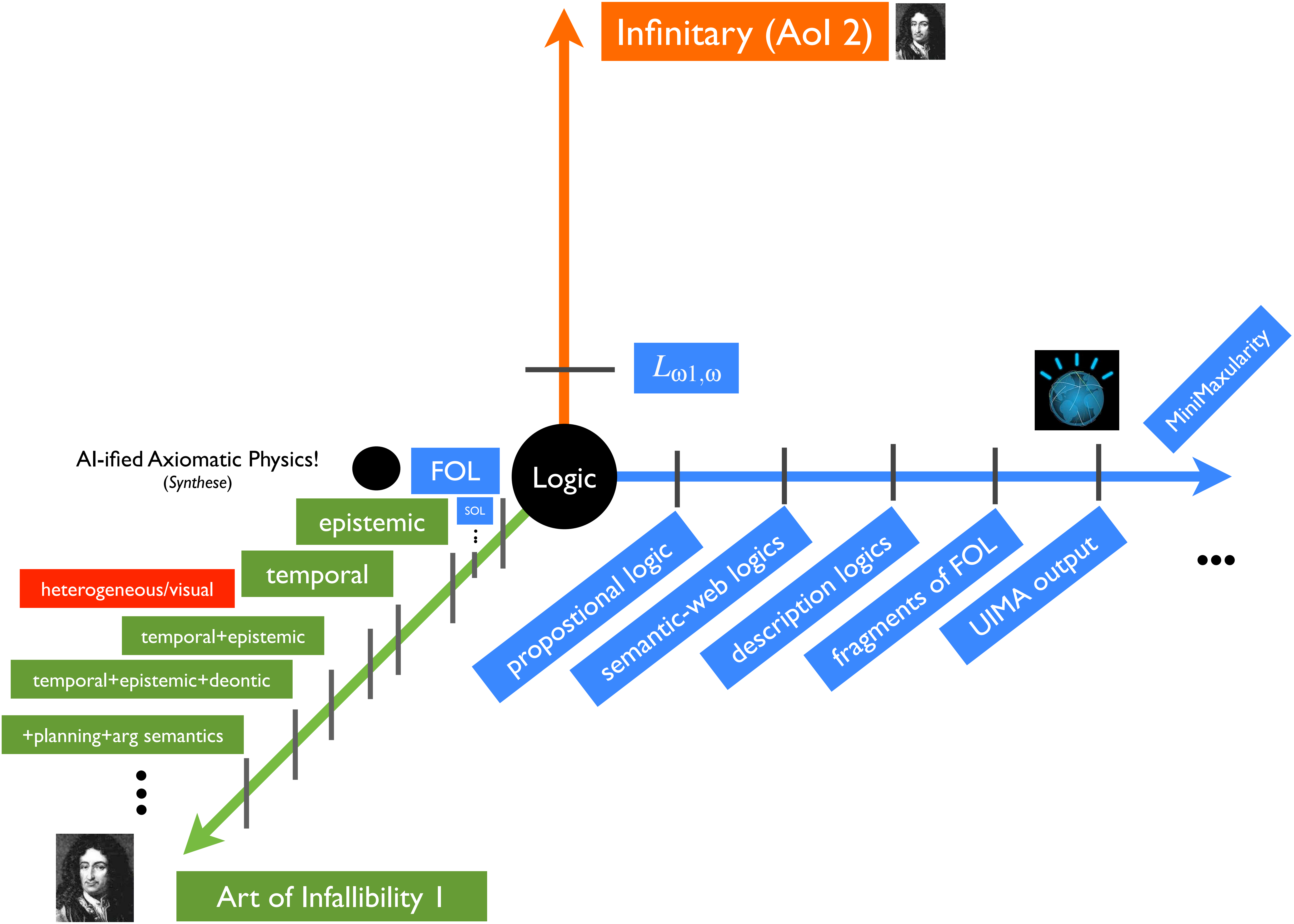

Infinitary (AoI 2)
$L_{\omega1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I



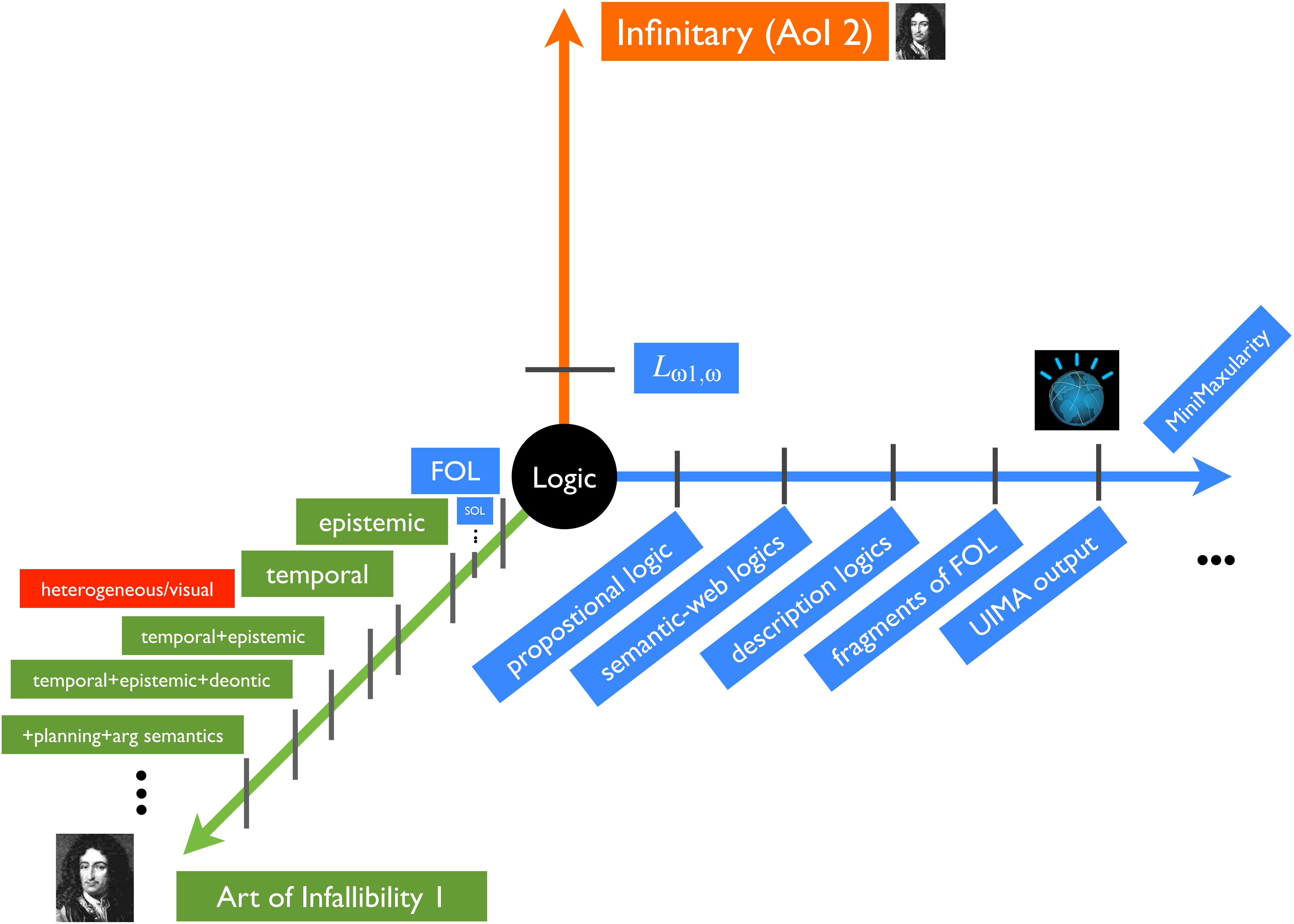
Infinitary (AoI 2)
$L_{\omega1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
ITS (Culture, Language, Math)
Art of Infallibility I



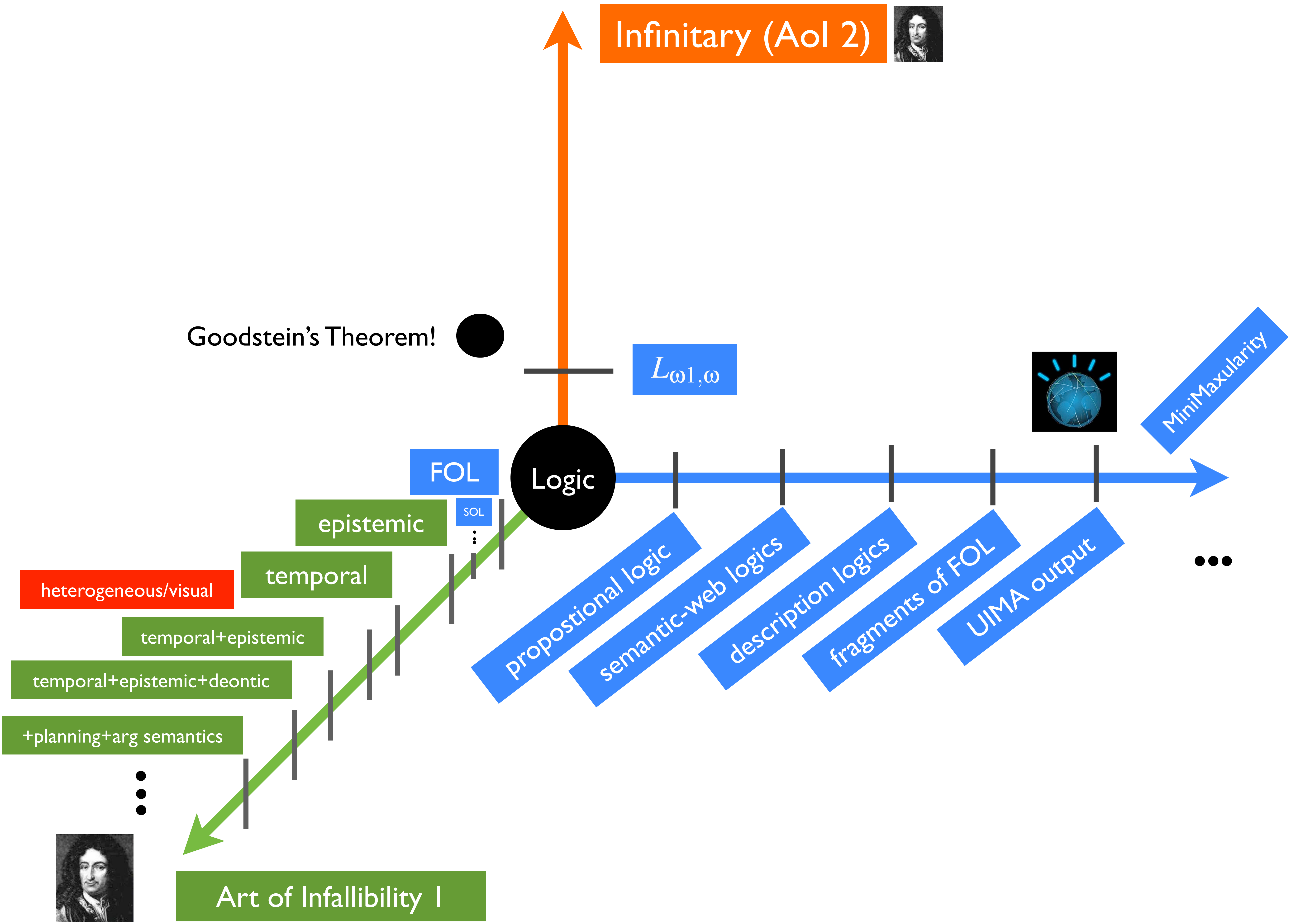
Infinitary (AoI 2)
$L_{\omega 1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I

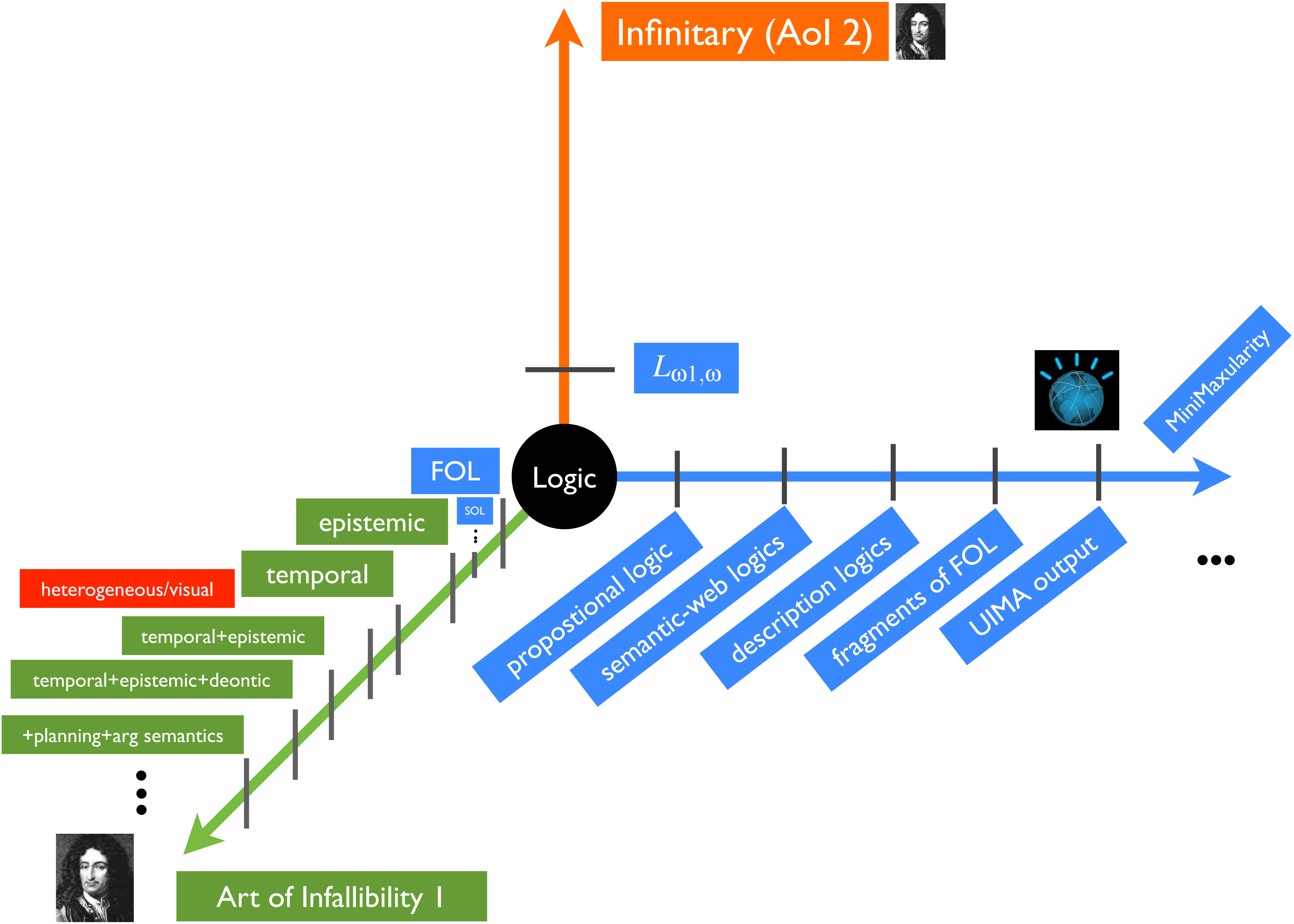


Infinitary (AoI 2)
$L_{\omega1,\omega}$
MiniMaxularity
AI-ified Axiomatic Physics!
(*Synthese*)
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I

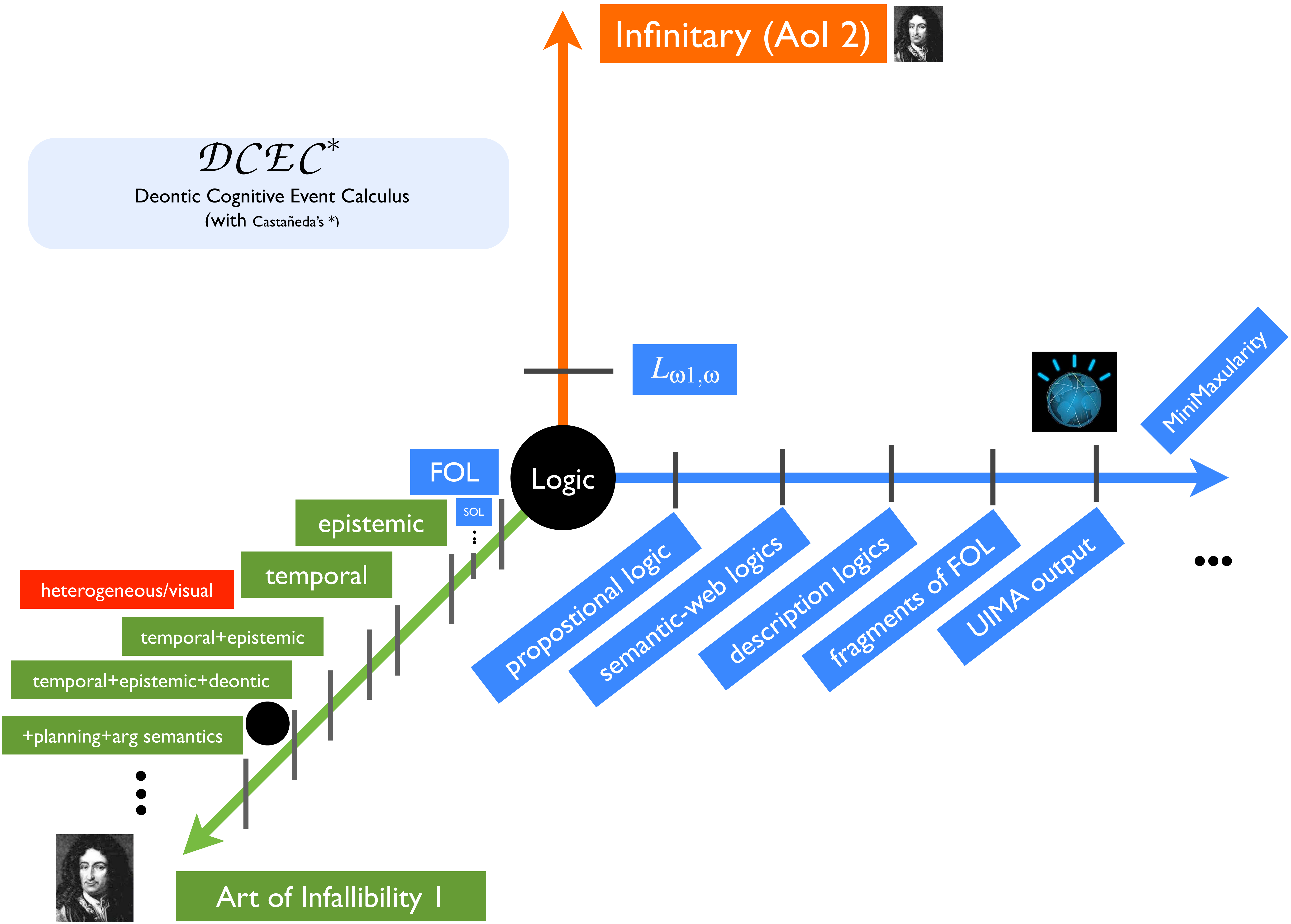


Infinitary (AoI 2)
$L_{\omega 1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I

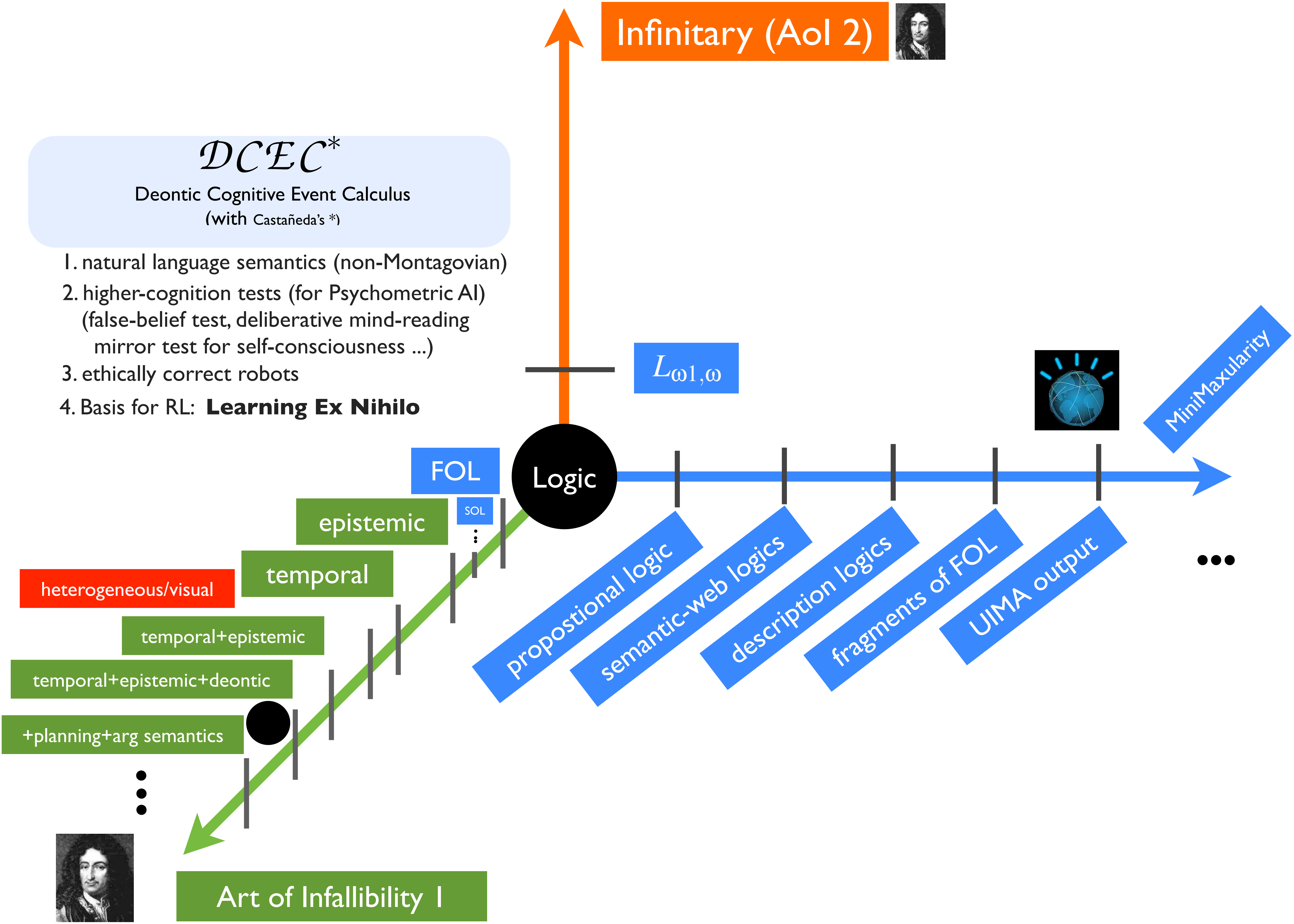

Infinitary (AoI 2)
Goodstein's Theorem!
$L_{\omega 1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I



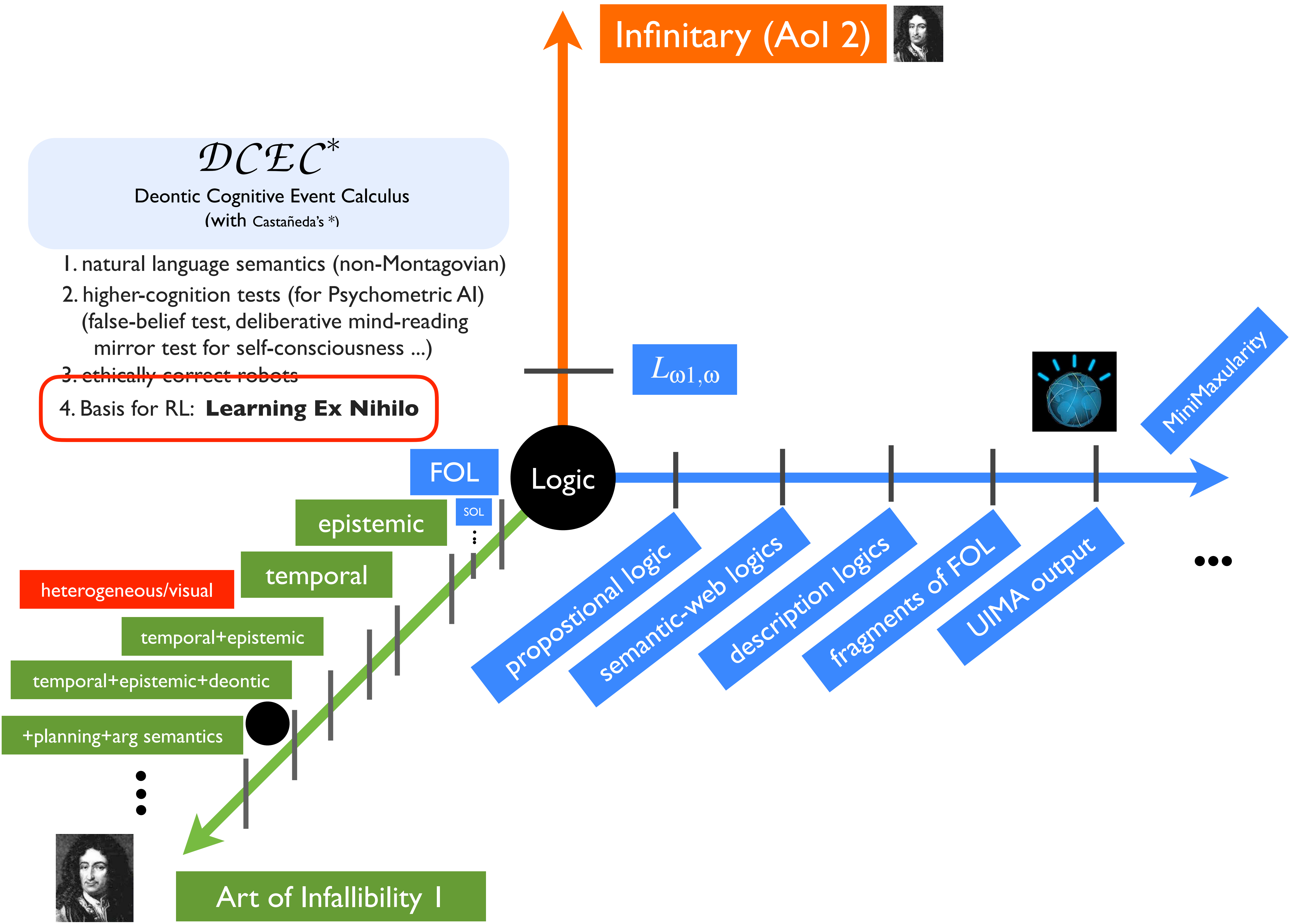
Infinitary (AoI 2)
$L_{\omega 1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I



Infinitary (AoI 2)
$\mathcal{DCEC}^*$
Deontic Cognitive Event Calculus
(with Castañeda's *)
$L_{\omega 1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I

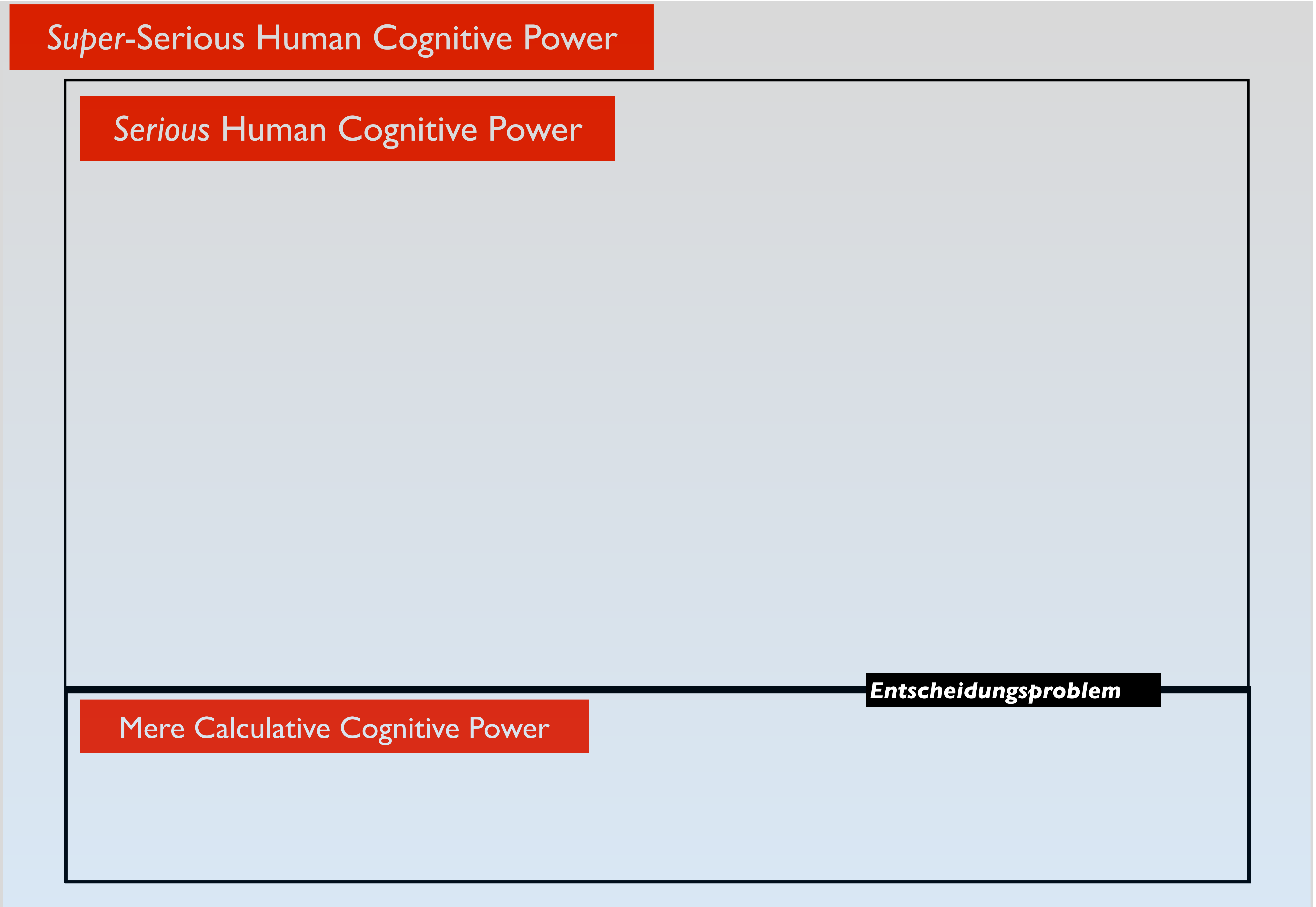


Infinitary (AoI 2)
$\mathcal{DCEC}^*$
Deontic Cognitive Event Calculus
(with Castañeda's *)
1. natural language semantics (non-Montagovian)
2. higher-cognition tests (for Psychometric AI)
(false-belief test, deliberative mind-reading
mirror test for self-consciousness ...)
3. ethically correct robots
4. Basis for RL: **Learning Ex Nihilo**
$L_{\omega 1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
...
Art of Infallibility I

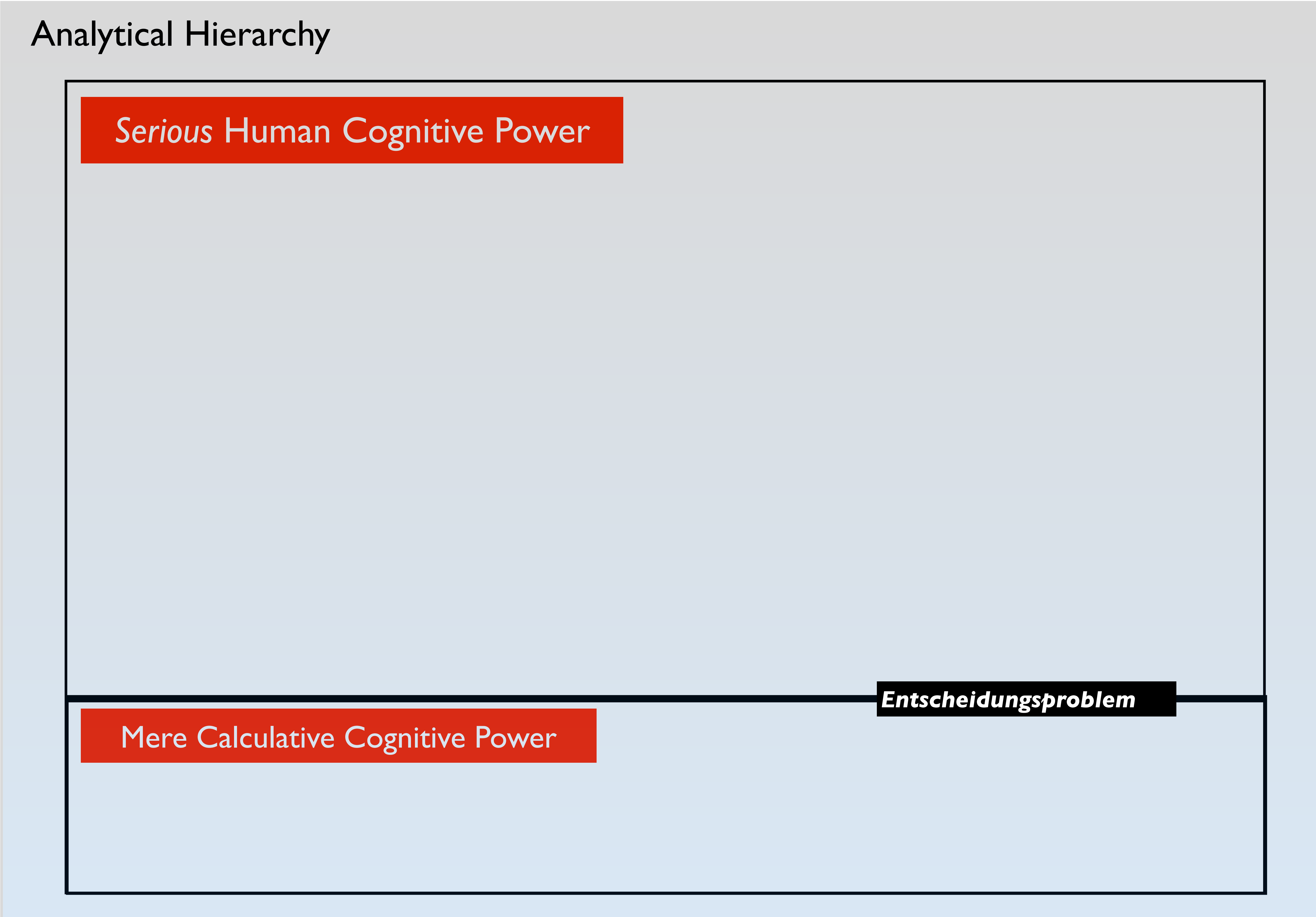


Infinitary (AoI 2)
$\mathcal{DCEC}^*$
Deontic Cognitive Event Calculus
(with Castañeda's *)
1. natural language semantics (non-Montagovian)
2. higher-cognition tests (for Psychometric AI)
(false-belief test, deliberative mind-reading
mirror test for self-consciousness ...)
~~3. ethically correct robots~~
4. Basis for RL: **Learning Ex Nihilo**
$L_{\omega1,\omega}$
MiniMaxularity
FOL
Logic
SOL
epistemic
temporal
heterogeneous/visual
temporal+epistemic
temporal+epistemic+deontic
+planning+arg semantics
propostional logic
semantic-web logics
description logics
fragments of FOL
UIMA output
Art of Infallibility I

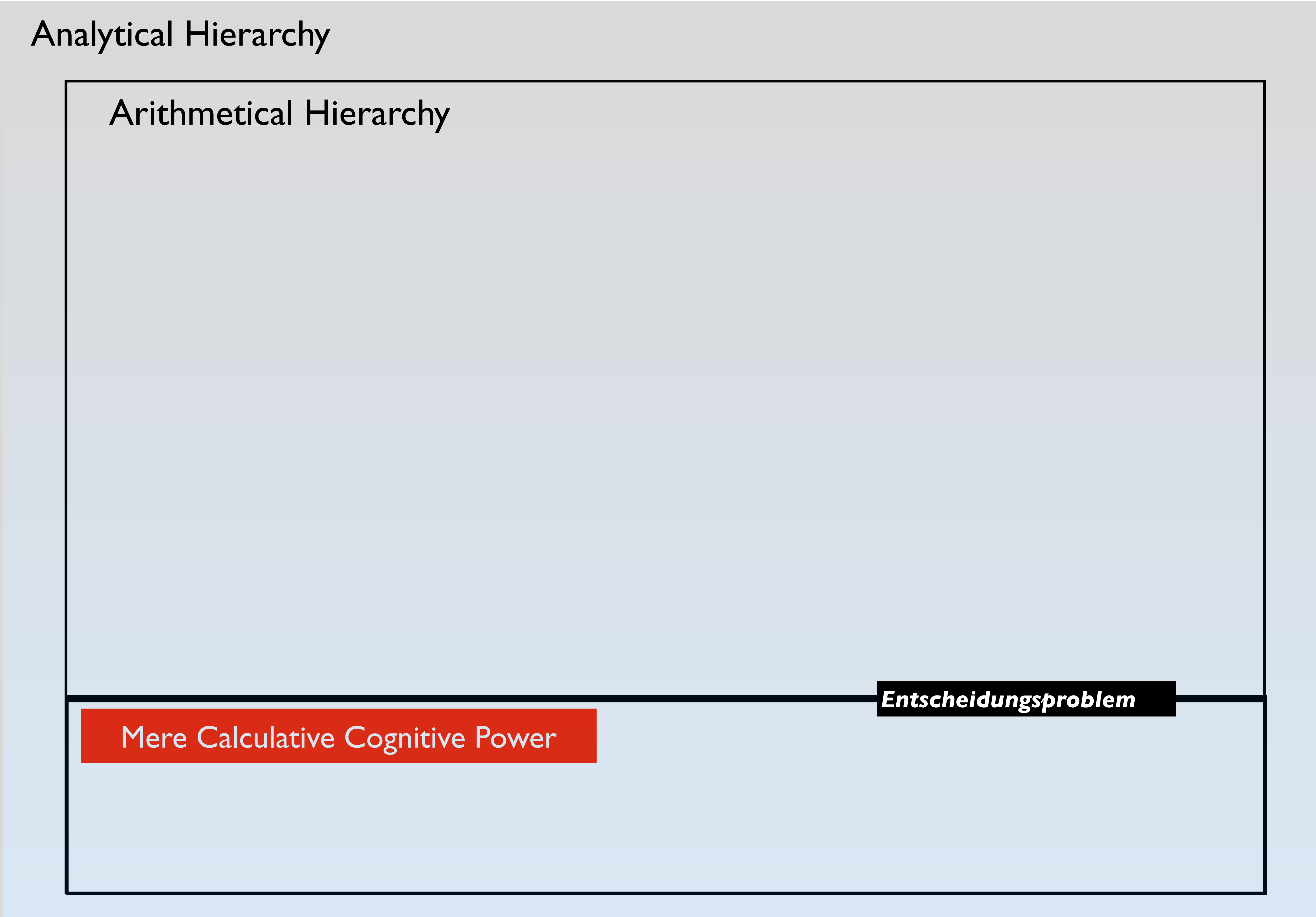
Animal-Level AI



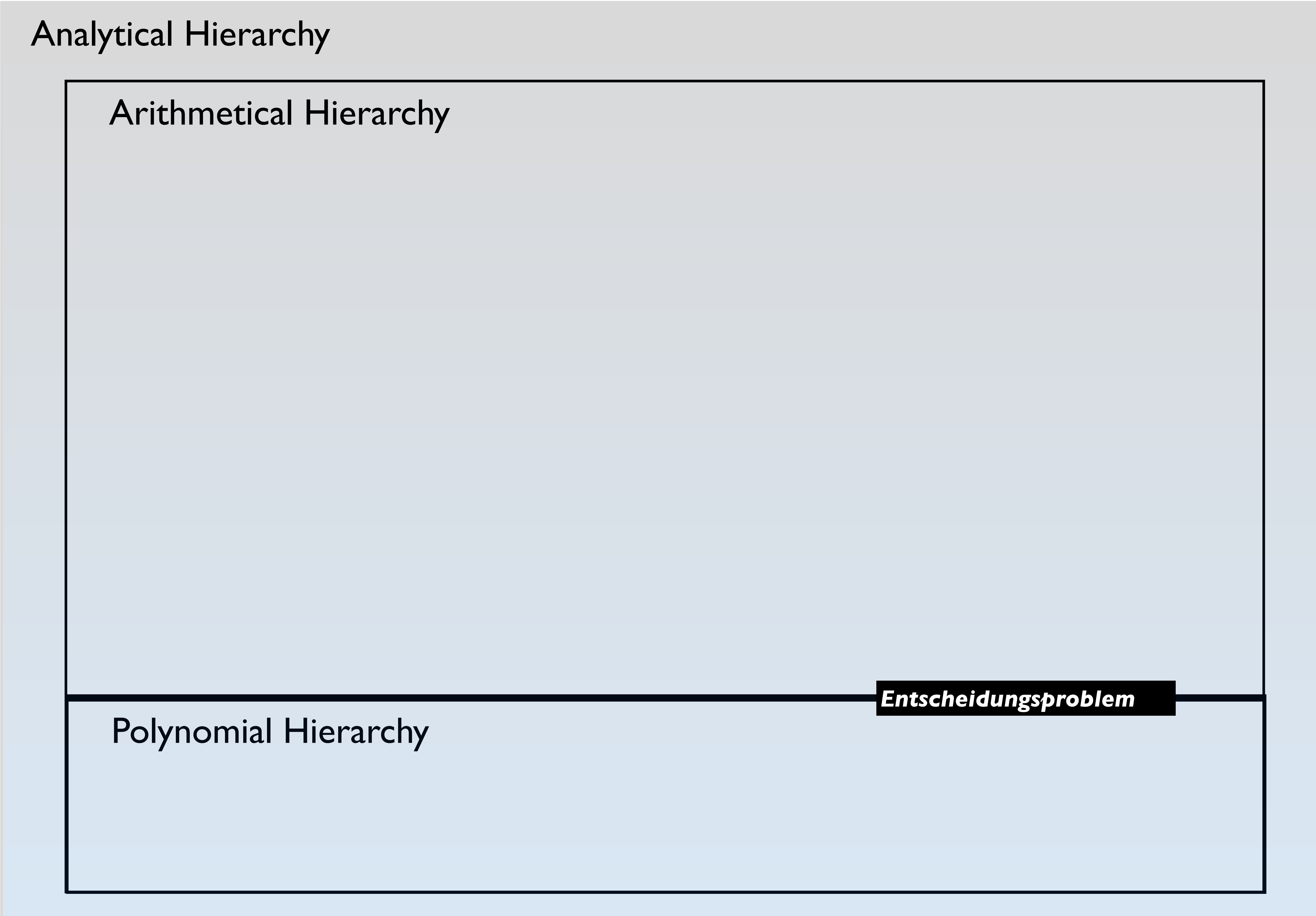
Animal-Level AI



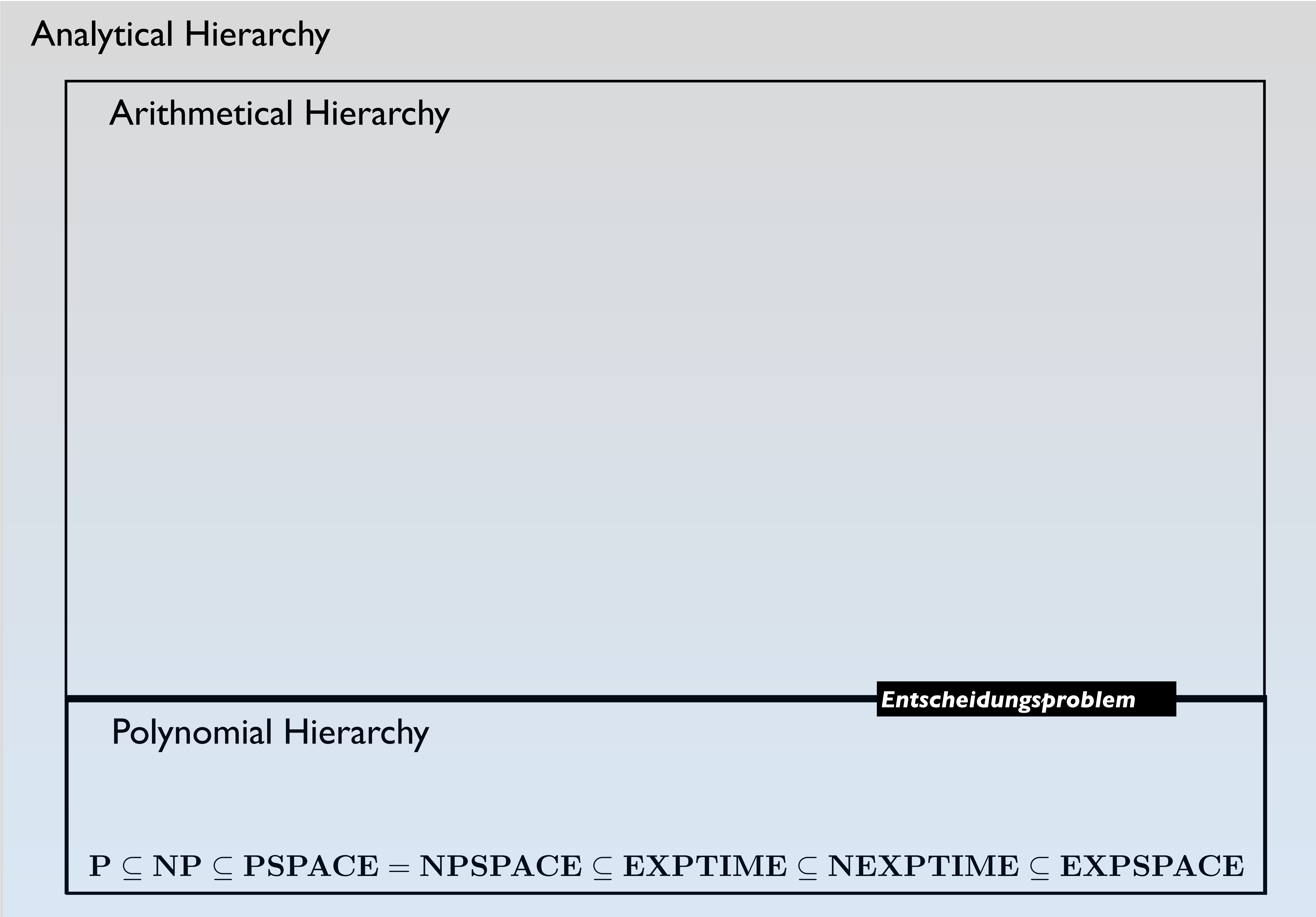
Animal-Level AI



Animal-Level AI



Animal-Level AI



Animal-Level AI

Analytical Hierarchy

Arithmetical Hierarchy

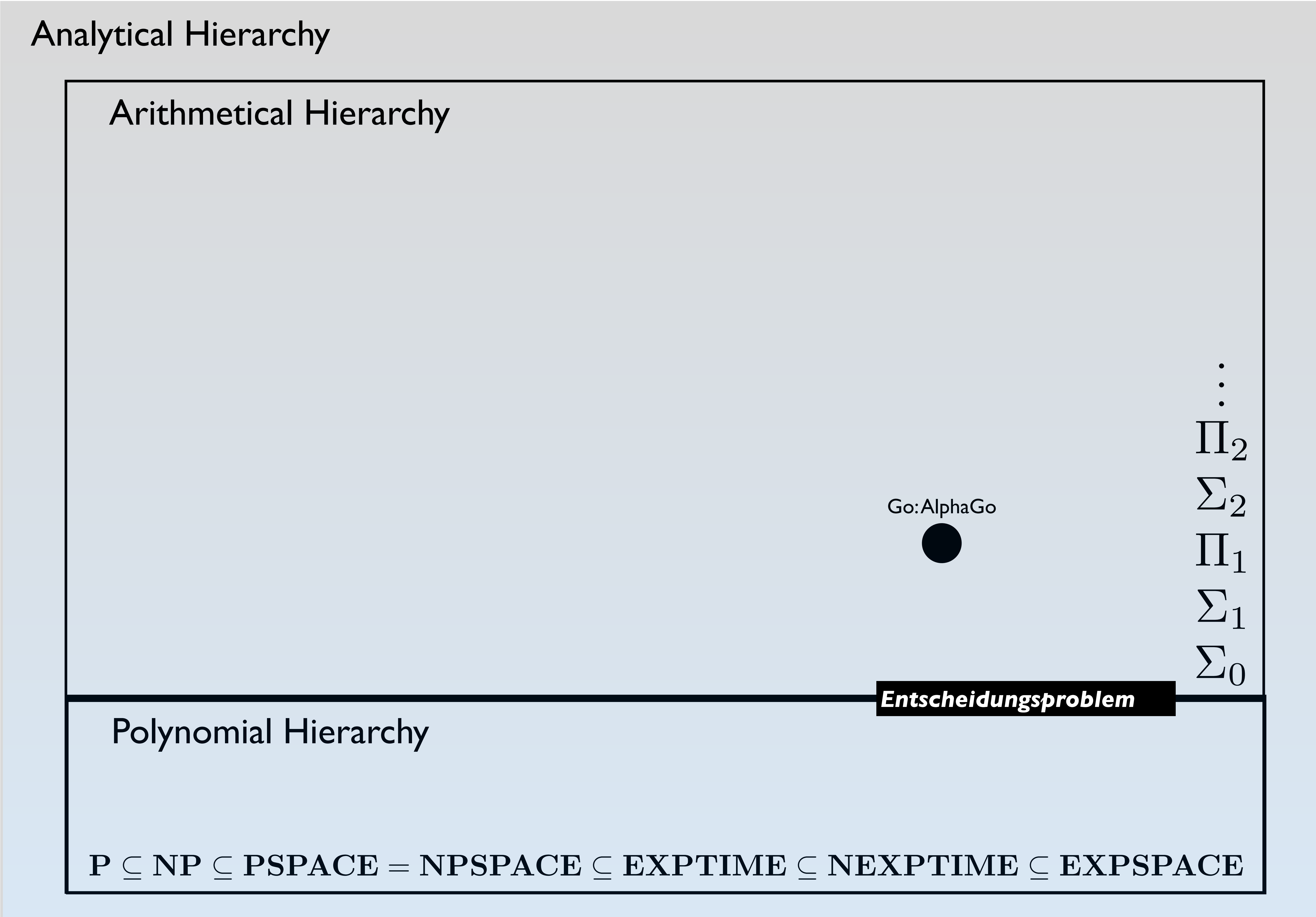
$$\vdots$$
$$\Pi_2$$
$$\Sigma_2$$
$$\Pi_1$$
$$\Sigma_1$$
$$\Sigma_0$$

Entscheidungsproblem

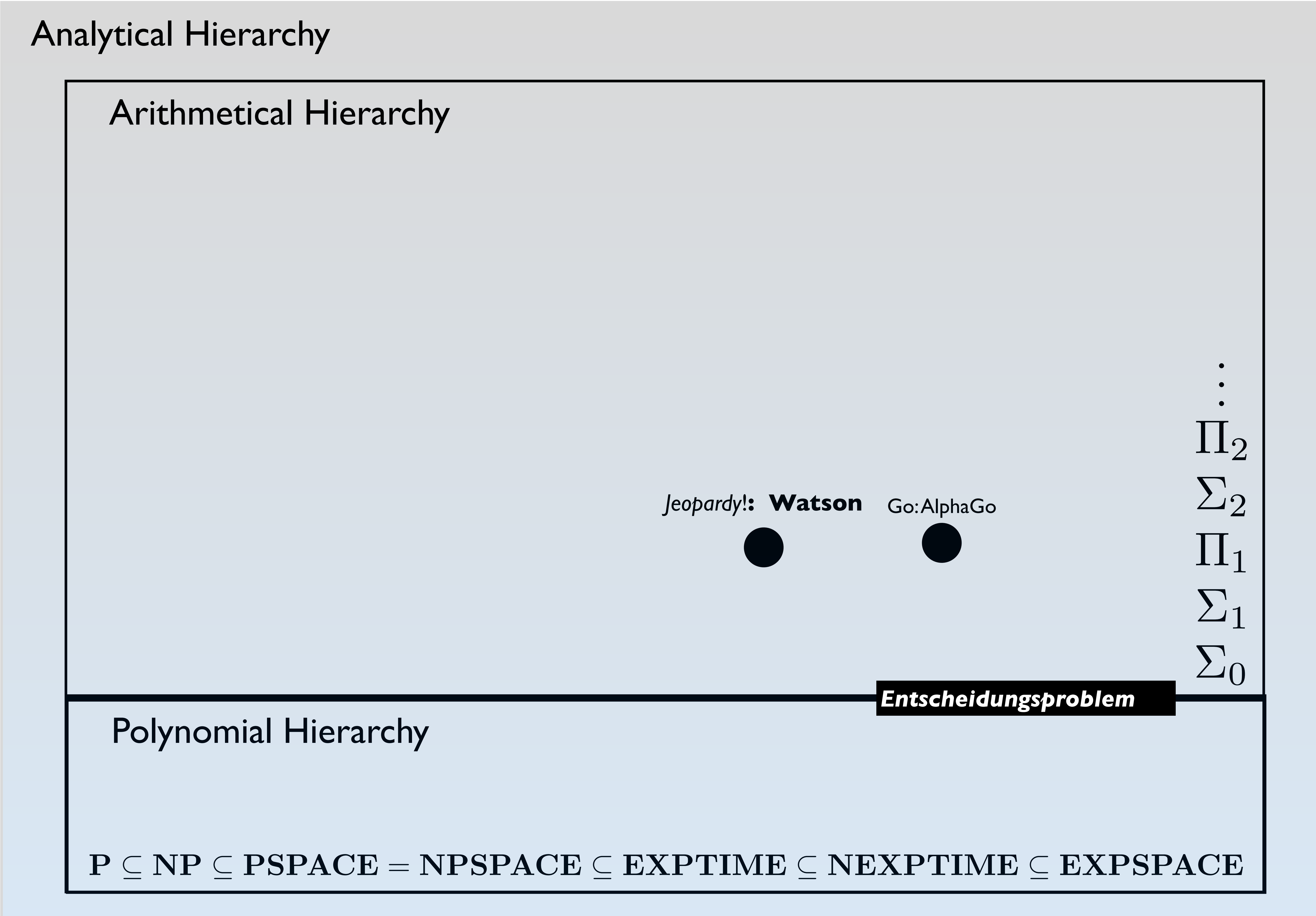
Polynomial Hierarchy

$$\mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE} = \mathbf{NPSPACE} \subseteq \mathbf{EXPTIME} \subseteq \mathbf{NEXPTIME} \subseteq \mathbf{EXPSPACE}$$

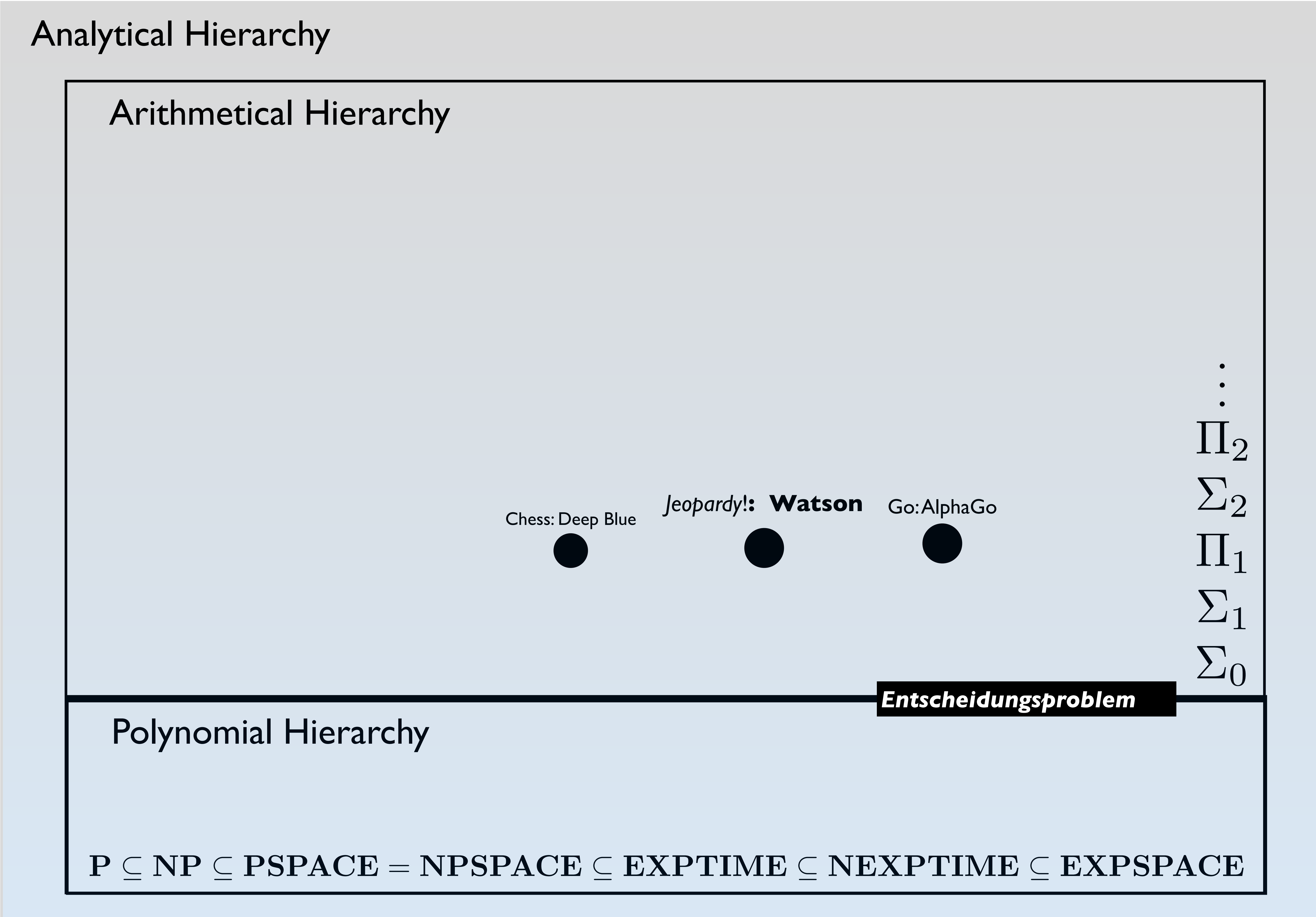
Animal-Level AI



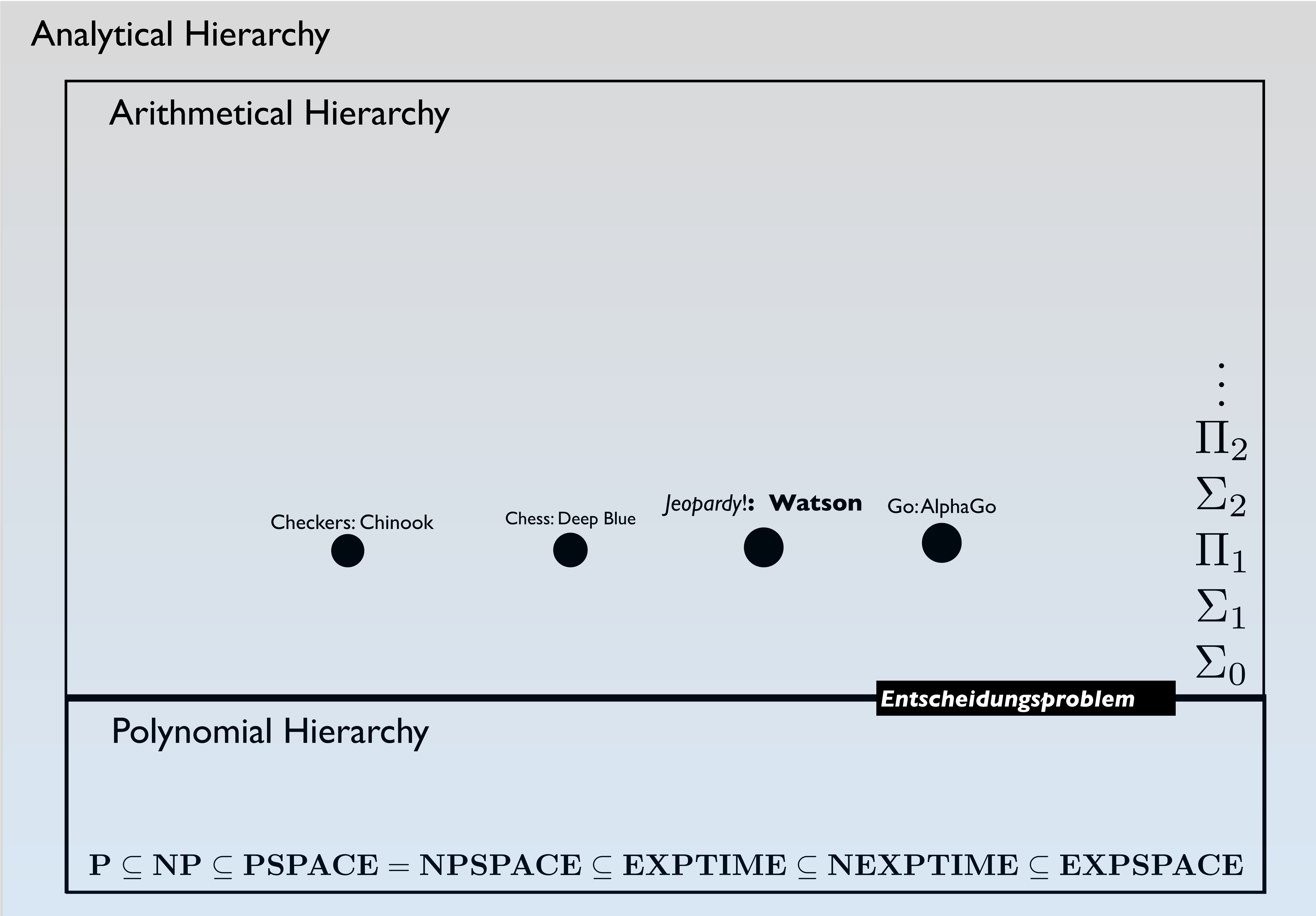
Animal-Level AI



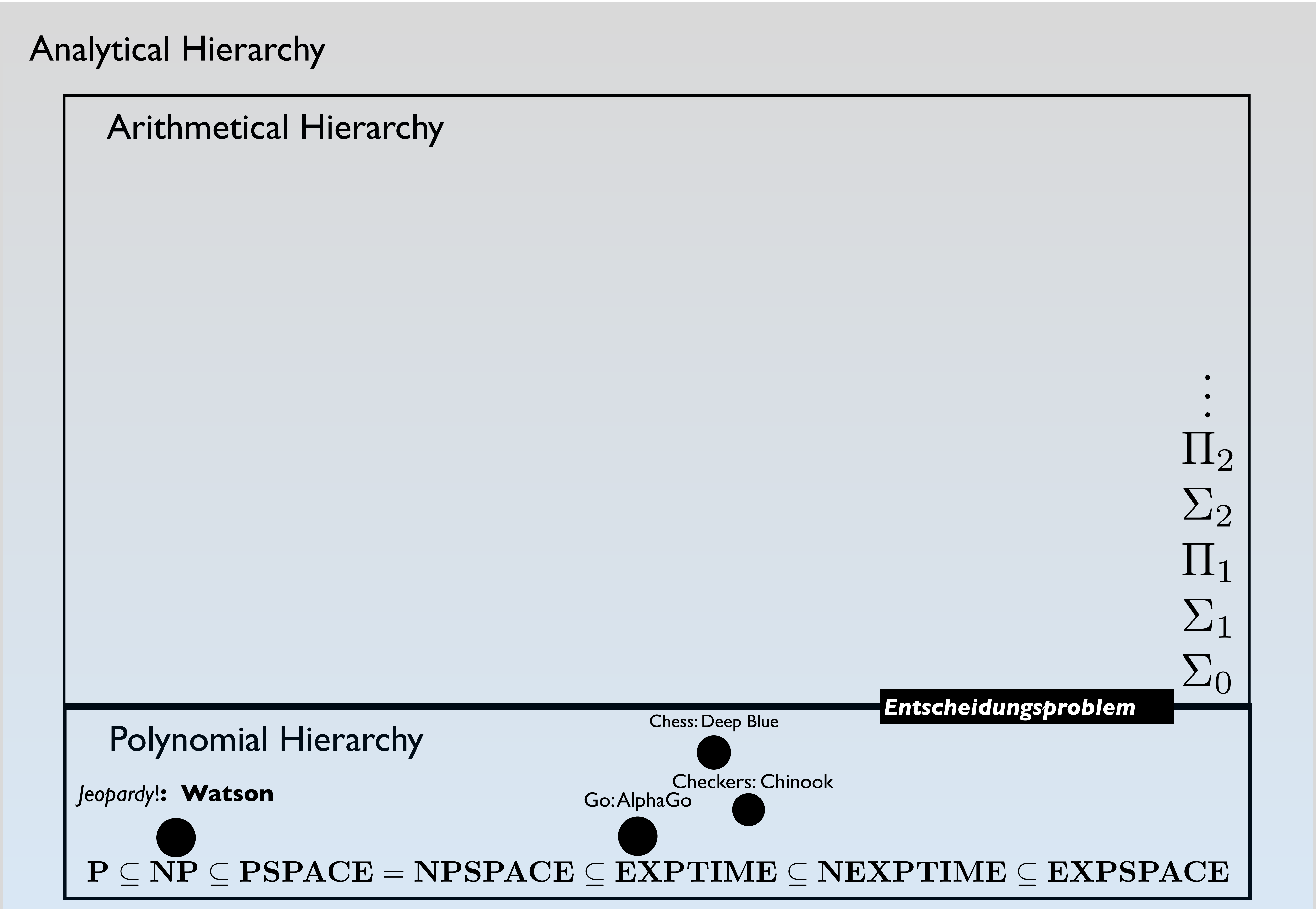
Animal-Level AI



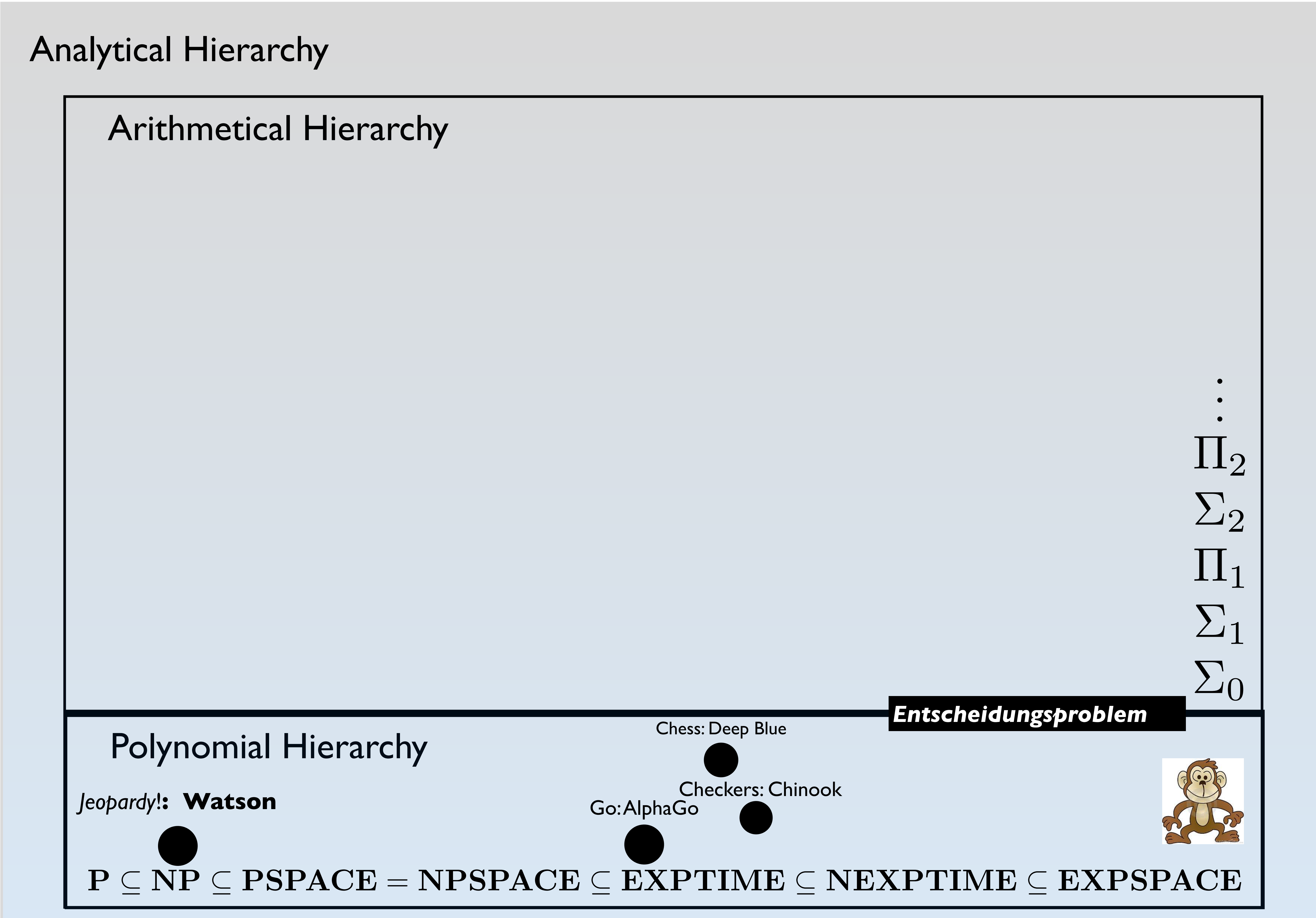
Animal-Level AI



Animal-Level AI



Animal-Level AI



Animal-Level AI

Analytical Hierarchy

Arithmetical Hierarchy

Church

$\vdots$

Π_2

Σ_2

Π_1

Σ_1

Σ_0

Entscheidungsproblem

Polynomial Hierarchy

Chess: Deep Blue

Checkers: Chinook

Go: AlphaGo

Jeopardy!: **Watson**

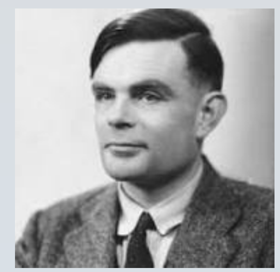
$$\mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE} = \mathbf{NPSPACE} \subseteq \mathbf{EXPTIME} \subseteq \mathbf{NEXPTIME} \subseteq \mathbf{EXPSPACE}$$

Animal-Level AI

Analytical Hierarchy

Arithmetical Hierarchy

Church



Turing

$\vdots$

Π_2

Σ_2

Π_1

Σ_1

Σ_0

Entscheidungsproblem

Polynomial Hierarchy

Chess: Deep Blue

Checkers: Chinook

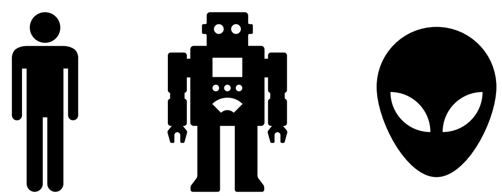
Go: AlphaGo

Jeopardy!: **Watson**

$$\mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE} = \mathbf{NPSPACE} \subseteq \mathbf{EXPTIME} \subseteq \mathbf{NEXPTIME} \subseteq \mathbf{EXPSPACE}$$

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)

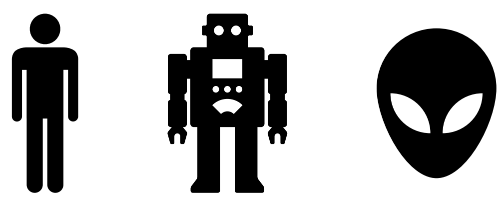


$$
\begin{tikzcd}
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)

We say that a relation $R(u, y_1, \ldots, y_n)$ is polytime iff there is a deterministic Turing Machine $\mathfrak{m}$ and a polynomial p s.t. $\mathfrak{m}$ decides this relation in $p(|u|)$.

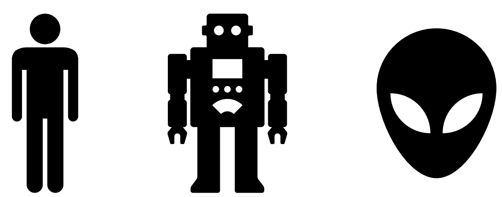


$$
\begin{tikzcd}
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)

We say that a relation $R(u, y_1, \ldots, y_n)$ is polytime iff there is a deterministic Turing Machine $\mathfrak{m}$ and a polynomial p s.t. $\mathfrak{m}$ decides this relation in $p(|u|)$.

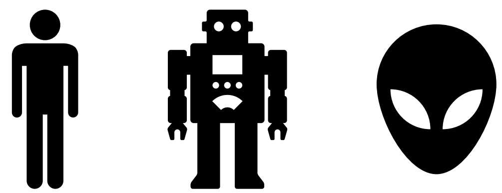


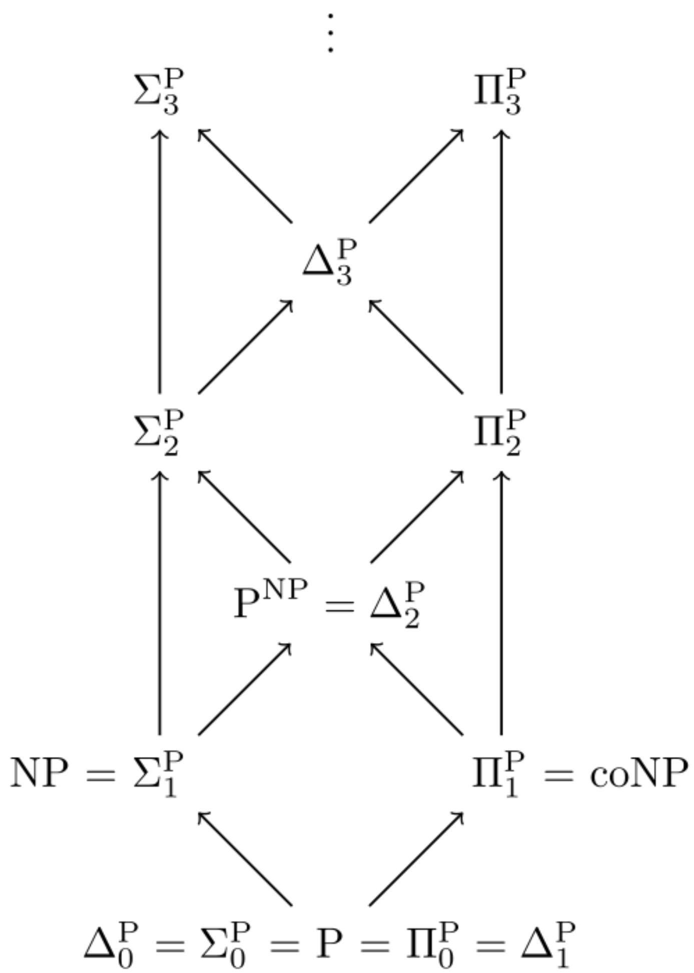
$L \in \mathbf{NP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\exists y R(u, y)$.

$$\begin{tikzcd}
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)





We say that a relation $R(u, y_1, \ldots, y_n)$ is polytime iff there is a deterministic Turing Machine $\mathfrak{m}$ and a polynomial p s.t. $\mathfrak{m}$ decides this relation in $p(|u|)$.

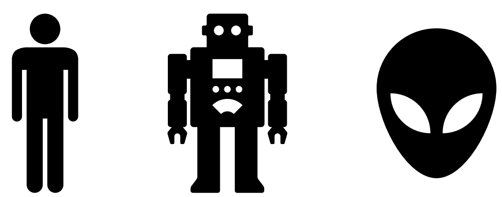
$L \in \mathbf{NP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\exists y R(u, y)$.

E.g.: We can prove $\mathbf{SAT} \in \mathbf{NP}$ because we have a polytime relation R s.t. $\phi \in \mathbf{SAT}$ iff $\exists y R(\phi \in \mathscr{L}_{pc}, \langle$assignments to Boolean vars$\rangle)$, where these assignments produce truth.

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)

We say that a relation $R(u, y_1, \ldots, y_n)$ is polytime iff there is a deterministic Turing Machine $\mathfrak{m}$ and a polynomial p s.t. $\mathfrak{m}$ decides this relation in $p(|u|)$.



$$
\begin{tikzcd}[column sep=small]
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

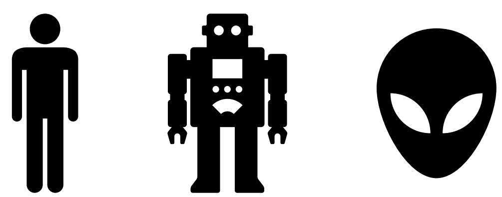
$L \in \mathbf{NP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\exists y R(u, y)$.

E.g.: We can prove $\mathbf{SAT} \in \mathbf{NP}$ because we have a polytime relation R s.t. $\phi \in \mathbf{SAT}$ iff $\exists y R(\phi \in \mathscr{L}_{pc}, \langle$assignments to Boolean vars$\rangle)$, where these assignments produce truth.

$L \in \mathbf{coNP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\forall y R(u, y)$.

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)



$$\begin{tikzcd}
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

We say that a relation $R(u, y_1, \ldots, y_n)$ is polytime iff there is a deterministic Turing Machine $\mathfrak{m}$ and a polynomial p s.t. $\mathfrak{m}$ decides this relation in $p(|u|)$.

$L \in \mathbf{NP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\exists y R(u, y)$.

E.g.: We can prove $\mathbf{SAT} \in \mathbf{NP}$ because we have a polytime relation R s.t. $\phi \in \mathbf{SAT}$ iff $\exists y R(\phi \in \mathscr{L}_{pc}, \langle$assignments to Boolean vars$\rangle)$, where these assignments produce truth.

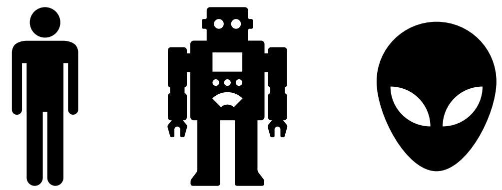
$L \in \mathbf{coNP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\forall y R(u, y)$.

To prove $\mathbf{coSAT} \in \mathbf{coNP}$, we note that we have a polytime relation R s.t. $\phi \in \mathbf{coSAT}$ iff $\forall y R(\phi \in \mathscr{L}_{pc}, \langle$assignments to Boolean vars$\rangle)$, where the assignments produce *falsity*.

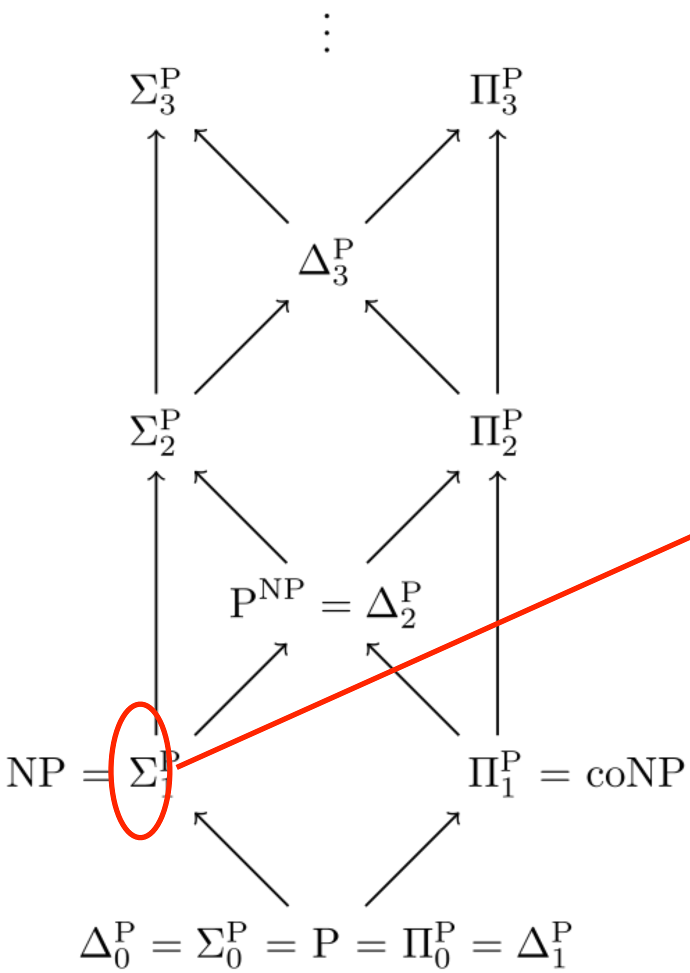
Polynomial Hierarchy, Part I

(via formal logic, directly; a start)

We say that a relation $R(u, y_1, \ldots, y_n)$ is polytime iff there is a deterministic Turing Machine $\mathfrak{m}$ and a polynomial p s.t. $\mathfrak{m}$ decides this relation in $p(|u|)$.



$L \in \mathbf{NP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\exists y R(u, y)$.



E.g.: We can prove $\mathbf{SAT} \in \mathbf{NP}$ because we have a polytime relation R s.t. $\phi \in \mathbf{SAT}$ iff $\exists y R(\phi \in \mathscr{L}_{pc}, \langle$assignments to Boolean vars$\rangle)$, where these assignments produce truth.

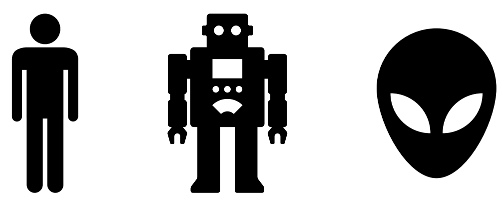
$L \in \mathbf{coNP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\forall y R(u, y)$.

To prove $\mathbf{coSAT} \in \mathbf{coNP}$, we note that we have a polytime relation R s.t. $\phi \in \mathbf{coSAT}$ iff $\forall y R(\phi \in \mathscr{L}_{pc}, \langle$assignments to Boolean vars$\rangle)$, where the assignments produce *falsity*.

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)

We say that a relation $R(u, y_1, \ldots, y_n)$ is polytime iff there is a deterministic Turing Machine $\mathfrak{m}$ and a polynomial p s.t. $\mathfrak{m}$ decides this relation in $p(|u|)$.



$$\begin{tikzcd}
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

$L \in \mathbf{NP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\exists y R(u, y)$.

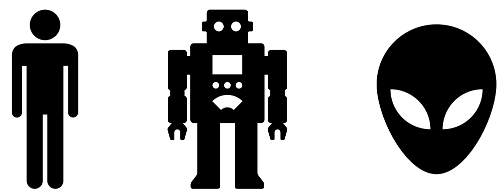
E.g.: We can prove $\mathbf{SAT} \in \mathbf{NP}$ because we have a polytime relation R s.t. $\phi \in \mathbf{SAT}$ iff $\exists y R(\phi \in \mathscr{L}_{pc}, \langle$assignments to Boolean vars$\rangle)$, where these assignments produce truth.

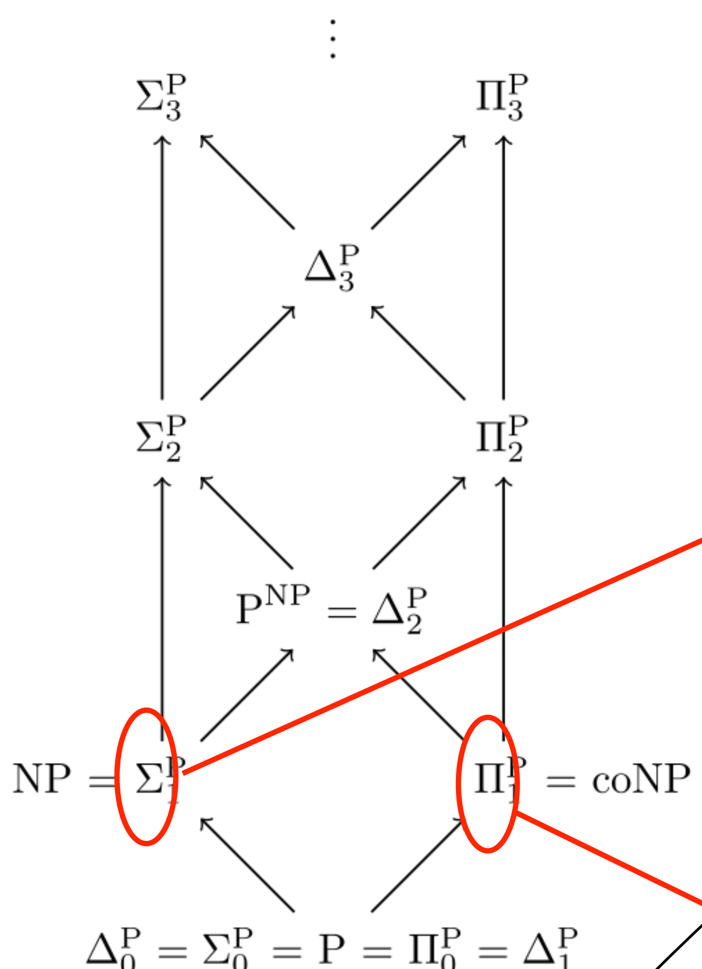
$L \in \mathbf{coNP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\forall y R(u, y)$.

To prove $\mathbf{coSAT} \in \mathbf{coNP}$, we note that we have a polytime relation R s.t. $\phi \in \mathbf{coSAT}$ iff $\forall y R(\phi \in \mathscr{L}_{pc}, \langle$assignments to Boolean vars$\rangle)$, where the assignments produce *falsity*.

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)





We say that a relation $R(u, y_1, \ldots, y_n)$ is polytime iff there is a deterministic Turing Machine $\mathfrak{m}$ and a polynomial p s.t. $\mathfrak{m}$ decides this relation in $p(|u|)$.

$L \in \mathbf{NP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\exists y R(u, y)$.

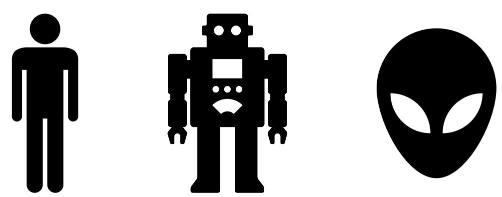
E.g.: We can prove $\mathbf{SAT} \in \mathbf{NP}$ because we have a polytime relation R s.t. $\phi \in \mathbf{SAT}$ iff $\exists y R(\phi \in \mathscr{L}_{pc}, \langle \text{assignments to Boolean vars} \rangle)$, where these assignments produce truth.

$L \in \mathbf{coNP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\forall y R(u, y)$.

To prove $\mathbf{coSAT} \in \mathbf{coNP}$, we note that we have a polytime relation R s.t. $\phi \in \mathbf{coSAT}$ iff $\forall y R(\phi \in \mathscr{L}_{pc}, \langle \text{assignments to Boolean vars} \rangle)$, where the assignments produce *falsity*.

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)



$$\begin{tikzcd}
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

We say that a relation $R(u, y_1, \ldots, y_n)$ is polytime iff there is a deterministic Turing Machine $\mathfrak{m}$ and a polynomial p s.t. $\mathfrak{m}$ decides this relation in $p(|u|)$.

$L \in \mathbf{NP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\exists y R(u, y)$.

E.g.: We can prove $\mathbf{SAT} \in \mathbf{NP}$ because we have a polytime relation R s.t. $\phi \in \mathbf{SAT}$ iff $\exists y R(\phi \in \mathscr{L}_{pc}, \langle \text{assignments to Boolean vars} \rangle)$, where these assignments produce truth.

(Or a truth-graph y in HS® with at least one open branch.)

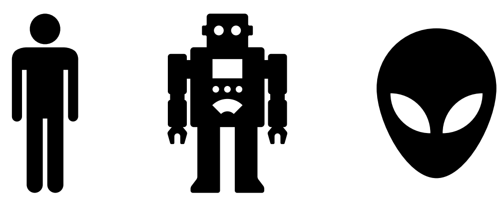
$L \in \mathbf{coNP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\forall y R(u, y)$.

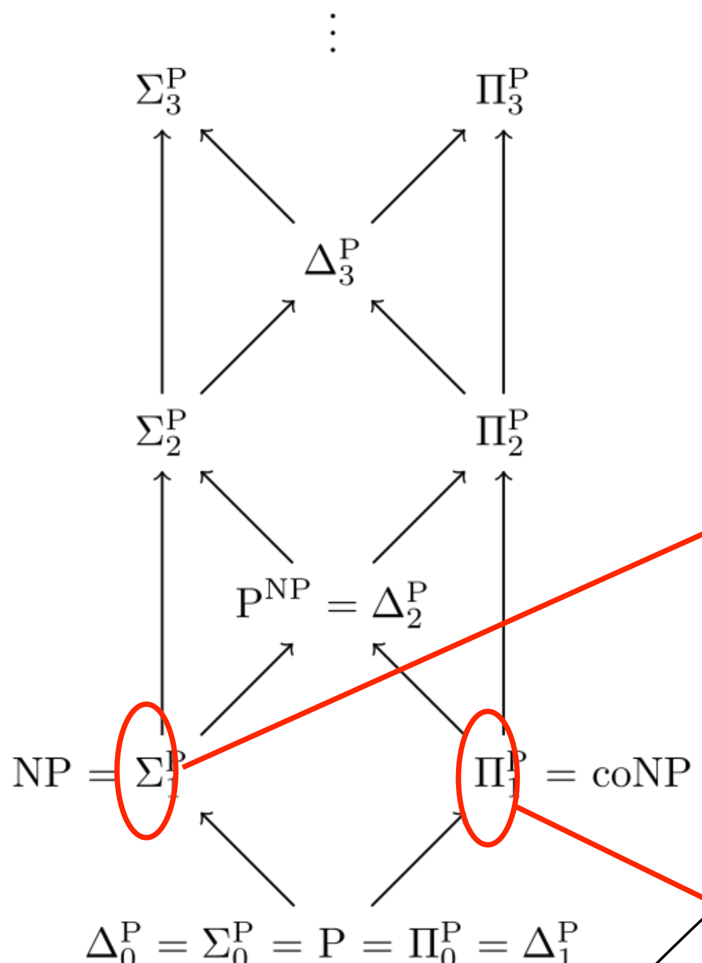
To prove $\mathbf{coSAT} \in \mathbf{coNP}$, we note that we have a polytime relation R s.t. $\phi \in \mathbf{coSAT}$ iff $\forall y R(\phi \in \mathscr{L}_{pc}, \langle \text{assignments to Boolean vars} \rangle)$, where the assignments produce *falsity*.

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)

We say that a relation $R(u, y_1, \ldots, y_n)$ is polytime iff there is a deterministic Turing Machine $\mathfrak{m}$ and a polynomial p s.t. $\mathfrak{m}$ decides this relation in $p(|u|)$.





$L \in \mathbf{NP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\exists y R(u, y)$.

E.g.: We can prove $\mathbf{SAT} \in \mathbf{NP}$ because we have a polytime relation R s.t. $\phi \in \mathbf{SAT}$ iff $\exists y R(\phi \in \mathscr{L}_{pc}, \langle\text{assignments to Boolean vars}\rangle)$, where these assignments produce truth.

$L \in \mathbf{coNP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\forall y R(u, y)$.

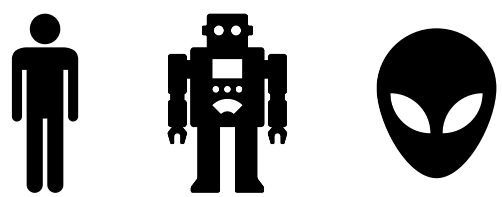
(Or a truth-graph y in HS® with at least one open branch.)

To prove $\mathbf{coSAT} \in \mathbf{coNP}$, we note that we have a polytime relation R s.t. $\phi \in \mathbf{coSAT}$ iff $\forall y R(\phi \in \mathscr{L}_{pc}, \langle\text{assignments to Boolean vars}\rangle)$, where the assignments produce *falsity*.

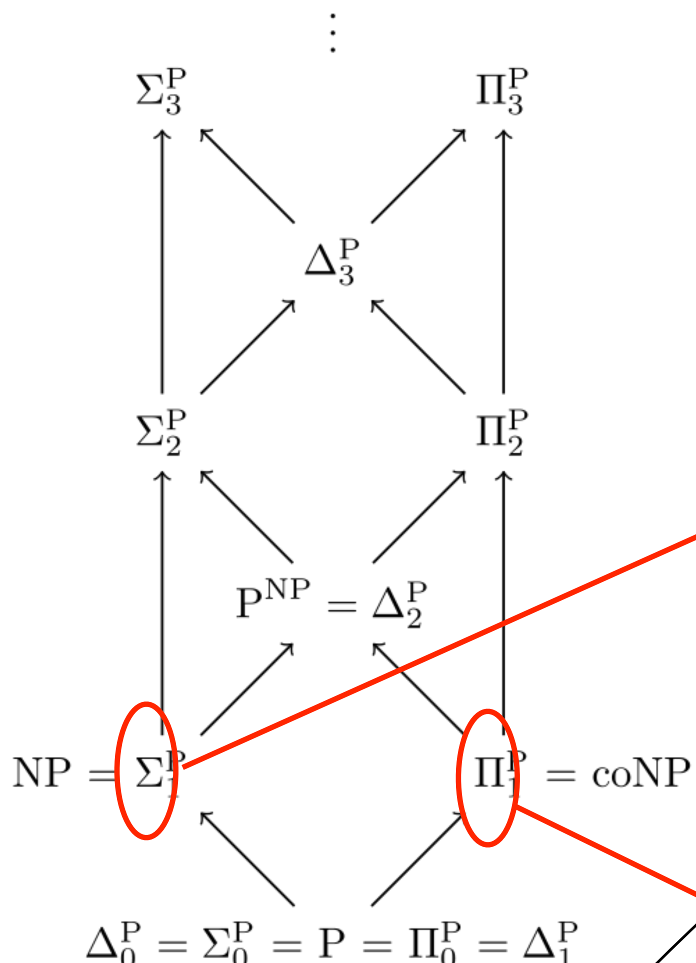
Polynomial Hierarchy, Part I

(via formal logic, directly; a start)

We say that a relation $R(u, y_1, \ldots, y_n)$ is polytime iff there is a deterministic Turing Machine $\mathfrak{m}$ and a polynomial p s.t. $\mathfrak{m}$ decides this relation in $p(|u|)$.



$L \in \mathbf{NP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\exists y R(u, y)$.



E.g.: We can prove $\mathbf{SAT} \in \mathbf{NP}$ because we have a polytime relation R s.t. $\phi \in \mathbf{SAT}$ iff $\exists y R(\phi \in \mathscr{L}_{pc}, \langle\text{assignments to Boolean vars}\rangle)$, where these assignments produce truth.

$L \in \mathbf{coNP}$ iff: there's a polytime relation R s.t. $u \in L$ iff $\forall y R(u, y)$.

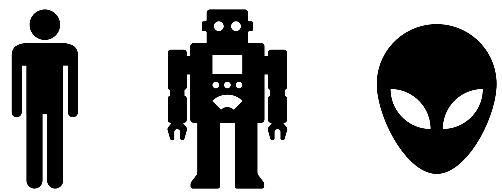
(Or a truth-graph *y* in HS® with at least one open branch.)

To prove $\mathbf{coSAT} \in \mathbf{coNP}$, we note that we have a polytime relation R s.t. $\phi \in \mathbf{coSAT}$ iff $\forall y R(\phi \in \mathscr{L}_{pc}, \langle\text{assignments to Boolean vars}\rangle)$, where the assignments produce *falsity*.

(Or every truth-graph *y* in HS® has no open branch.)

Polynomial Hierarchy, Part I

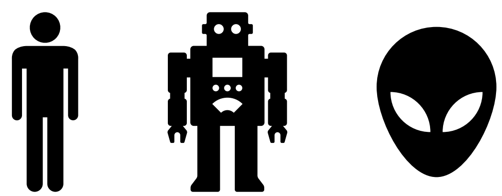
(via formal logic, directly; a start)



$$\begin{tikzcd}[column sep=small]
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)

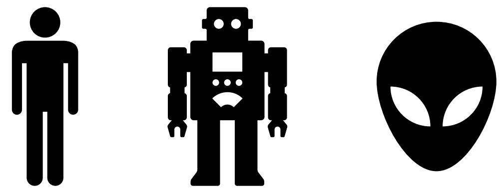


$$\begin{tikzcd}
& \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
& \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
& \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
& \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

"What's that Δ??"

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)

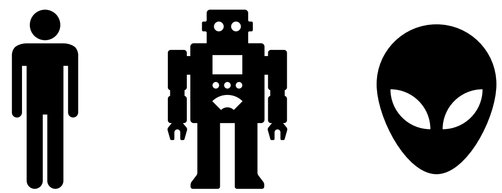


$$
\begin{tikzcd}[column sep=small]
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

"What's that Δ??"

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)

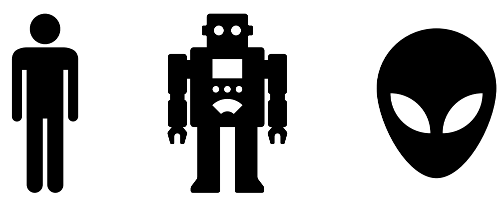


$$
\begin{tikzcd}
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

“What’s that Δ??”

Polynomial Hierarchy, Part I

(via formal logic, directly; a start)



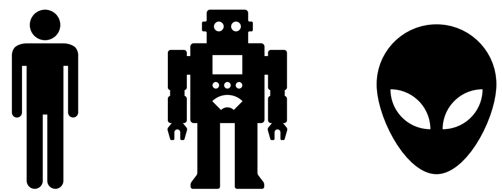
$$\begin{tikzcd}[column sep=small]
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

"What's that Δ??"

Ah. This is *not* a direct analogue to the AH. The arrows going up do indicate containment, but the purely "logicist" notation based on quantifiers is apparently mixed here (dangerously). The "Delta notation" is the oracle approach to building up PH. The availability of an oracle e.g. for NP questions from P-solving machine would subsume both NP and coNP, etc.

Polynomial Hierarchy, Part II

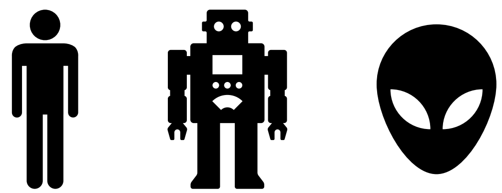
(via formal logic, directly)



$$
\begin{tikzcd}[column sep=small]
& \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
& \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
& \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
& \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

Polynomial Hierarchy, Part II

(via formal logic, directly)

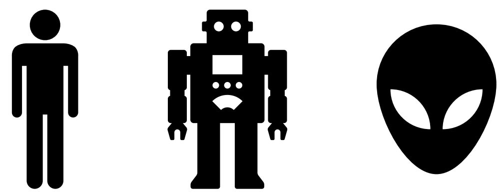


$$
\begin{tikzcd}
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

$\boxed{\text{Eg:}}$

Polynomial Hierarchy, Part II

(via formal logic, directly)



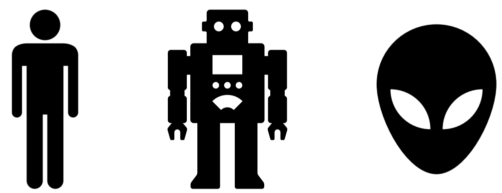
$$\begin{tikzcd}
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

$\boxed{\text{Eg:}}$

$$\langle \phi_1, k \rangle \in L \text{ iff } \exists \phi_2 \forall \alpha KLogEquiv(\phi_1, \phi_2, |\phi_2| \leq k, \alpha(\phi_1) = \alpha(\phi_2))$$

Polynomial Hierarchy, Part II

(via formal logic, directly)



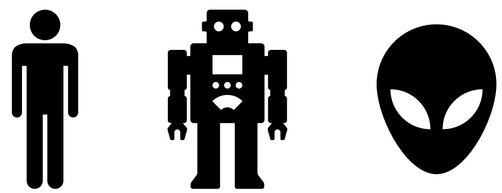
$$
\begin{tikzcd}[column sep=small]
 & \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
 & \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
 & \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
 & \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

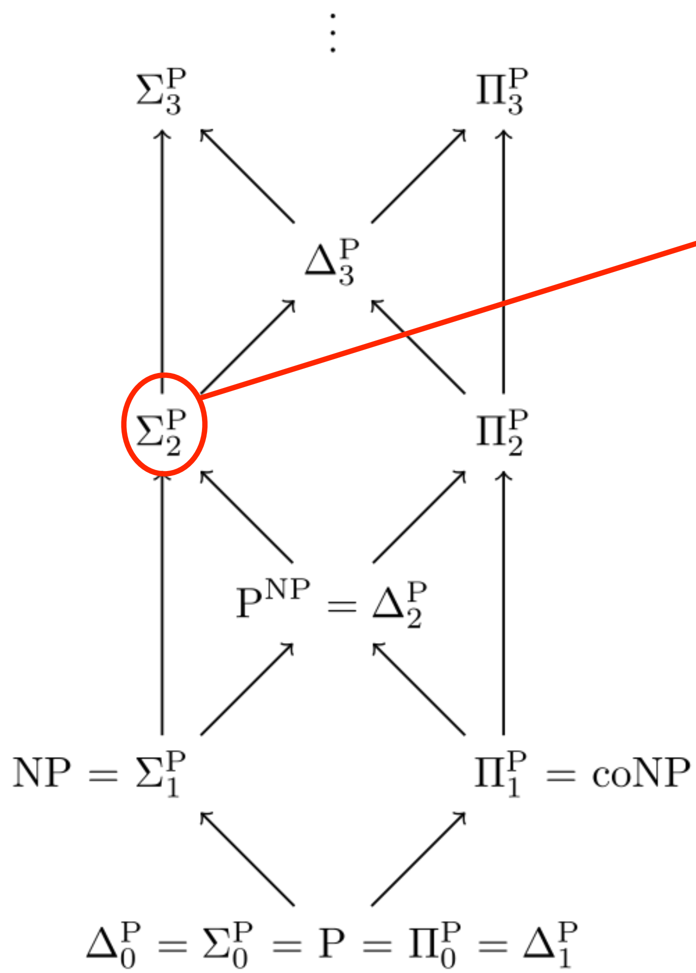
Eg:

$$\langle \phi_1, k \rangle \in L \text{ iff } \exists \phi_2 \forall \alpha KLogEquiv(\phi_1, \phi_2, |\phi_2| \leq k, \alpha(\phi_1) = \alpha(\phi_2))$$

Polynomial Hierarchy, Part II

(via formal logic, directly)





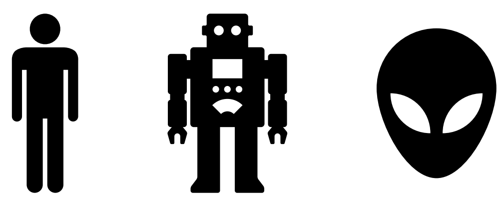
free variables

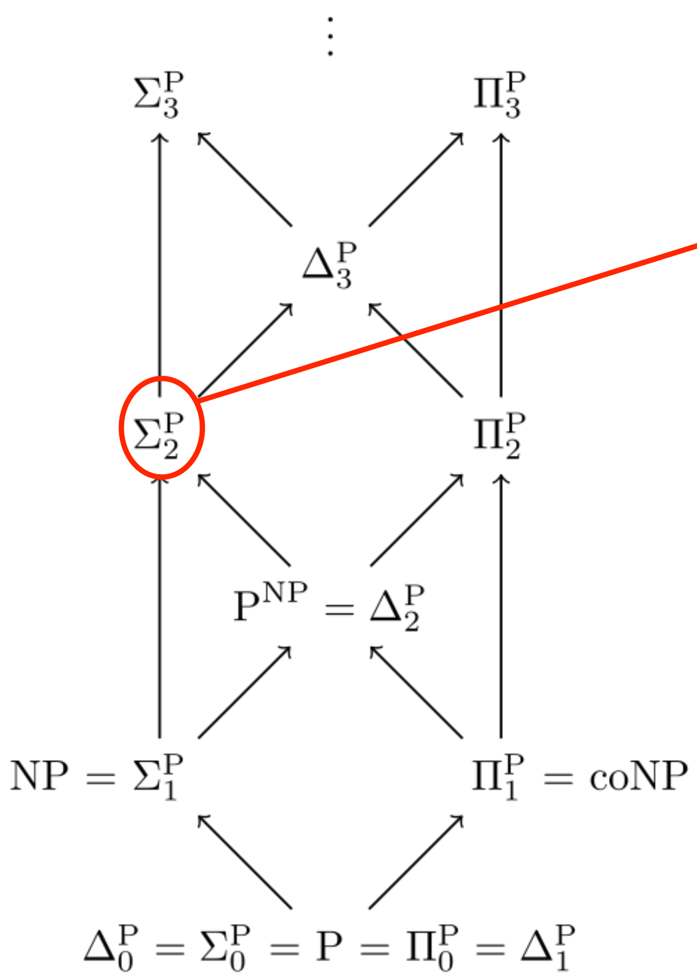
Eg:

$$\langle \phi_1, k \rangle \in L \text{ iff } \exists \phi_2 \forall \alpha KLogEquiv(\phi_1, \phi_2, |\phi_2| \leq k, \alpha(\phi_1) = \alpha(\phi_2))$$

Polynomial Hierarchy, Part II

(via formal logic, directly)





free variables

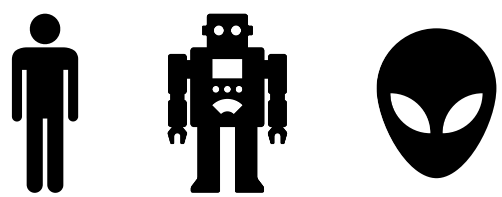
Eg:

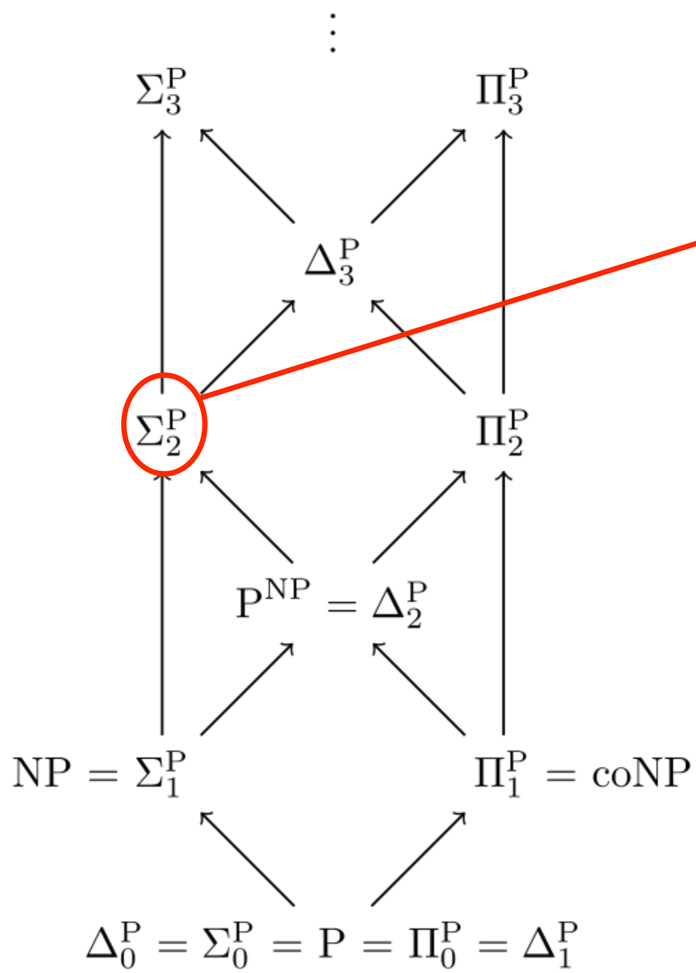
$$\langle \phi_1, k\rangle \in L \text{ iff } \exists \phi_2 \forall \alpha \, KLogEquiv(\phi_1, \phi_2, |\phi_2| \leq k, \alpha(\phi_1) = \alpha(\phi_2))$$

Now we generalize:

Polynomial Hierarchy, Part II

(via formal logic, directly)





free variables

Eg:

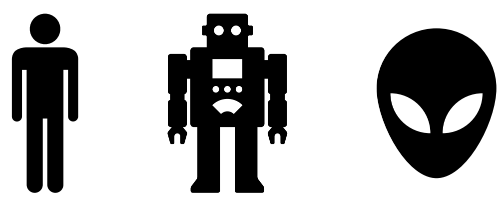
$$\langle \phi_1, k \rangle \in L \text{ iff } \exists \phi_2 \forall \alpha\, KLogEquiv(\phi_1, \phi_2, |\phi_2| \leq k, \alpha(\phi_1) = \alpha(\phi_2))$$

Now we generalize:

$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

Polynomial Hierarchy, Part II

(via formal logic, directly)



$$
\begin{tikzcd}[column sep=small]
& \vdots & \\
\Sigma_3^{\mathrm{P}} & & \Pi_3^{\mathrm{P}} \\
& \Delta_3^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\Sigma_2^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_2^{\mathrm{P}} \arrow[uu] \arrow[ul] \\
& \mathrm{P}^{\mathrm{NP}} = \Delta_2^{\mathrm{P}} \arrow[ul] \arrow[ur] & \\
\mathrm{NP} = \Sigma_1^{\mathrm{P}} \arrow[uu] \arrow[ur] & & \Pi_1^{\mathrm{P}} = \mathrm{coNP} \arrow[uu] \arrow[ul] \\
& \Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \mathrm{P} = \Pi_0^{\mathrm{P}} = \Delta_1^{\mathrm{P}} \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

free variables

Eg:

$$\langle \phi_1, k \rangle \in L \text{ iff } \exists \phi_2 \forall \alpha KLogEquiv(\phi_1, \phi_2, |\phi_2| \leq k, \alpha(\phi_1) = \alpha(\phi_2))$$

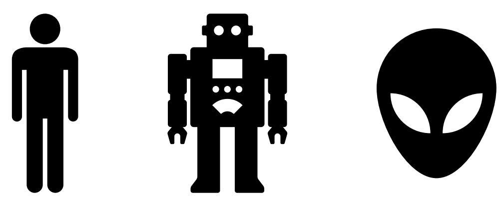
Now we generalize:

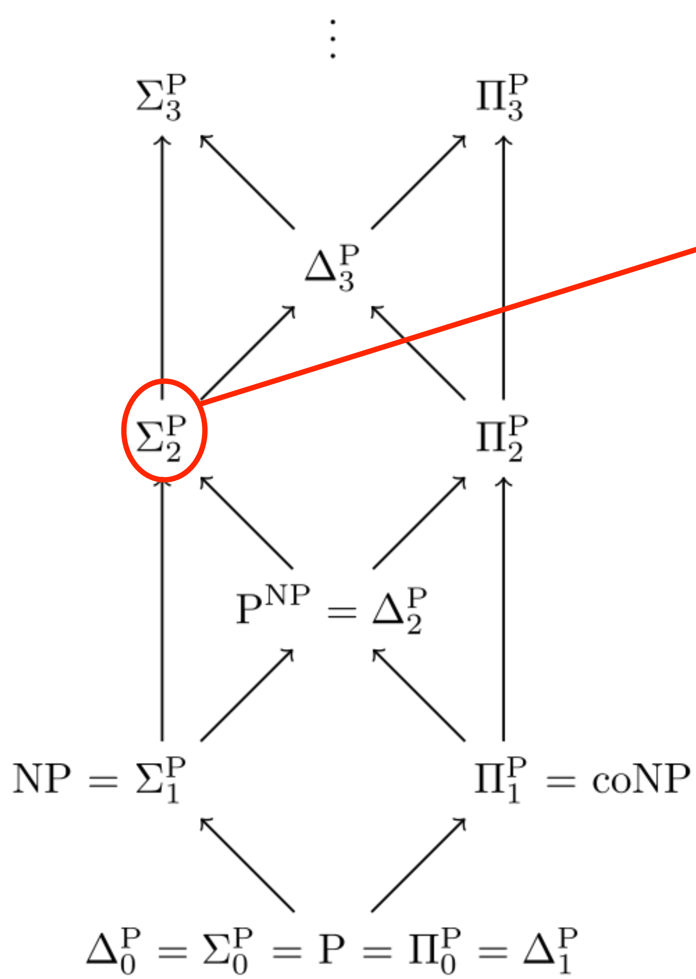
$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Polynomial Hierarchy, Part II

(via formal logic, directly)





free variables

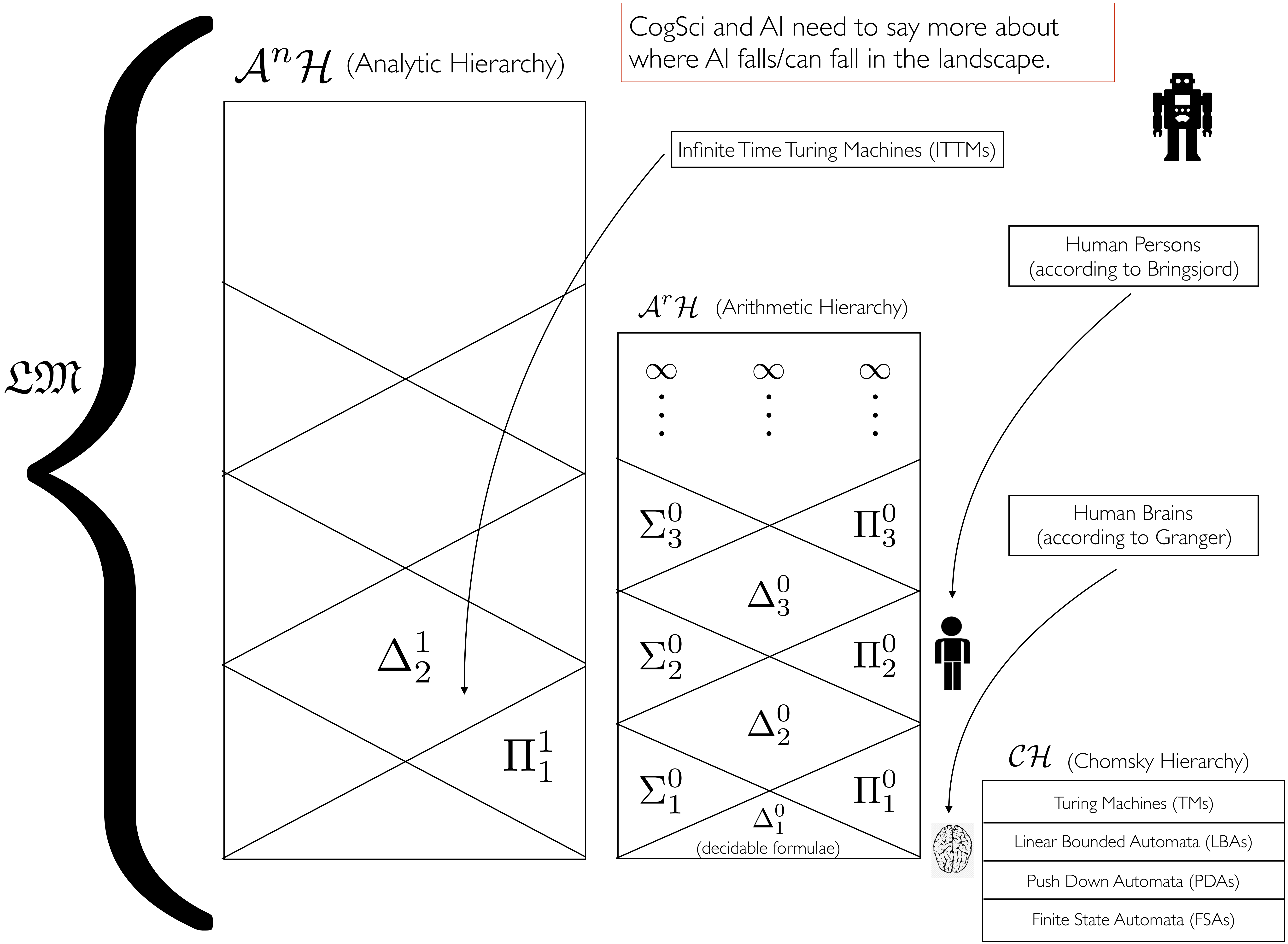
Eg:

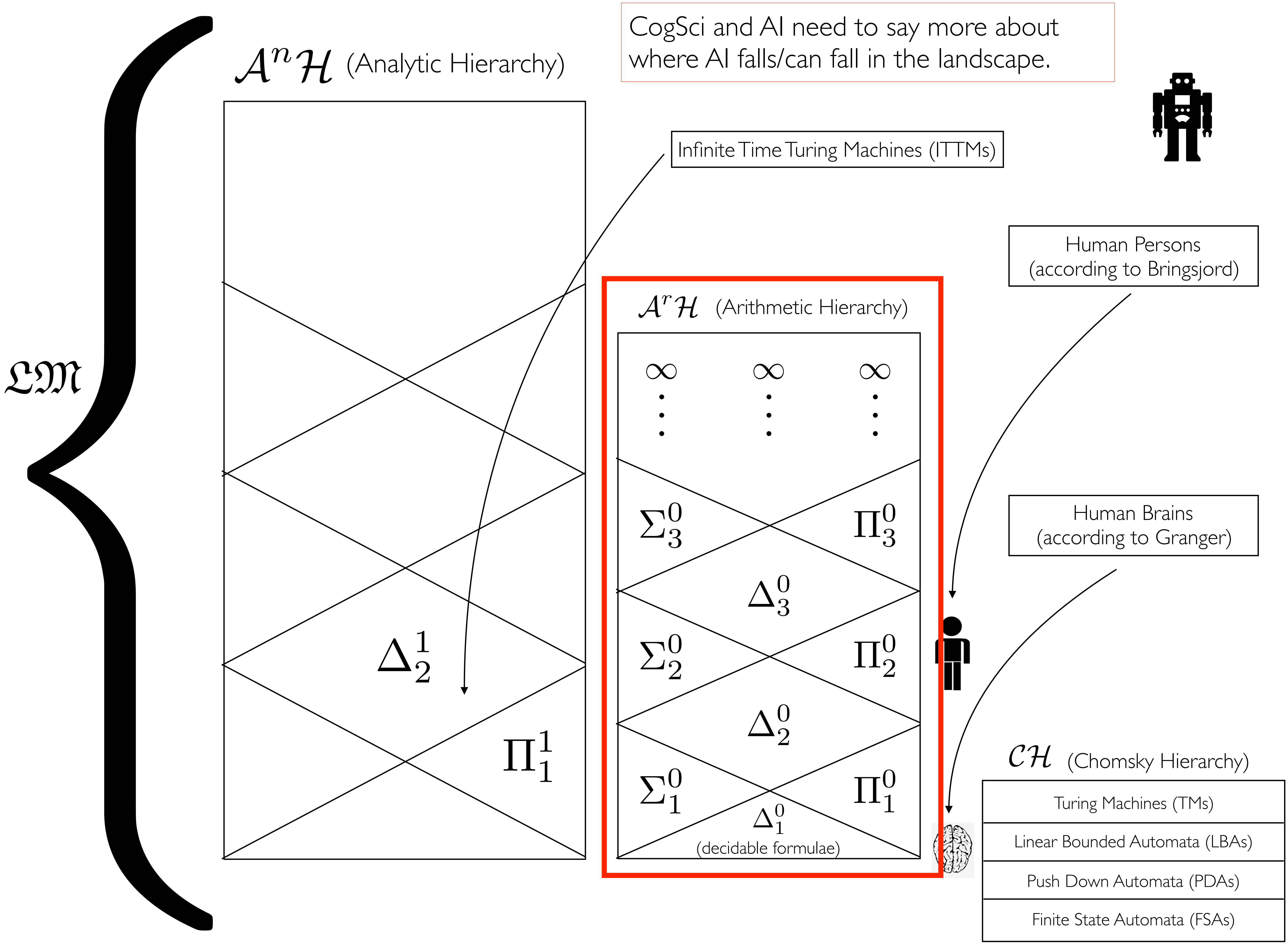
$$\langle \phi_1, k\rangle \in L \text{ iff } \exists\phi_2\forall\alpha\, KLogEquiv(\phi_1, \phi_2, |\phi_2| \le k, \alpha(\phi_1) = \alpha(\phi_2))$$

Now we generalize:

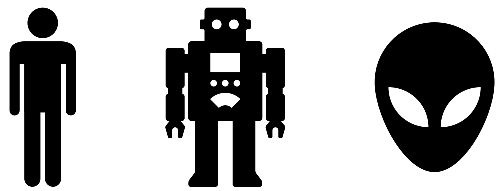
$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$


CogSci and AI need to say more about where AI falls/can fall in the landscape.
$\mathcal{LM}$
$\mathcal{A}^n\mathcal{H}$ (Analytic Hierarchy)
Δ^1_2
Π^1_1
Infinite Time Turing Machines (ITTMs)
$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)
∞
∞
∞
Σ^0_3
Π^0_3
Δ^0_3
Σ^0_2
Π^0_2
Δ^0_2
Σ^0_1
Π^0_1
Δ^0_1
(decidable formulae)
Human Persons
(according to Bringsjord)
Human Brains
(according to Granger)
$\mathcal{CH}$ (Chomsky Hierarchy)
Turing Machines (TMs)
Linear Bounded Automata (LBAs)
Push Down Automata (PDAs)
Finite State Automata (FSAs)

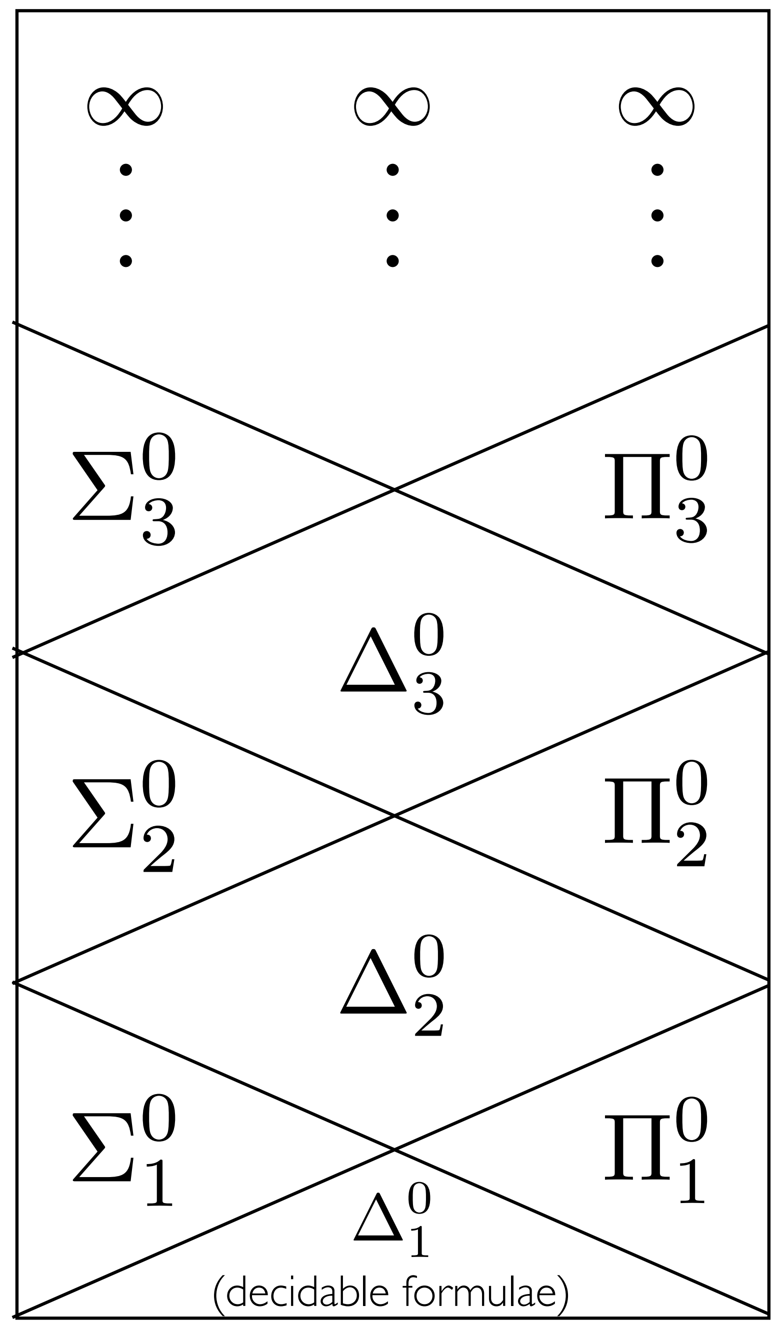


CogSci and AI need to say more about where AI falls/can fall in the landscape.
$\mathfrak{LM}$
$\mathcal{A}^n\mathcal{H}$ (Analytic Hierarchy)
Infinite Time Turing Machines (ITTMs)
Δ^1_2
Π^1_1
$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)
∞
∞
∞
Σ^0_3
Π^0_3
Δ^0_3
Σ^0_2
Π^0_2
Δ^0_2
Σ^0_1
Π^0_1
Δ^0_1
(decidable formulae)
Human Persons (according to Bringsjord)
Human Brains (according to Granger)
$\mathcal{CH}$ (Chomsky Hierarchy)
Turing Machines (TMs)
Linear Bounded Automata (LBAs)
Push Down Automata (PDAs)
Finite State Automata (FSAs)



$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v[\exists k(\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k'(\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)

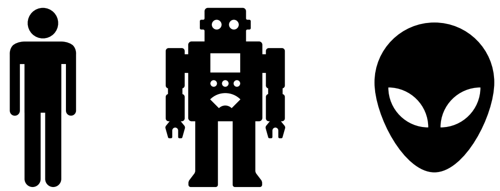


semi-decidable

$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

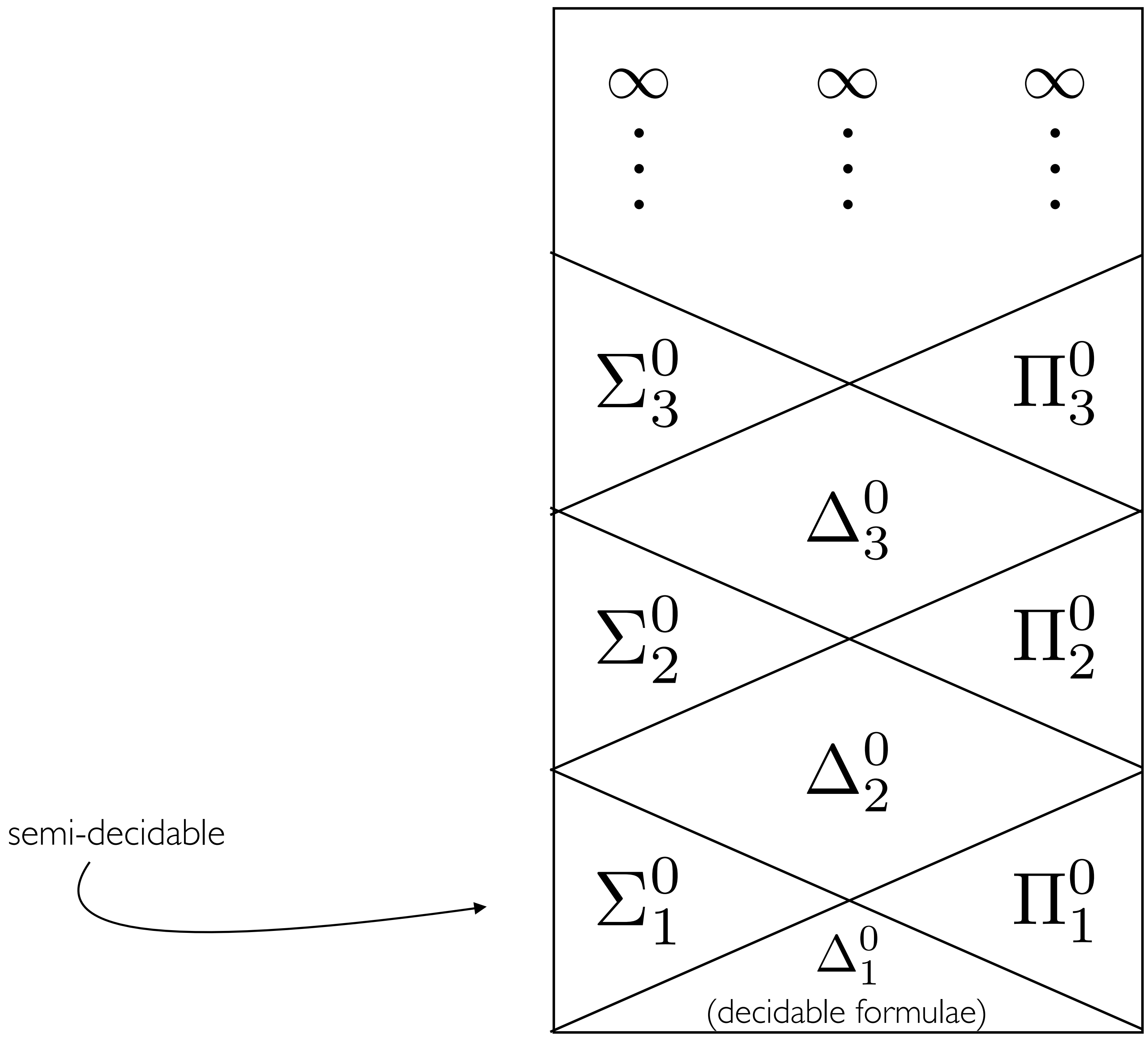
Arithmetic Hierarchy, Part I

Can you see the carryover from PH?



$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v[\exists k(\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k'(\langle \mathfrak{m}_2, u\rangle : v, k')]$$

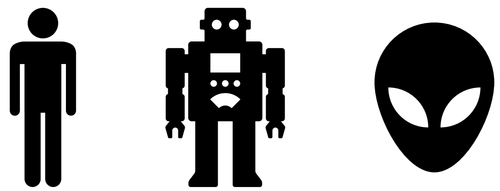
$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?



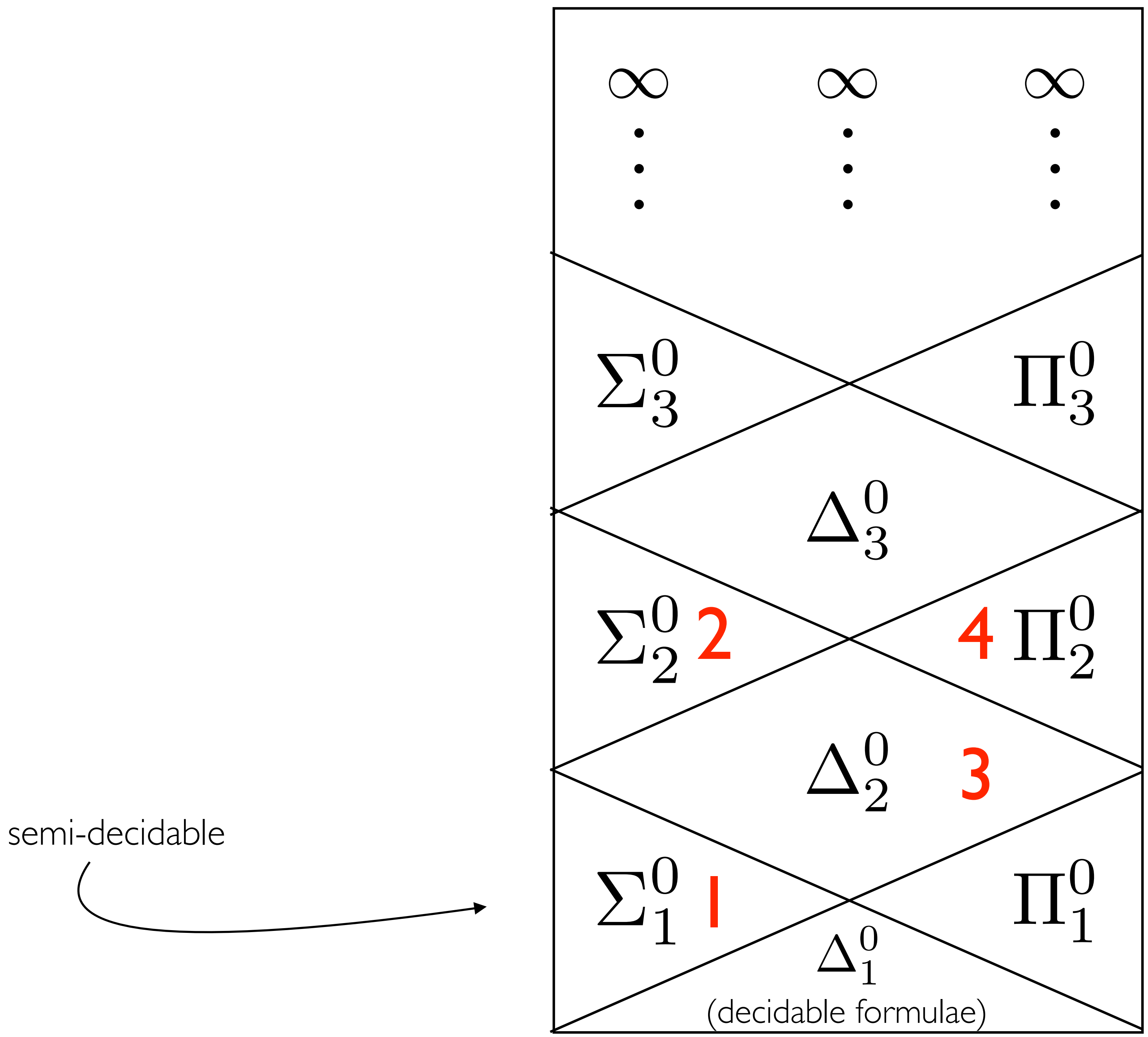
$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k(\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k'(\langle \mathfrak{m}_2, u\rangle : v, k')]$$

Let R be a Turing-decidable (= decidable, *simpliciter*) dyadic relation. Where is the set:

$$\{x : \exists y R(x, y)\},$$

1 2 3 or 4?

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



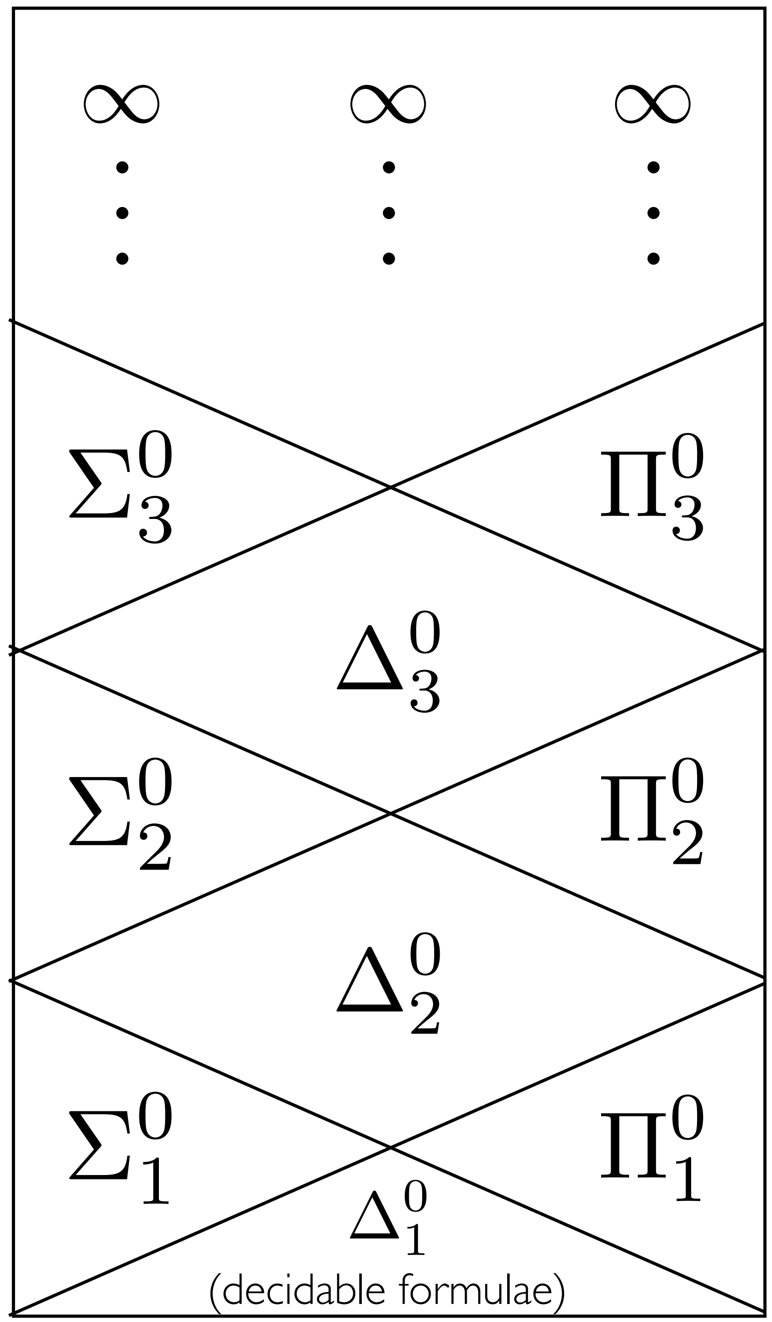
$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u \rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u \rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)

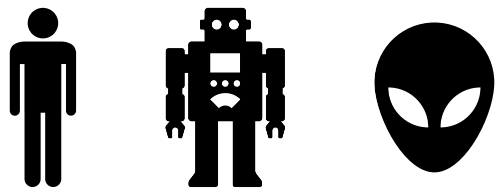


semi-decidable

$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

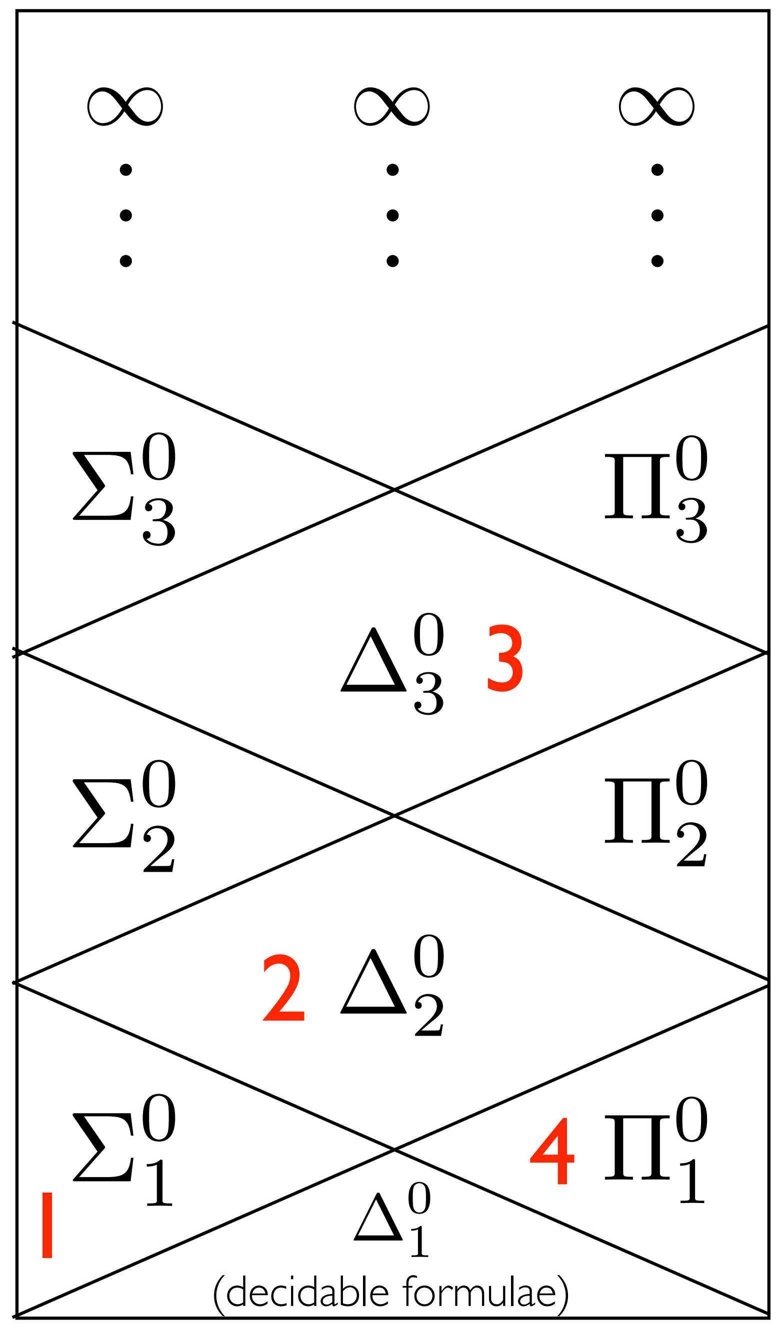
Arithmetic Hierarchy, Part I

Can you see the carryover from PH?



$$\textbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u \rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u \rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



semi-decidable

Let R be a Turing-decidable (= decidable, *simpliciter*) dyadic relation. Where is the set:

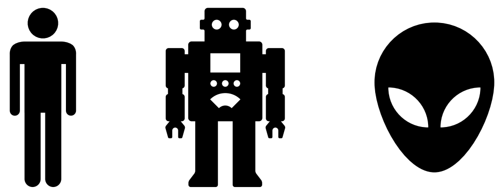
$$\{x : \forall y R(x, y)\},$$

1 2 3 or 4?

$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

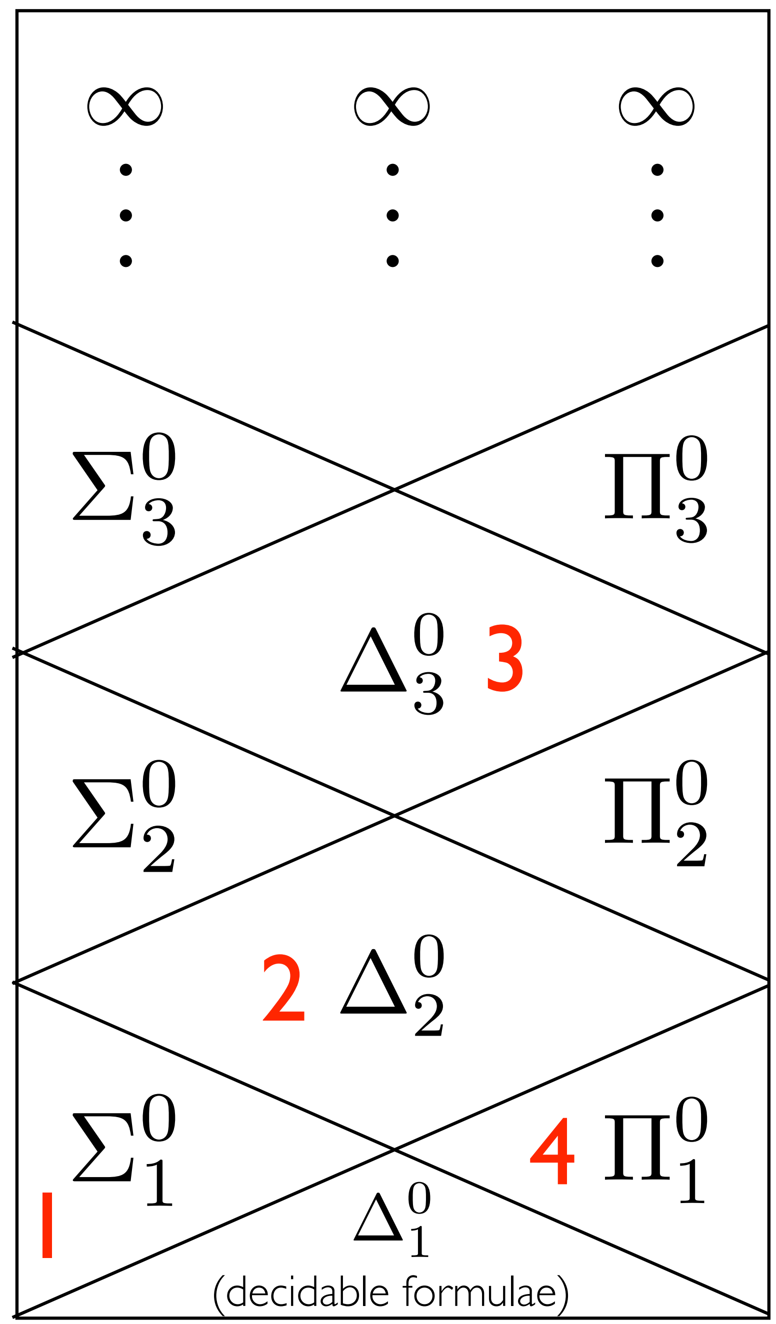
Arithmetic Hierarchy, Part I

Can you see the carryover from PH?



$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u \rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u \rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



semi-decidable

Let R be a Turing-decidable (= decidable, *simpliciter*) dyadic relation. Where is the set:

$$\{x : \forall y R(x, y)\},$$

1 2 3 or 4?

$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

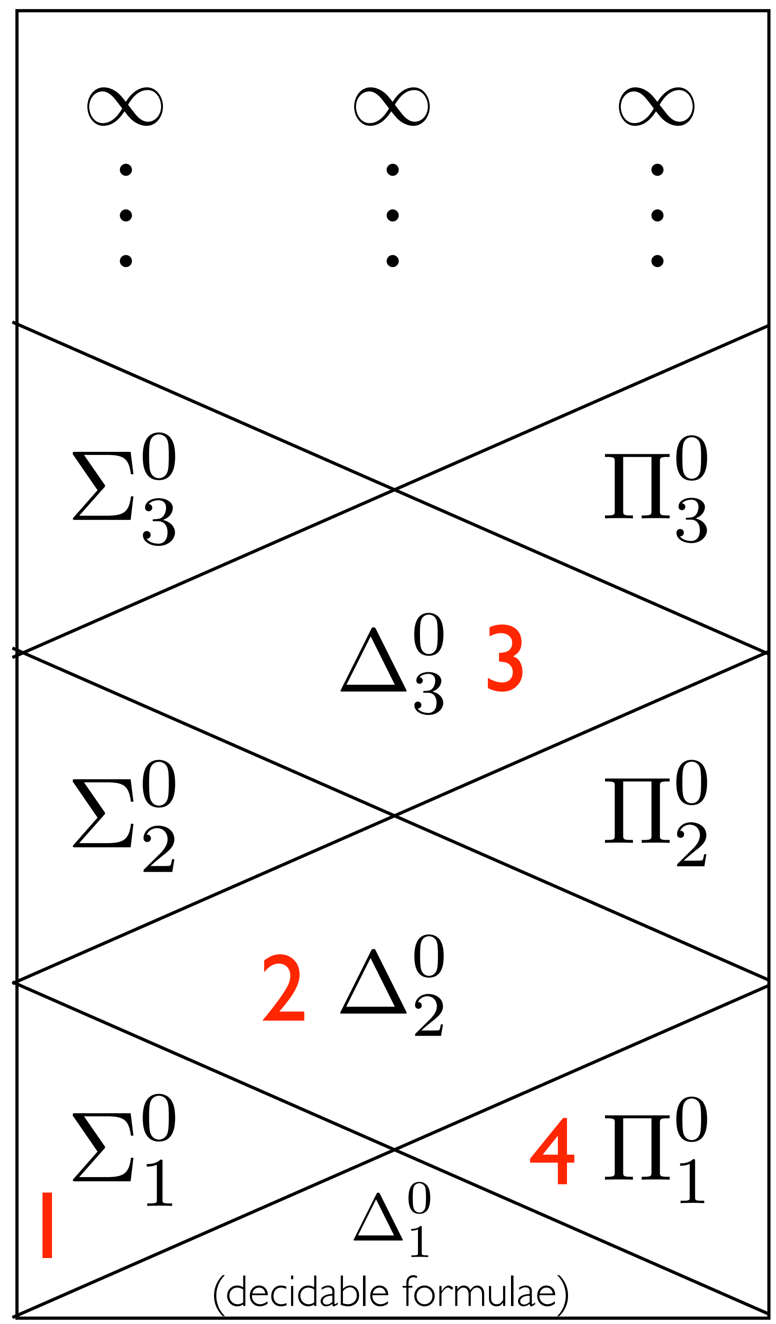
$$\Delta^0_1 = \Sigma^0_1 \cap \Pi^0_1$$

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



semi-decidable

Let R be a Turing-decidable (= decidable, *simpliciter*) dyadic relation. Where is the set:

$$\{x : \forall y R(x, y)\},$$

1 2 3 or 4?

$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

$$\Delta^0_1 = \Sigma^0_1 \cap \Pi^0_1$$

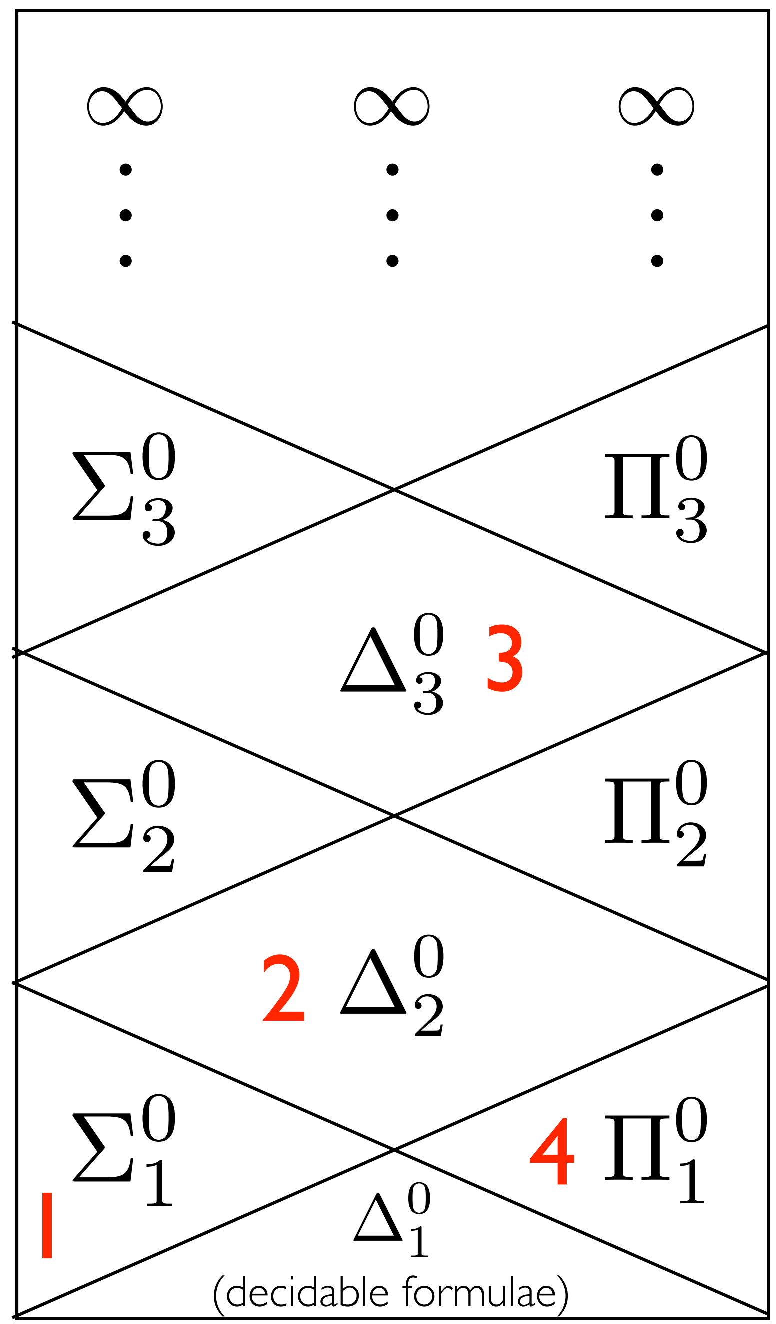
Arithmetic Hierarchy, Part I

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u \rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u \rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



semi-decidable

$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

Let R be a Turing-decidable (= decidable, *simpliciter*) dyadic relation. Where is the set:

$$\{x : \forall y R(x, y)\},$$

1 2 3 or 4?

$$x \in \Sigma_i \text{ iff } \exists R \; \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

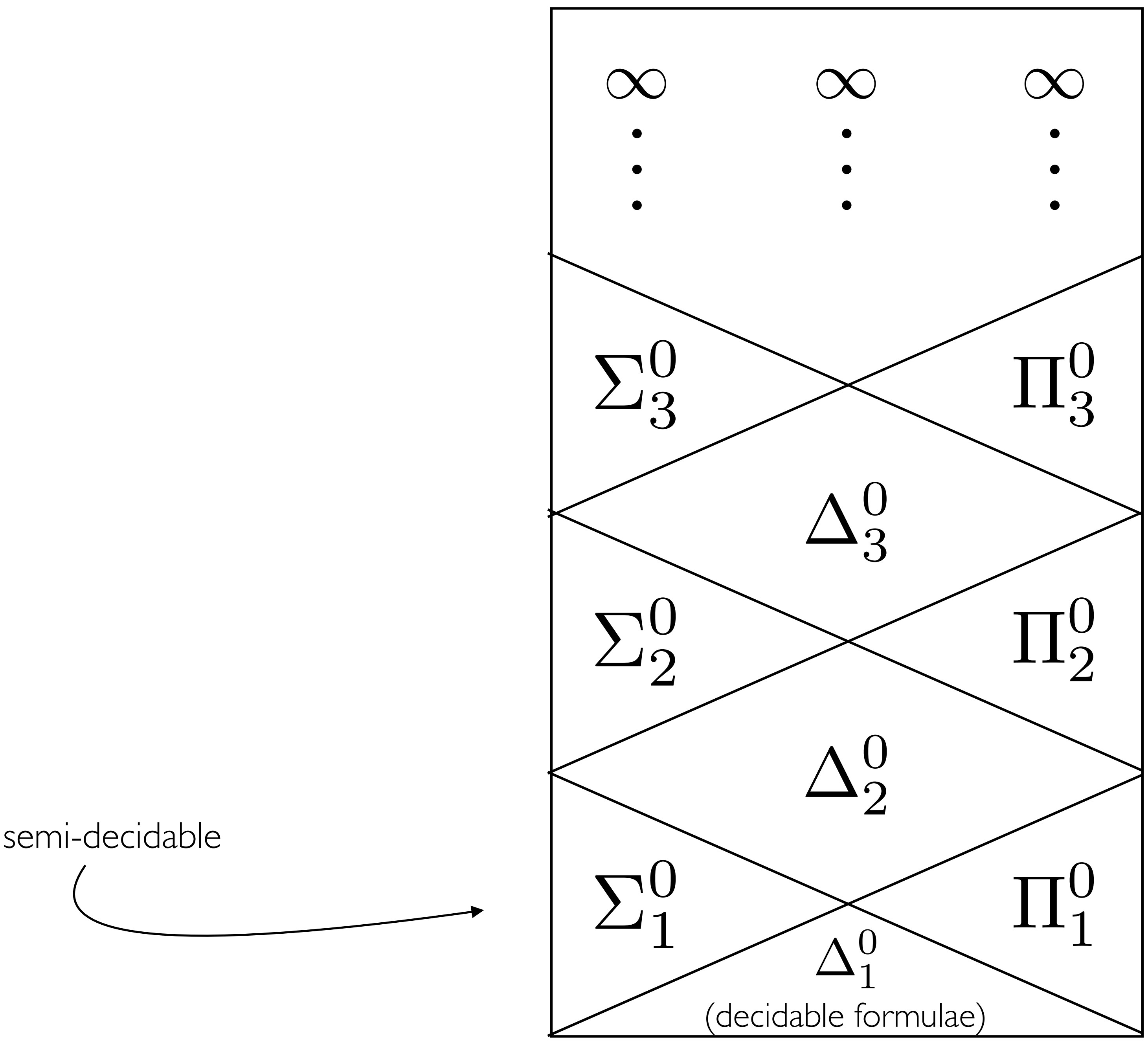
$$x \in \Pi_i \text{ iff } \exists R \; \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

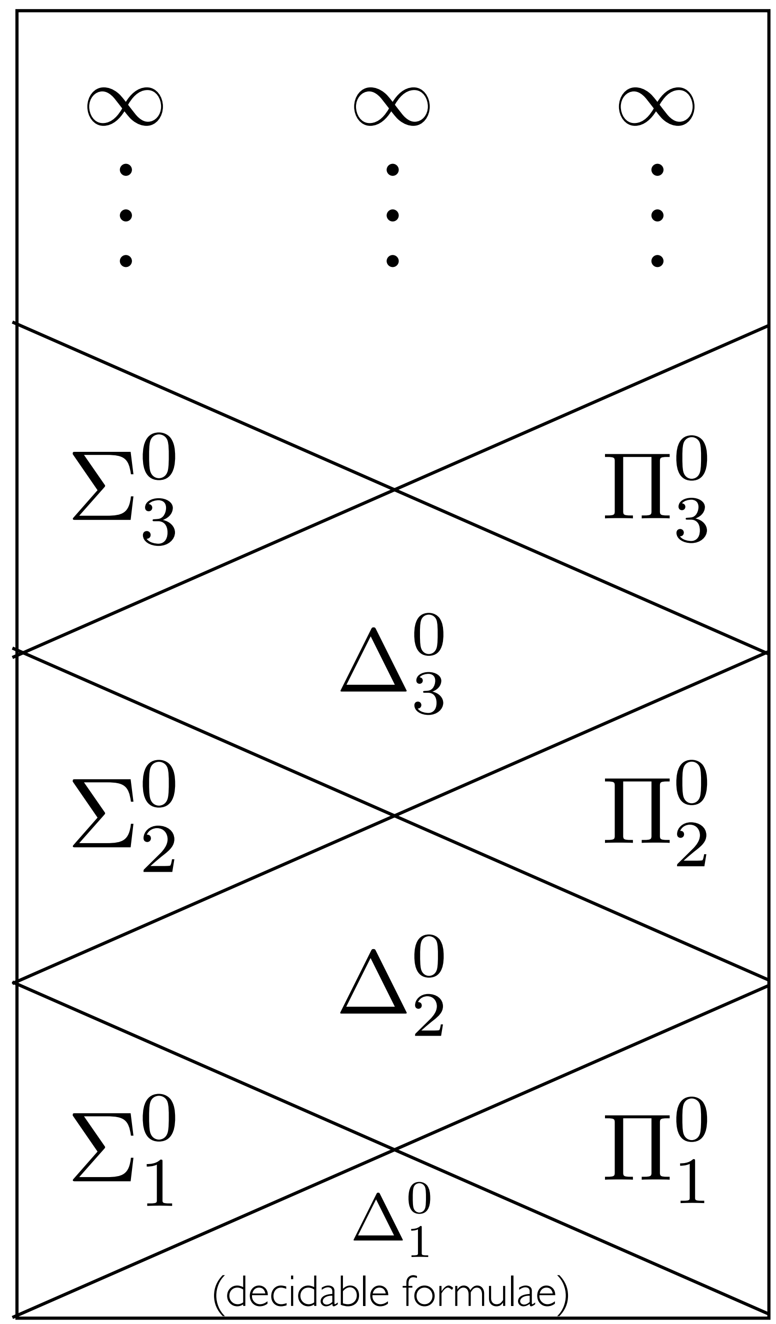
Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\textbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u \rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u \rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)

semi-decidable



$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

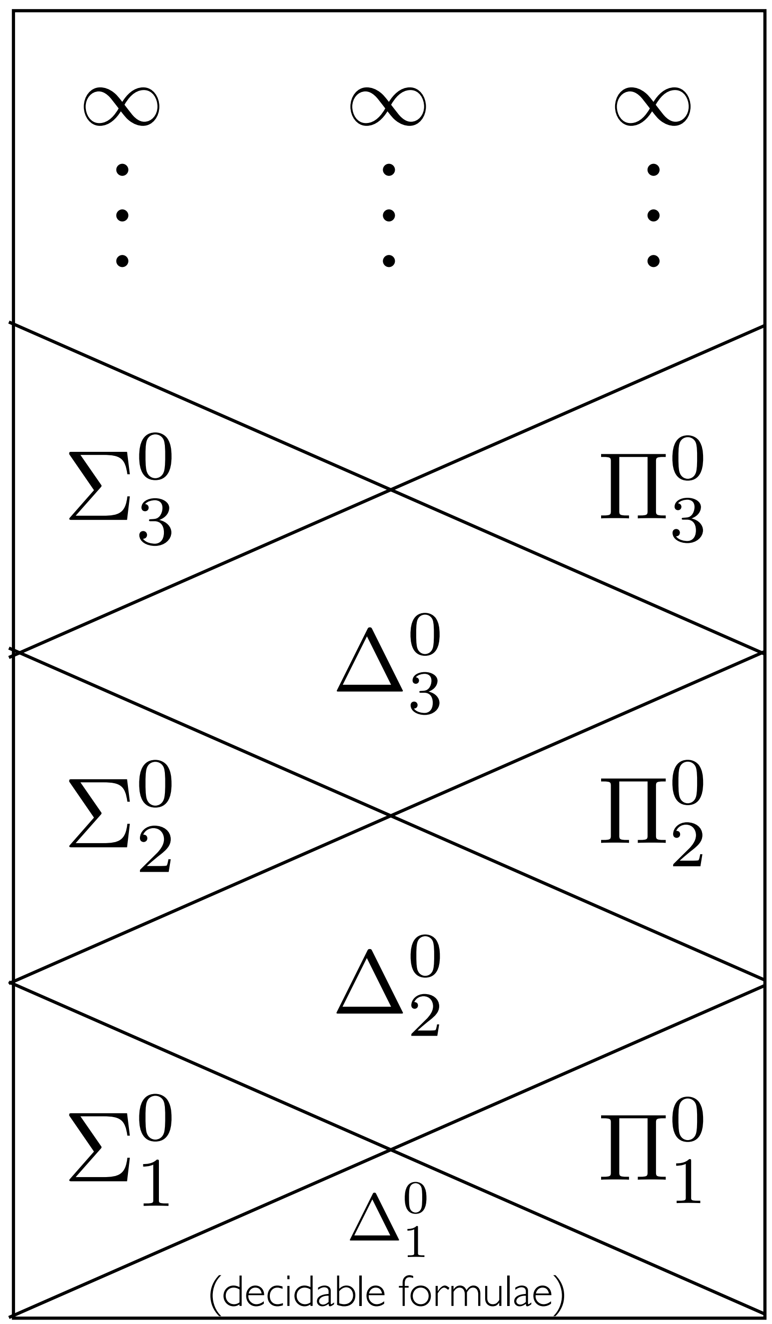
From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v[\exists k(\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k'(\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



semi-decidable

$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

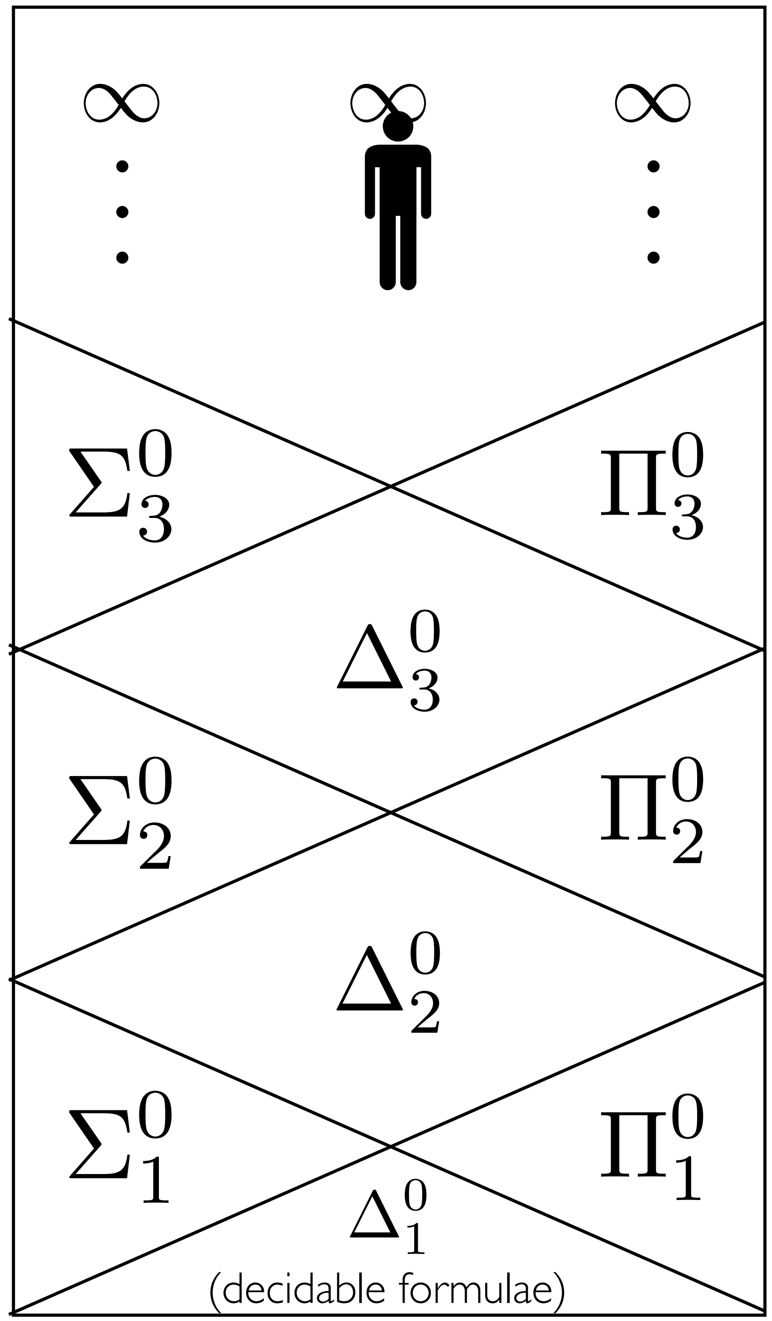
$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\textbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



semi-decidable

$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

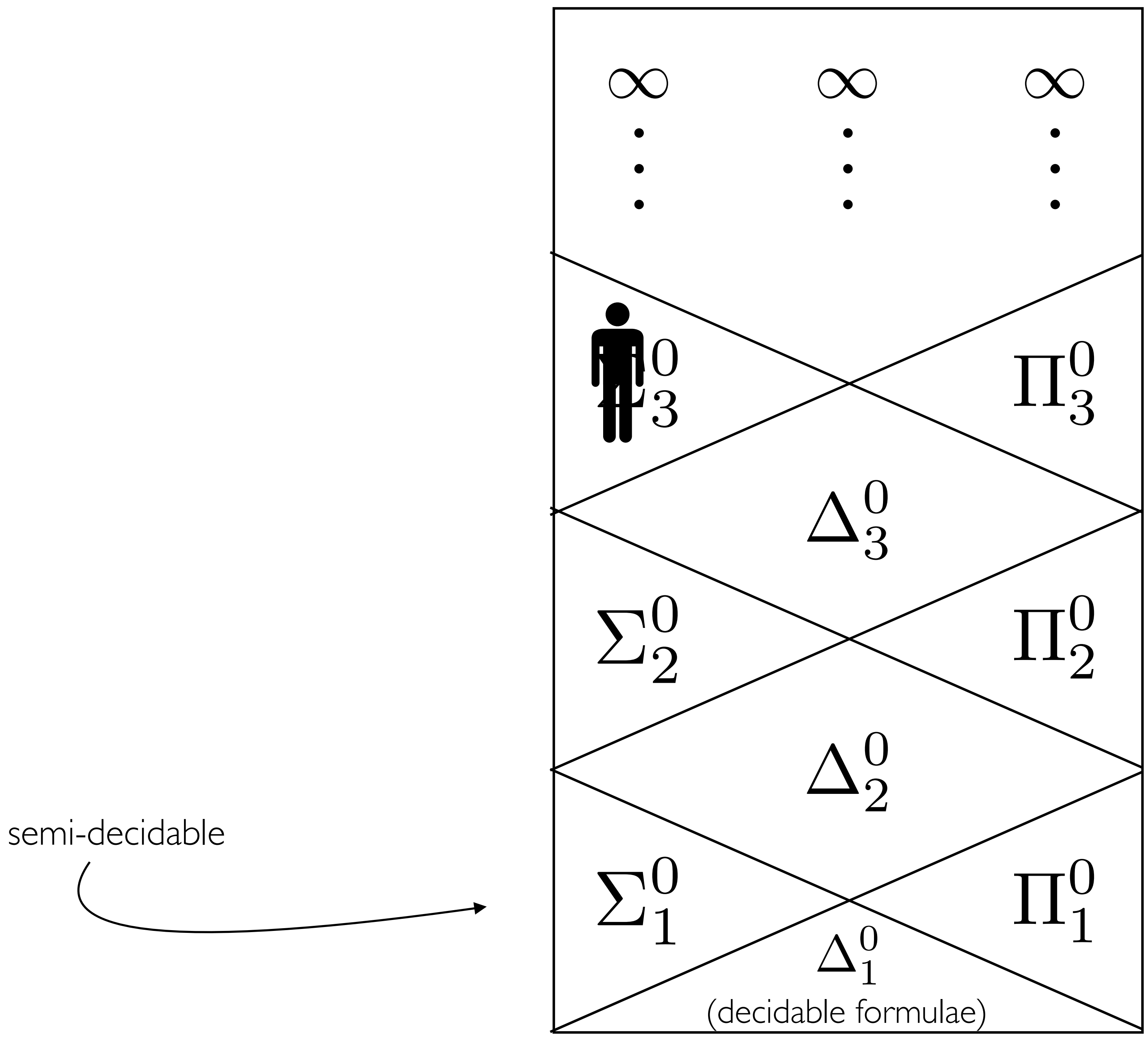
$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k(\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k'(\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

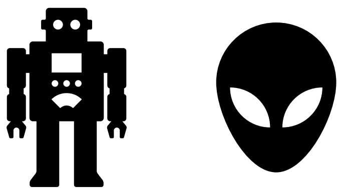
Try your hand at classifying! …

From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

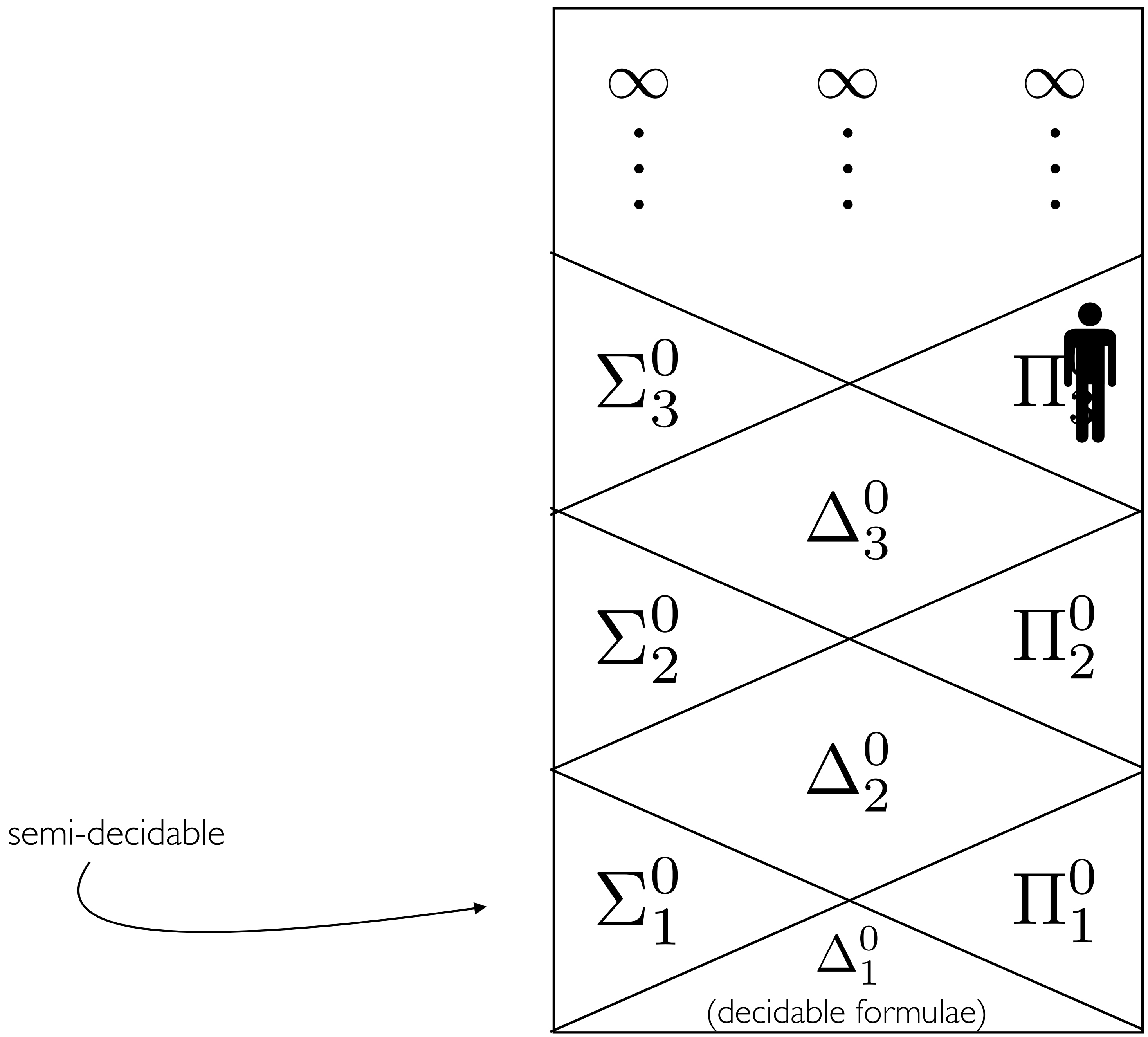
Arithmetic Hierarchy, Part I

Can you see the carryover from PH?



$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k(\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k'(\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

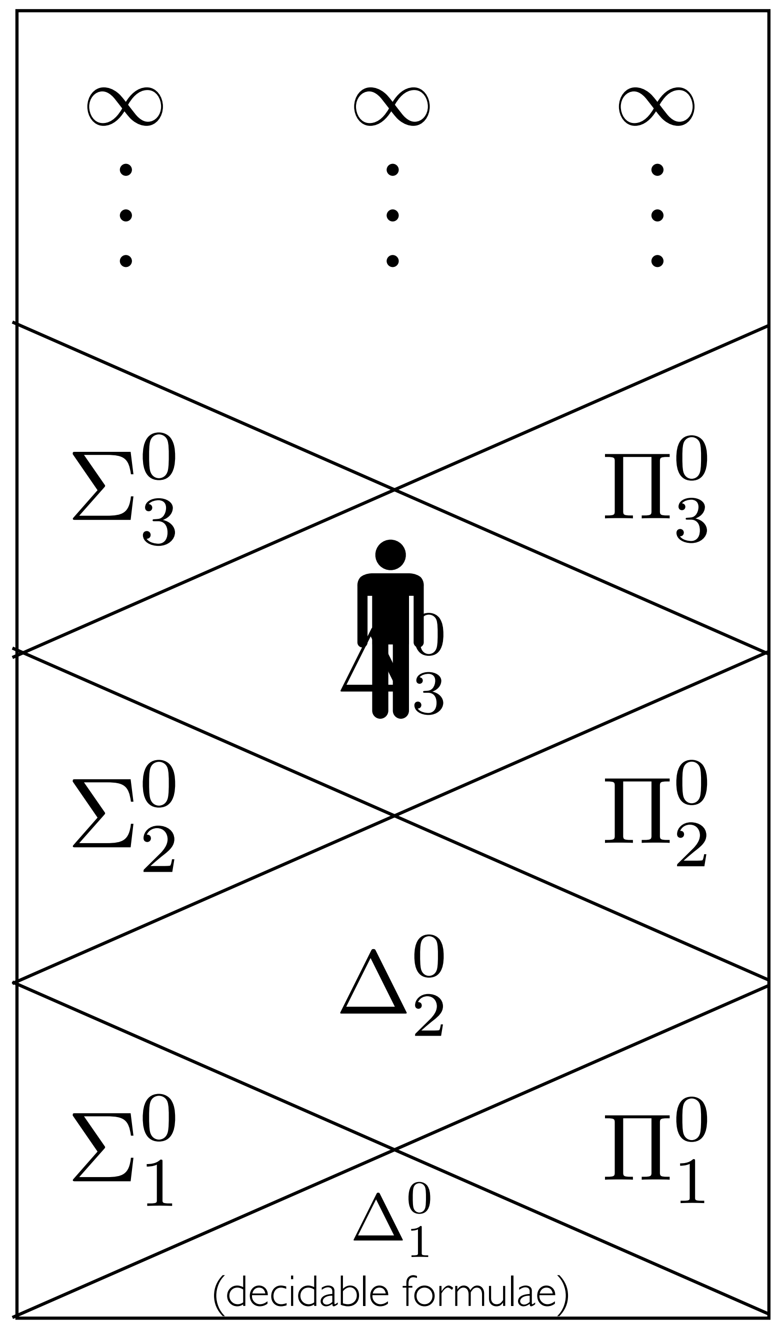
$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k(\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k'(\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



semi-decidable

$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

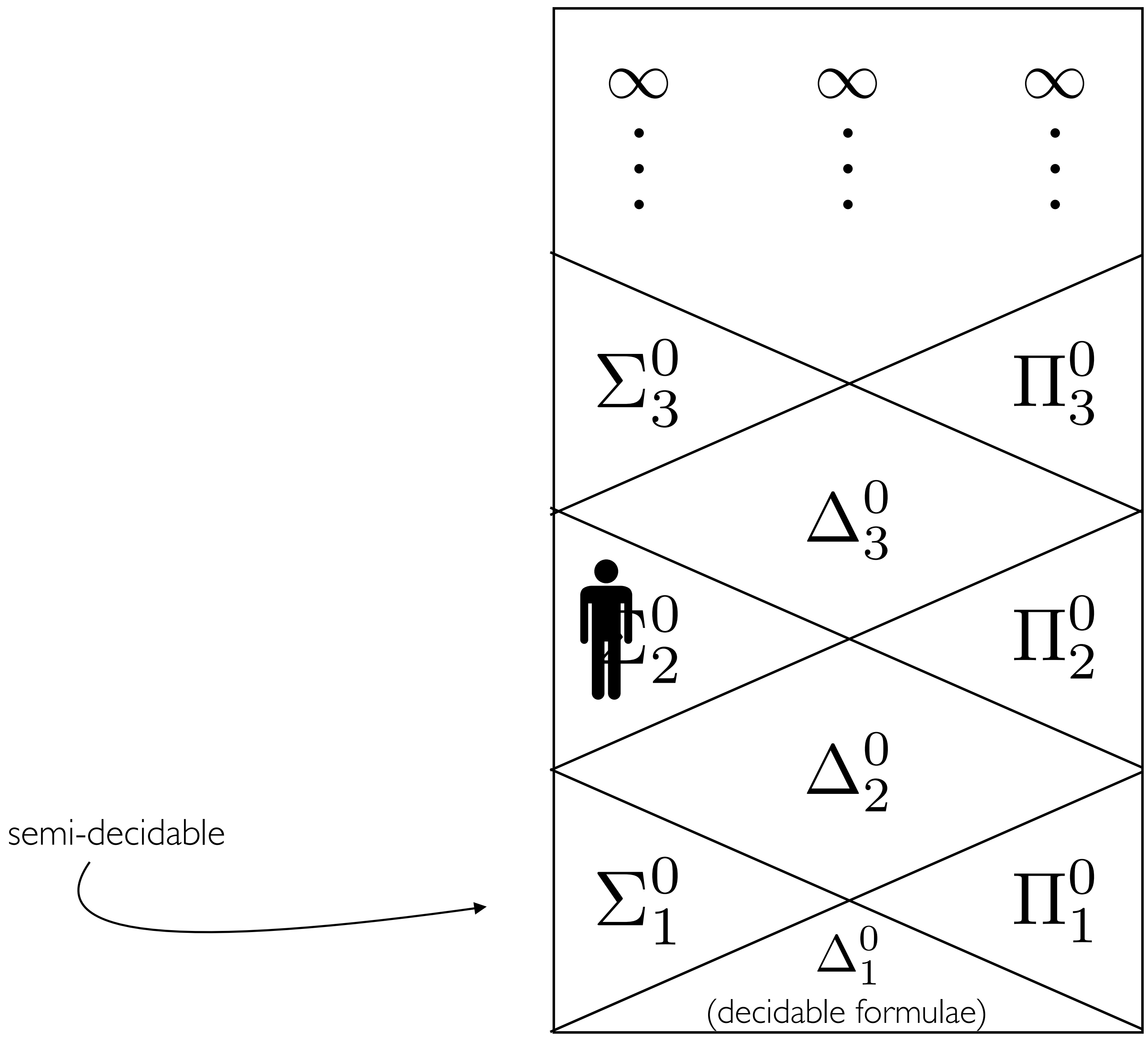
$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u \rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u \rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



$$x \in \Sigma_i \text{ iff } \exists R \; \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R \; \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

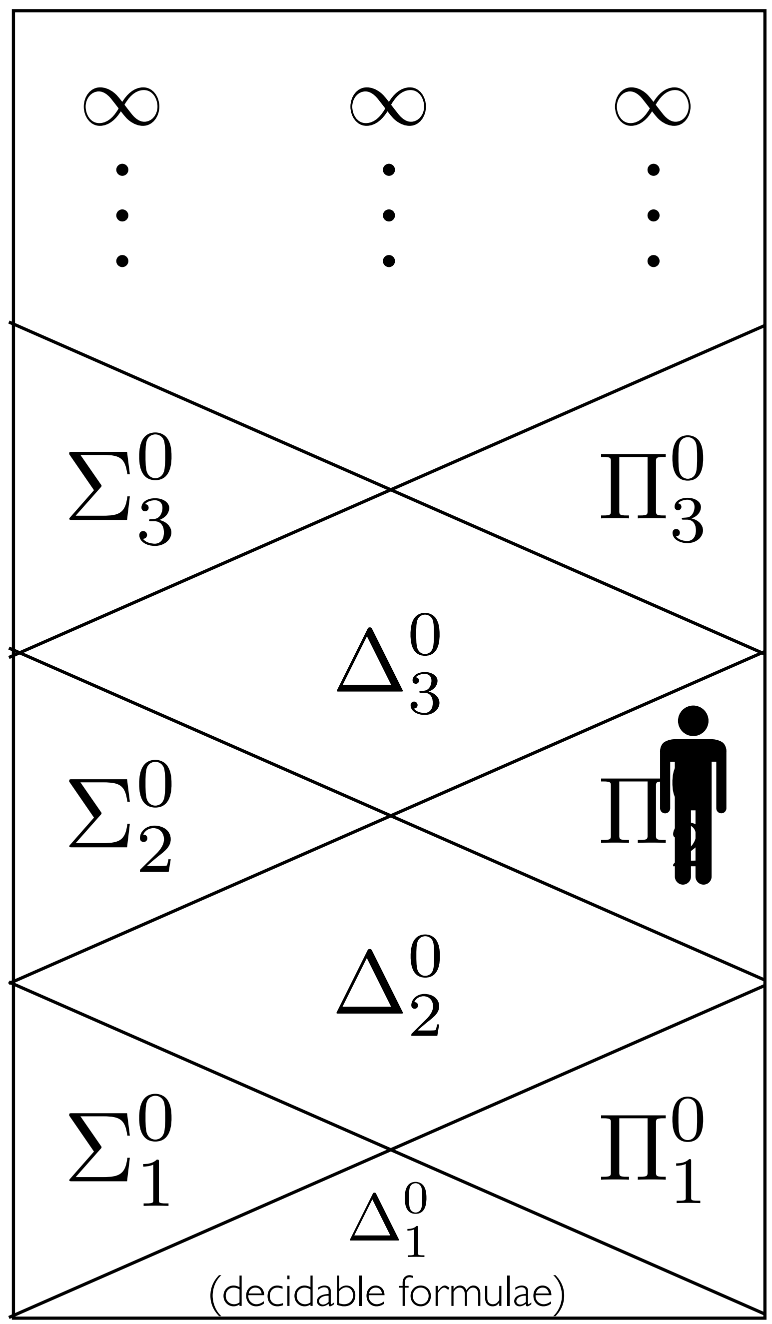
$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u \rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u \rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



semi-decidable

$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

$$x \in \Sigma_i \text{ iff } \exists R \; \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R \; \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

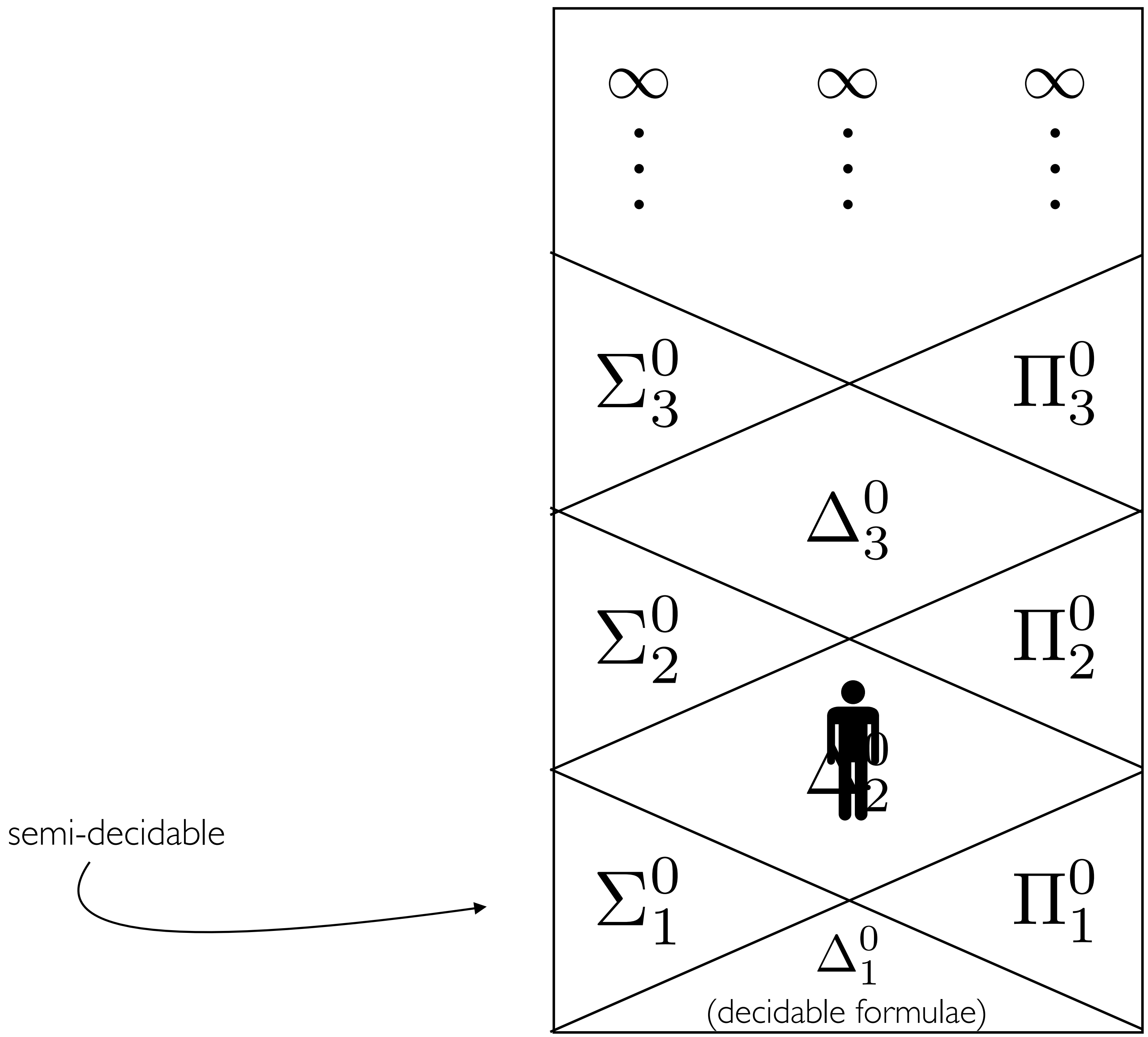
From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v[\exists k(\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k'(\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

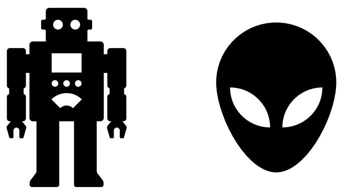
$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

Arithmetic Hierarchy, Part I

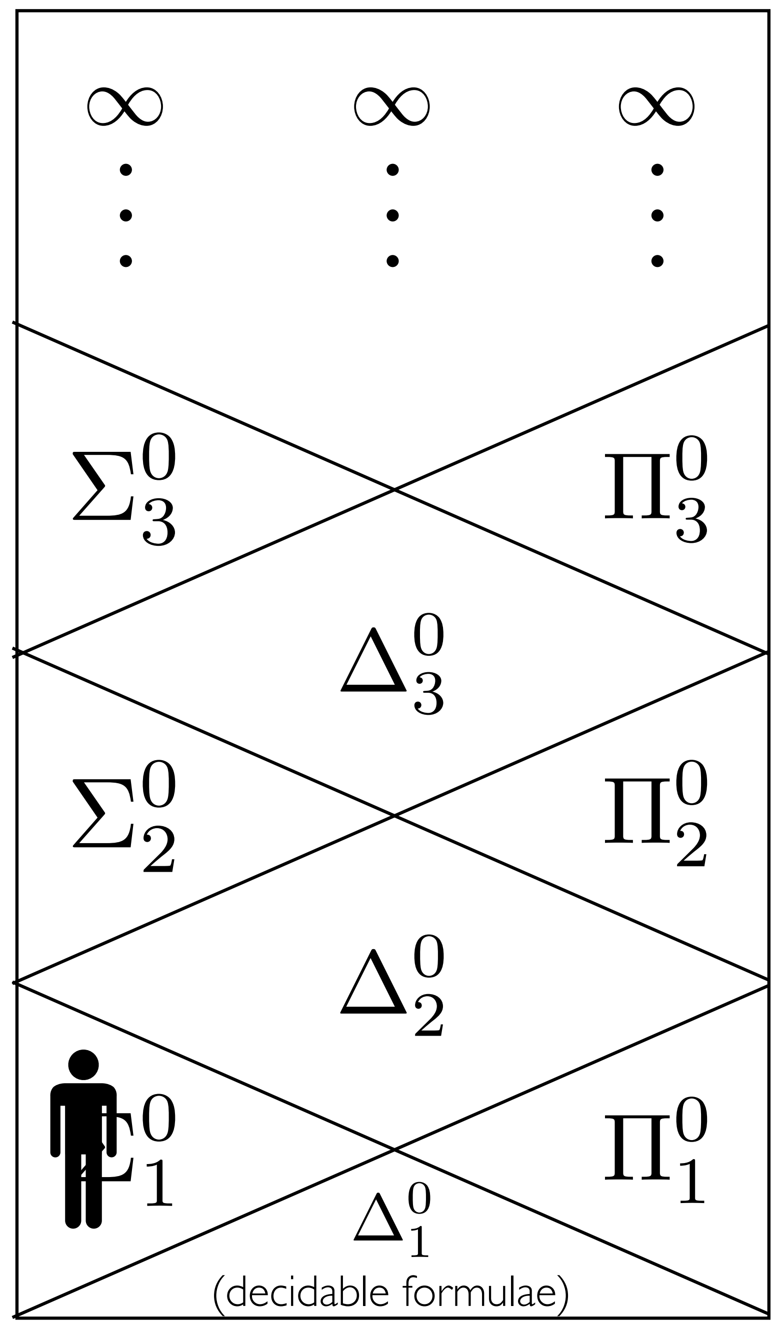
Can you see the carryover from PH?



$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v[\exists k(\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k'(\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)

semi-decidable



$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

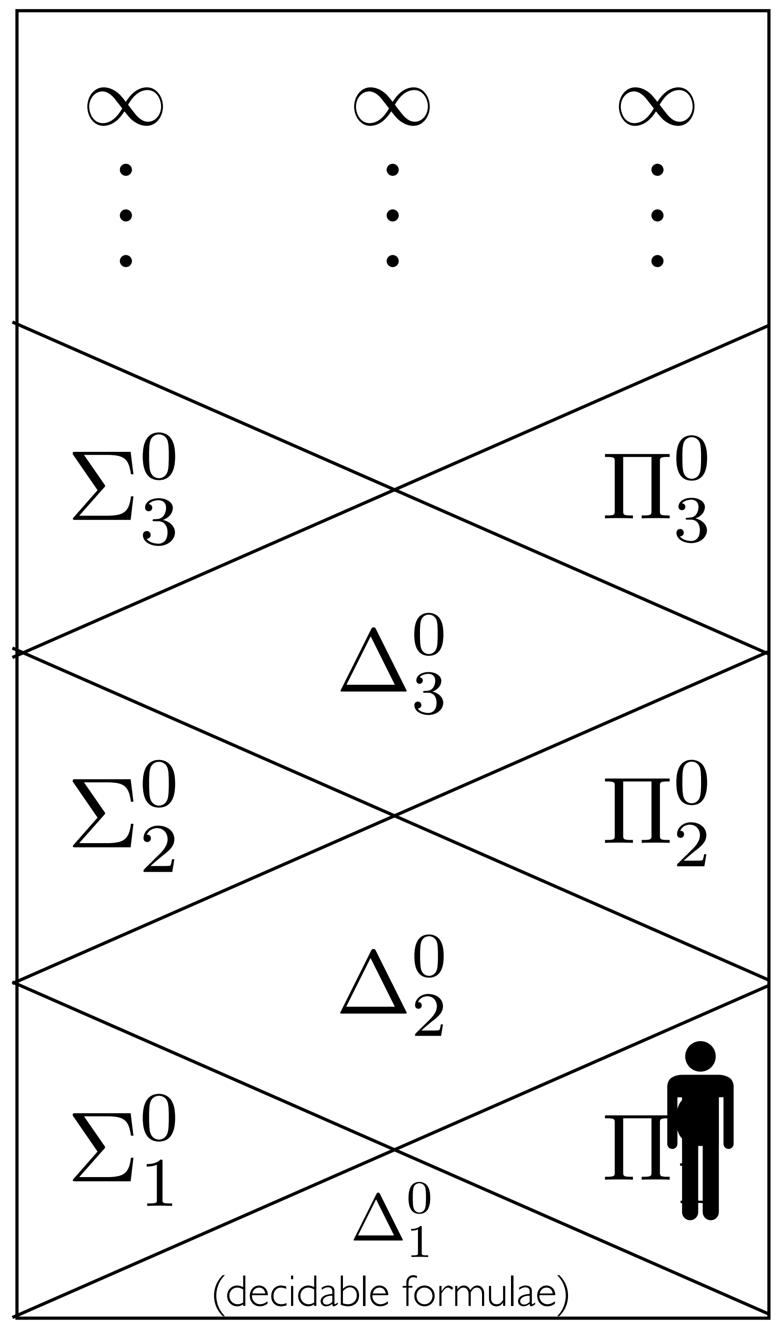
$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v[\exists k(\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k'(\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



semi-decidable

$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

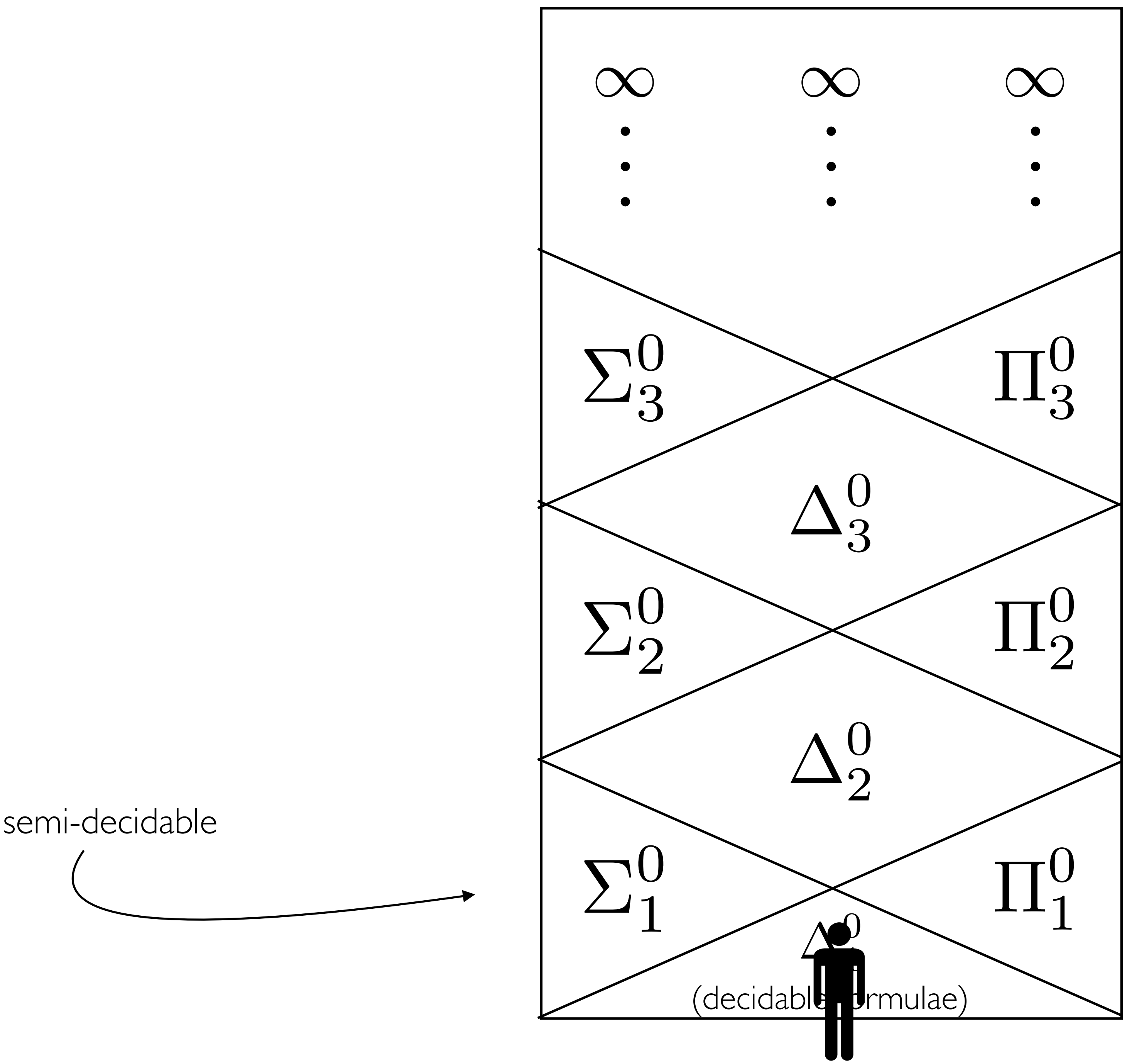
Arithmetic Hierarchy, Part I

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k(\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k'(\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

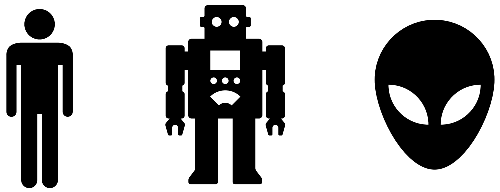
$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

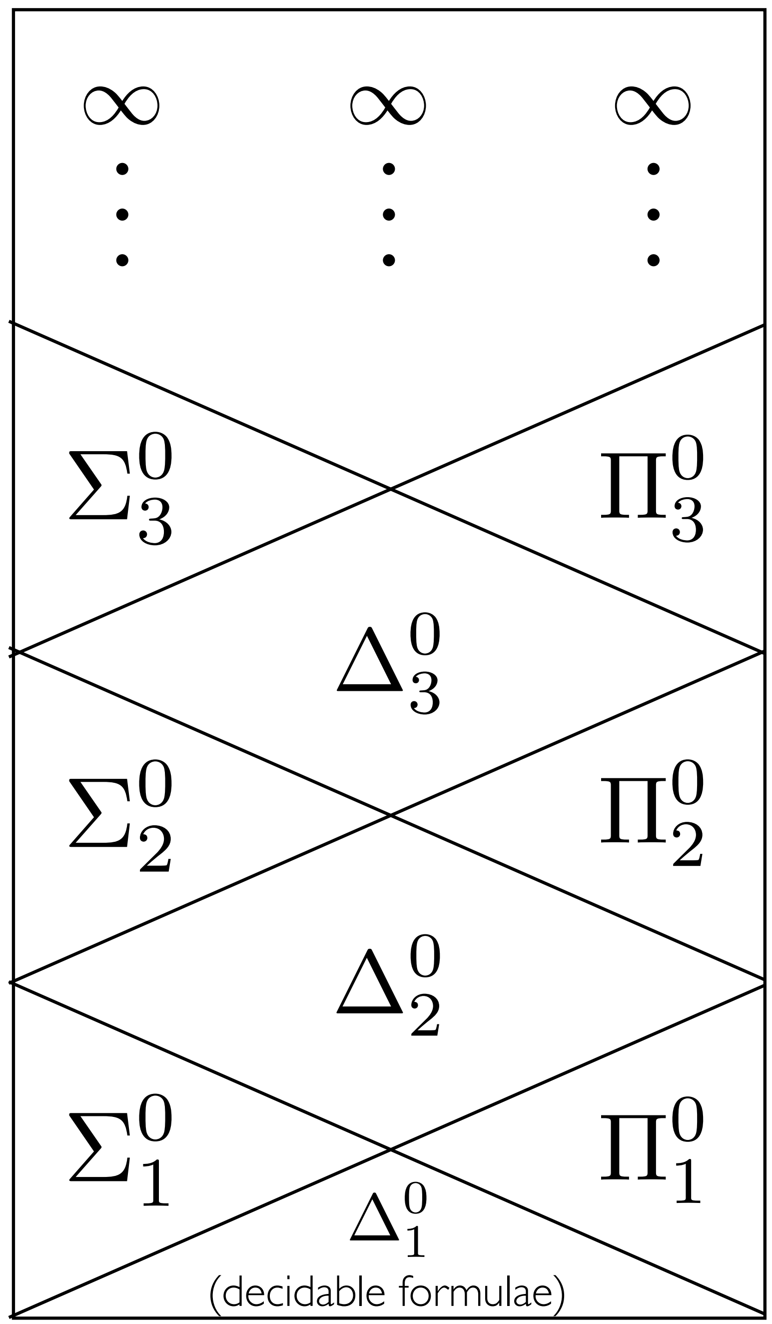
Arithmetic Hierarchy, Part I



Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



semi-decidable

$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

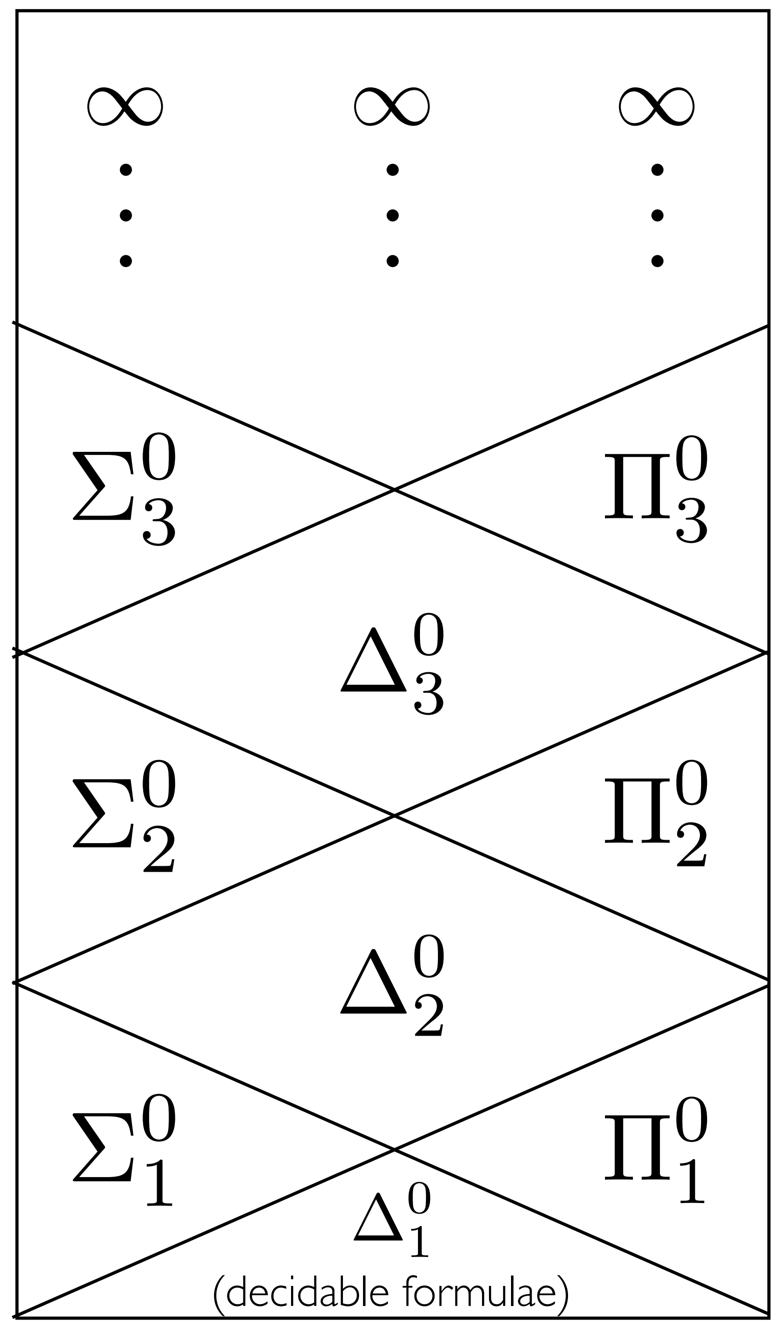
From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u \rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u \rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)

semi-decidable



$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! ...

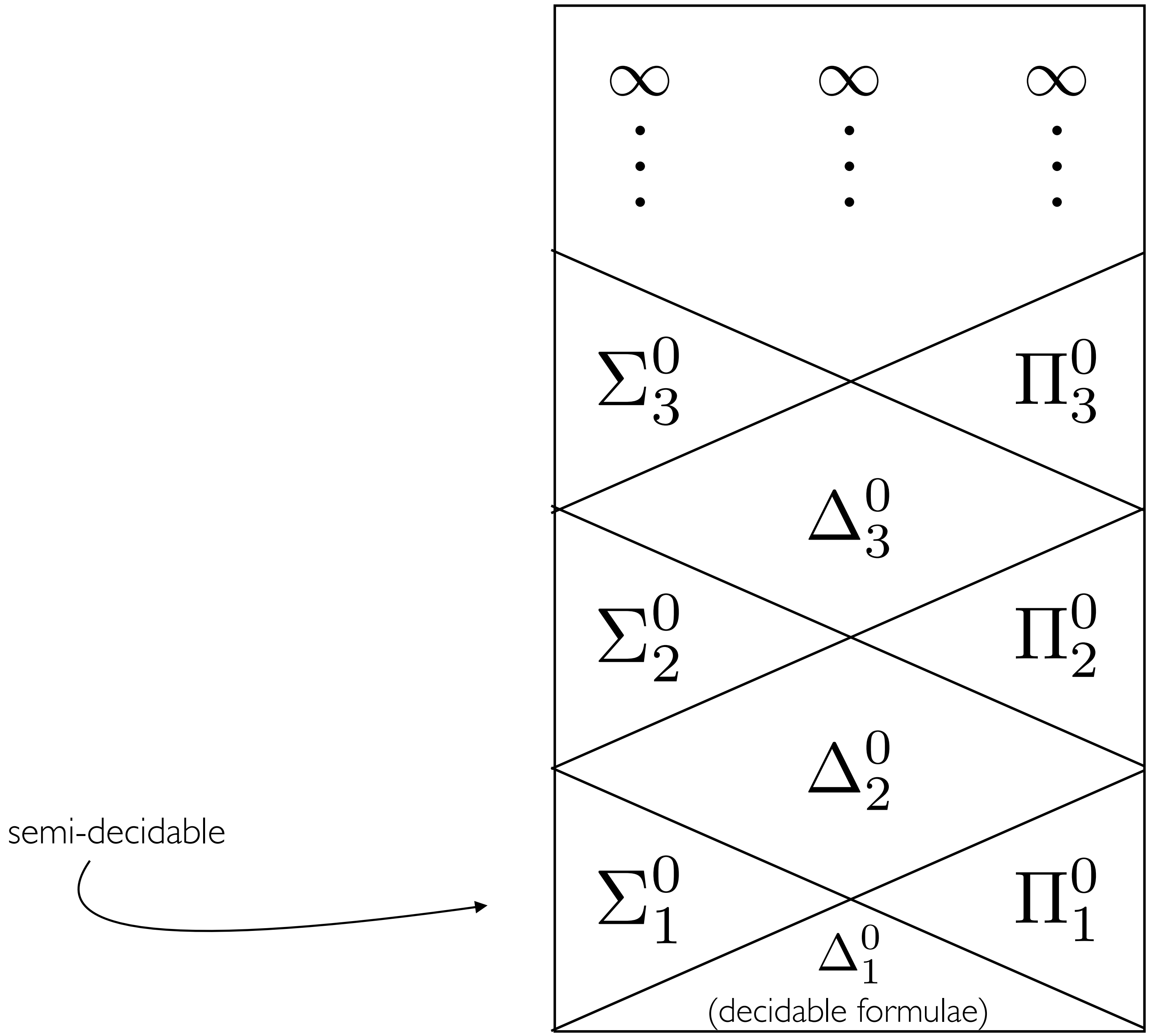
From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

Arithmetic Hierarchy, Part I

Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u \rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u \rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

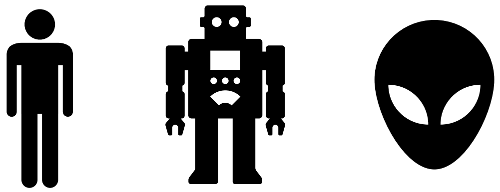
$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

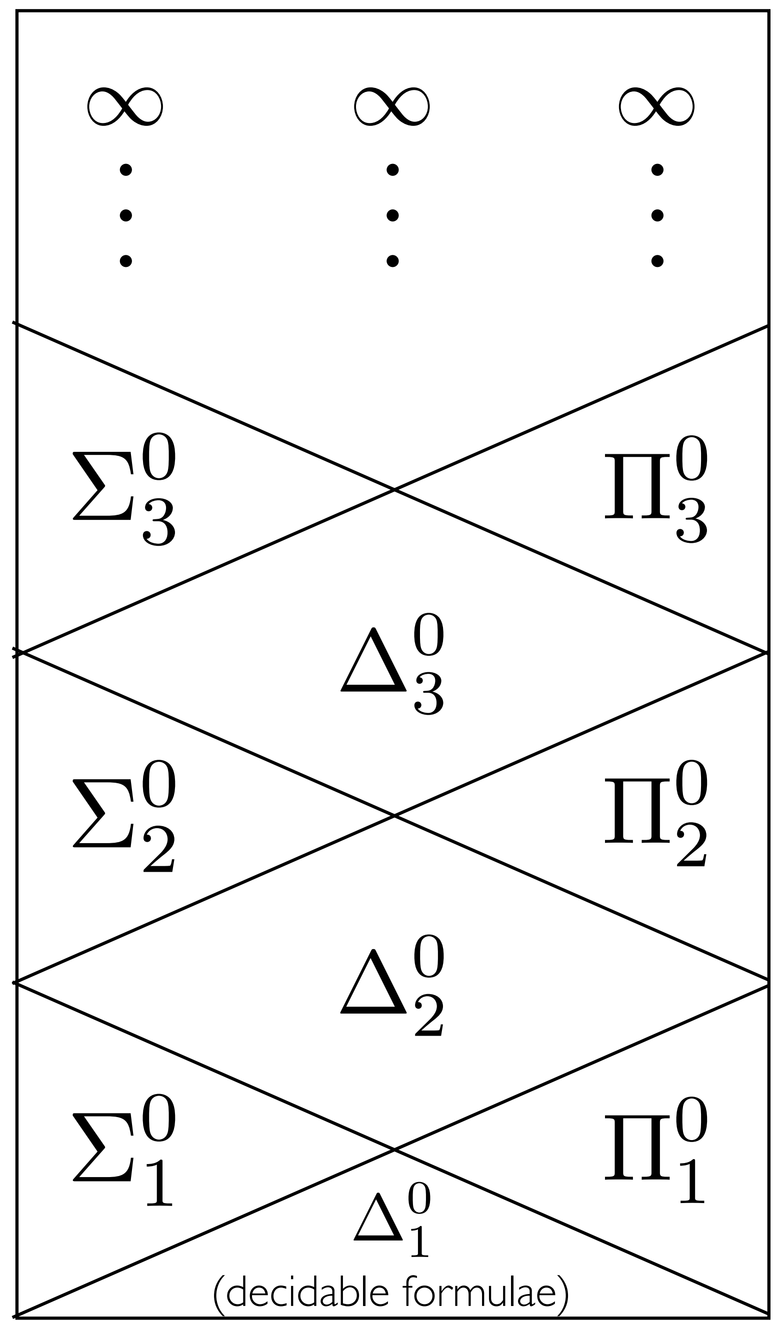
Arithmetic Hierarchy, Part I



Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k(\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k'(\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



semi-decidable

$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

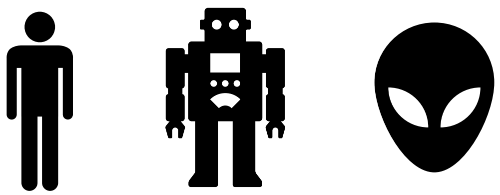
$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

The set to be classified is the set of all pairs of programs P_1 and P_2 s.t. both compute exactly the same functions.

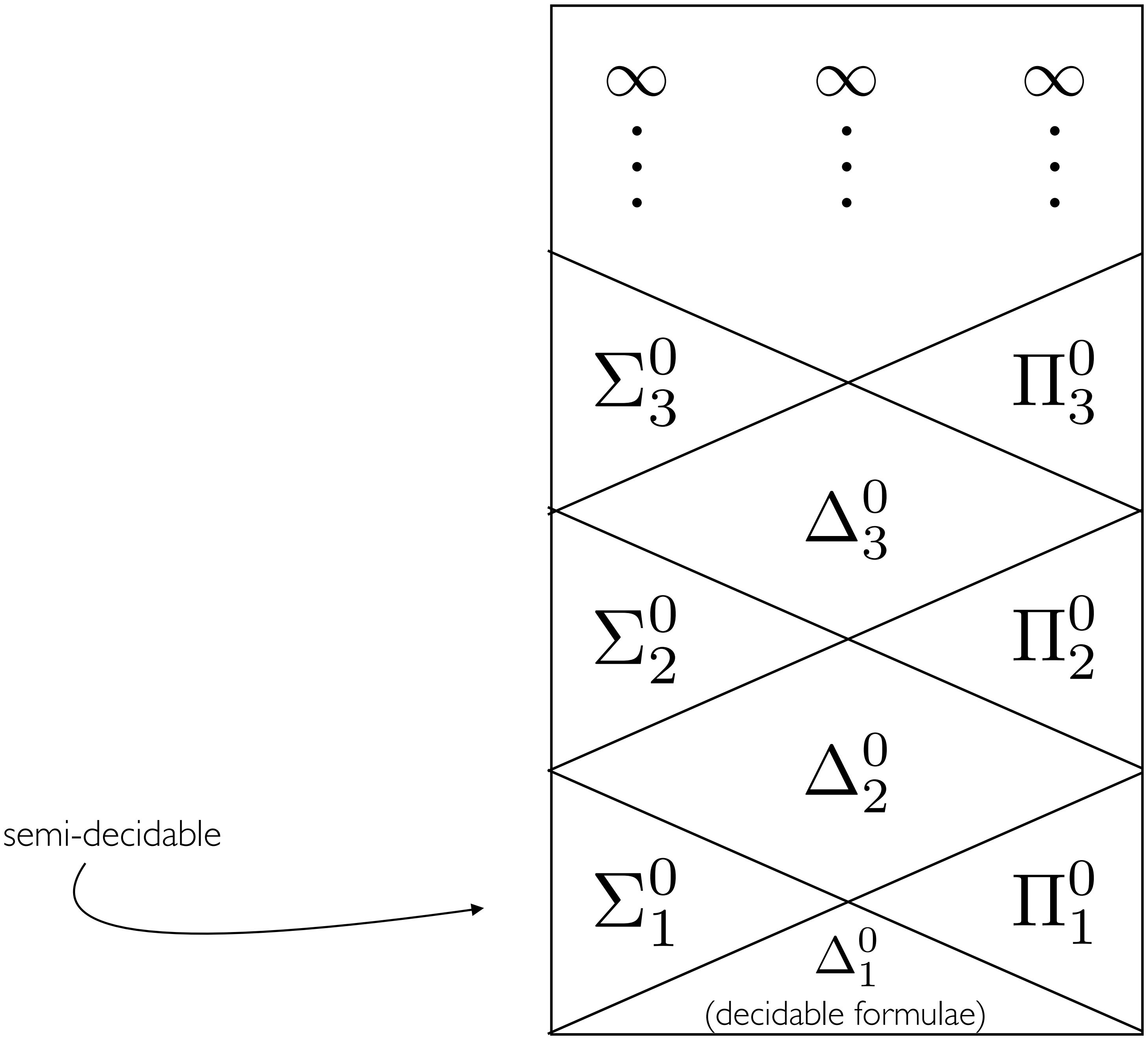
Arithmetic Hierarchy, Part I



Can you see the carryover from PH?

$$\mathbf{2SAMEFUNC} := \{\mathfrak{m}_1, \mathfrak{m}_2 : \forall u \forall v [\exists k (\langle \mathfrak{m}_1, u\rangle : v, k \leftrightarrow \exists k' (\langle \mathfrak{m}_2, u\rangle : v, k')]$$

$\mathcal{A}^r\mathcal{H}$ (Arithmetic Hierarchy)



$$x \in \Sigma_i \text{ iff } \exists R\ \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \forall \text{ if } i \text{ even}; Q_i = \exists \text{ if } i \text{ odd})$$

$$x \in \Pi_i \text{ iff } \exists R\ \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \ldots, y_i)$$
$$(Q_i = \exists \text{ if } j \text{ even}; Q_i = \forall \text{ if } j \text{ odd})$$

Try your hand at classifying! …

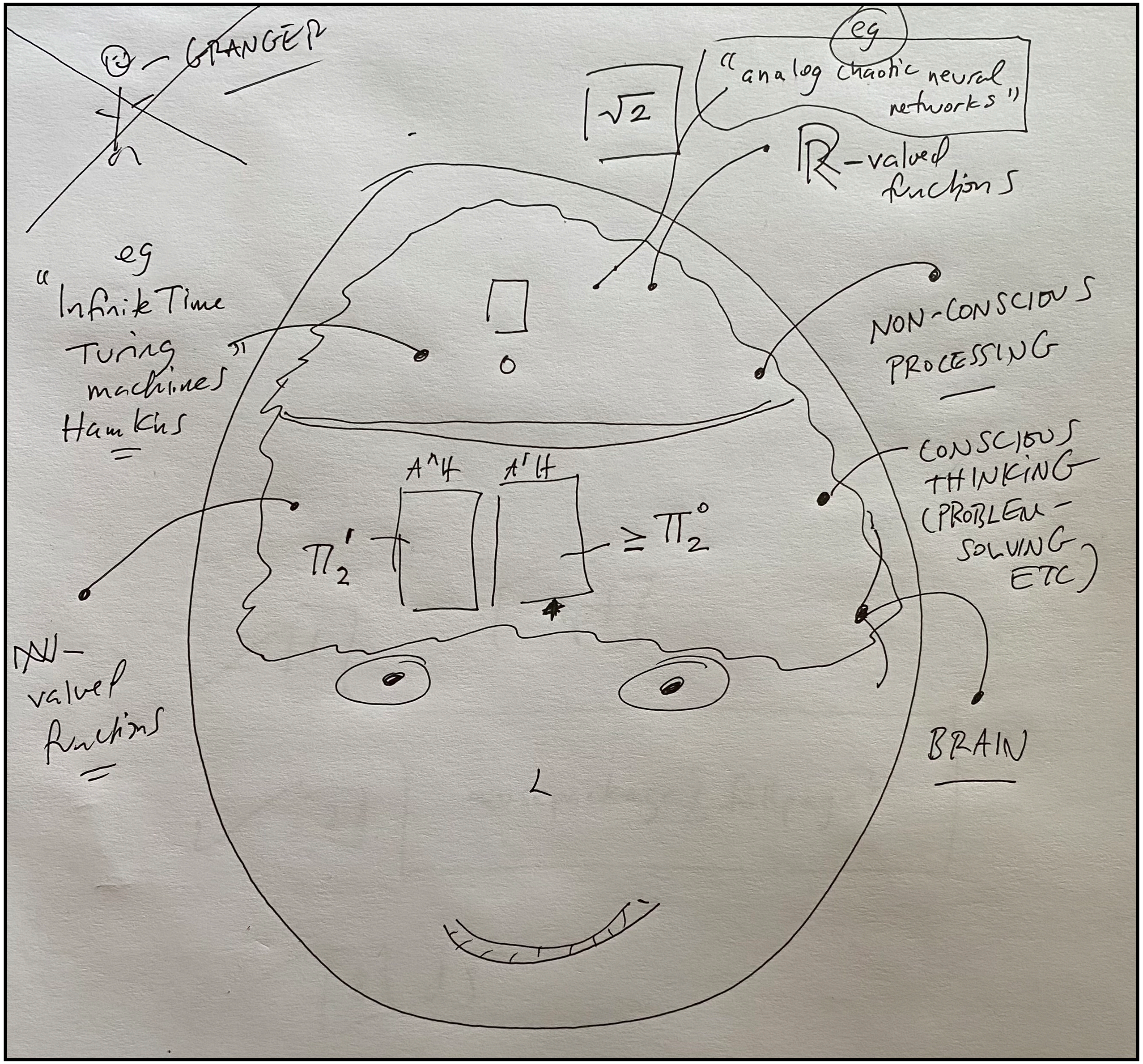
From Kleene: The set to be classified, $\mathscr{K}$, consists of all those inputs to a given Turing machine $\mathfrak{m}$ that results in this machine halting after some number of steps.

The set to be classified is the set of all pairs of programs P_1 and P_2 s.t. both compute exactly the same functions.

$$\Delta_1^0 = \Sigma_1^0 \cap \Pi_1^0$$

```
(forall (u v) (exists (k1 k2)
                   (iff (comp m1 u v k1)
                        (comp m2 u v k2))))
```

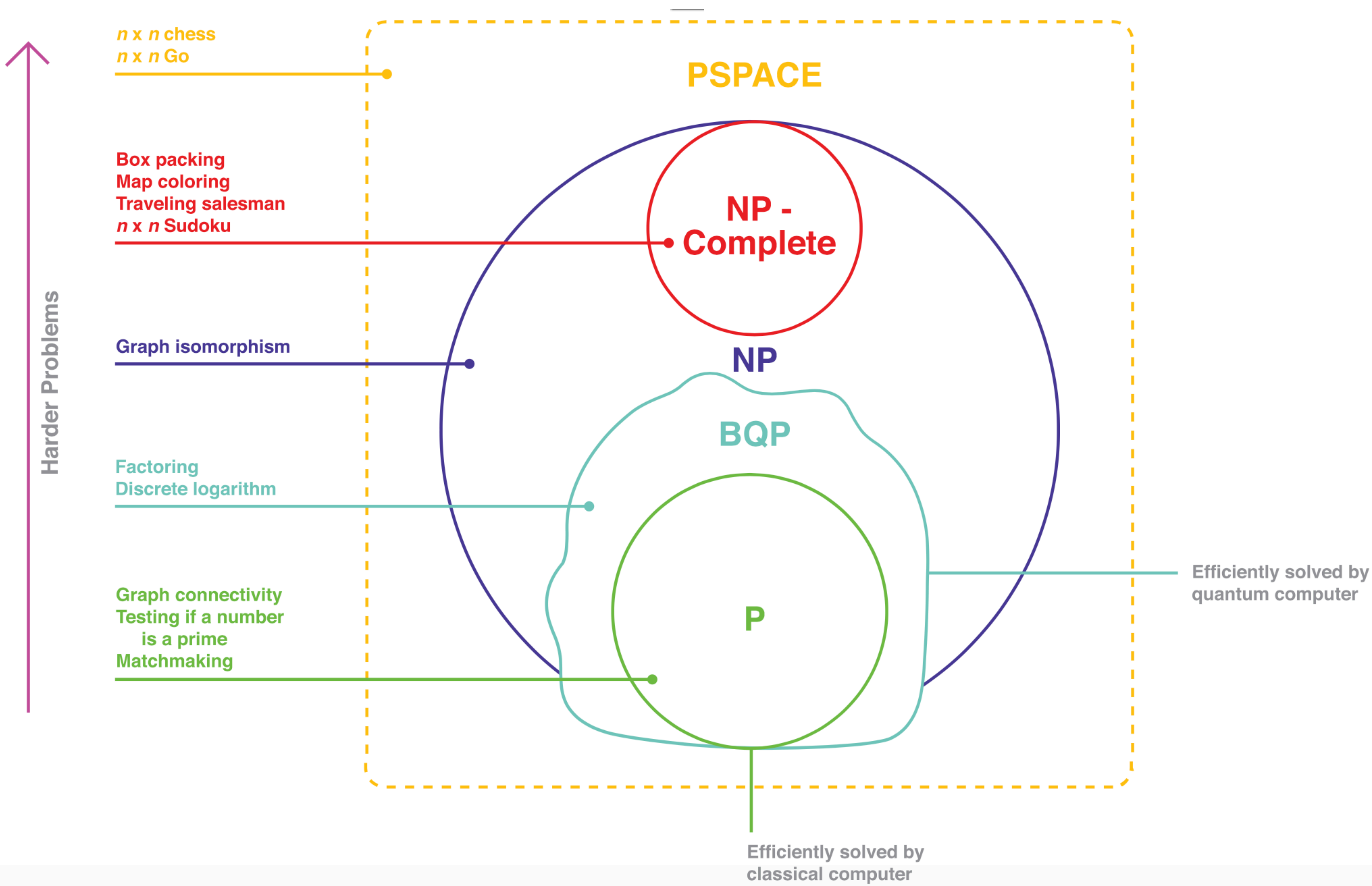
Arithmetic Hierarchy, Part I



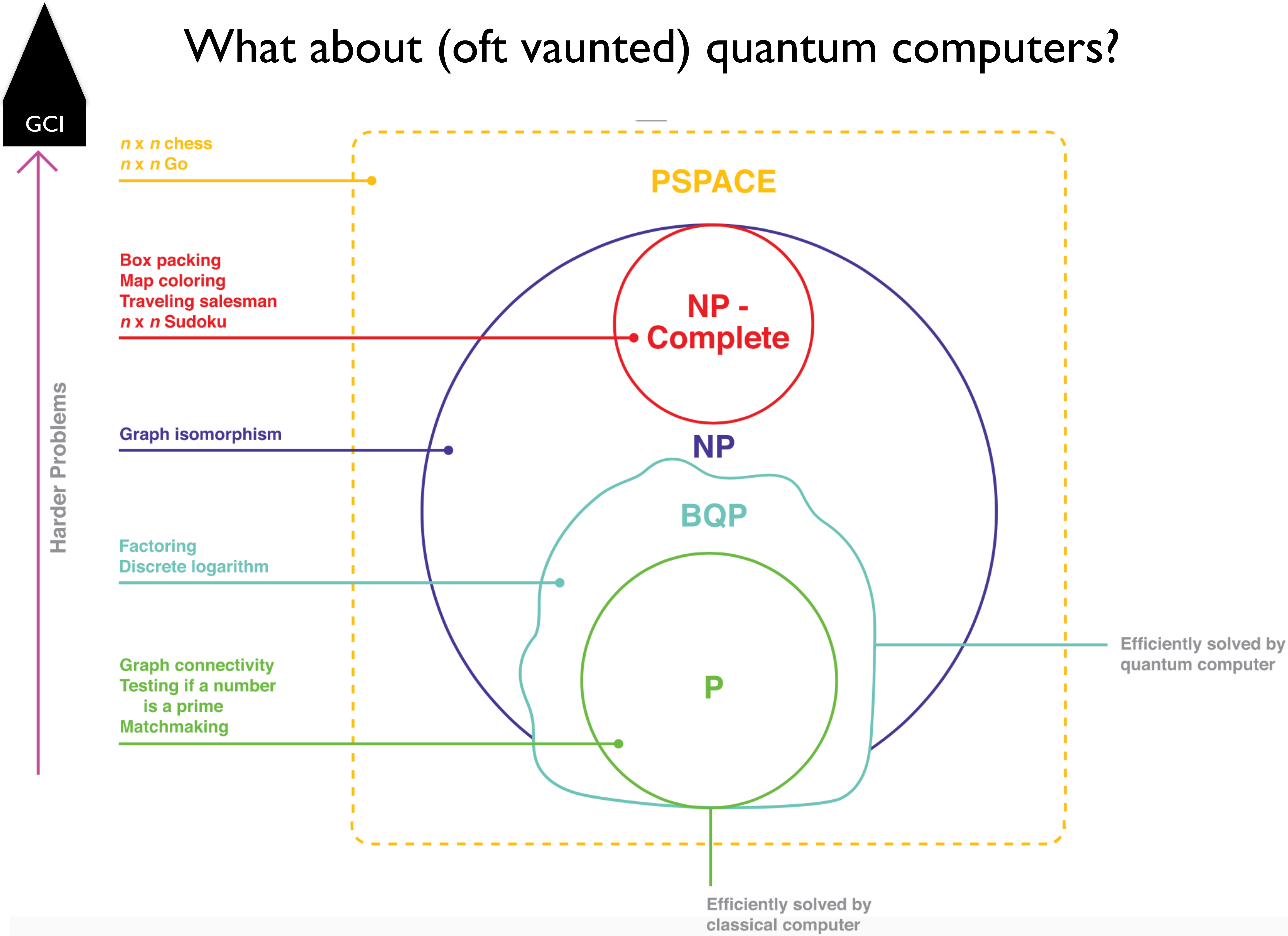
GRANGER
eg
"analog chaotic neural networks"
√2
ℝ-valued functions
eg
"Infinite Time Turing machines"
Hamkins
NON-CONSCIOUS PROCESSING
CONSCIOUS THINKING (PROBLEM-SOLVING ETC)
Δ^1_4
Δ^1_4
Π^1_2
≥ Π^0_2
ℕ-valued functions
BRAIN

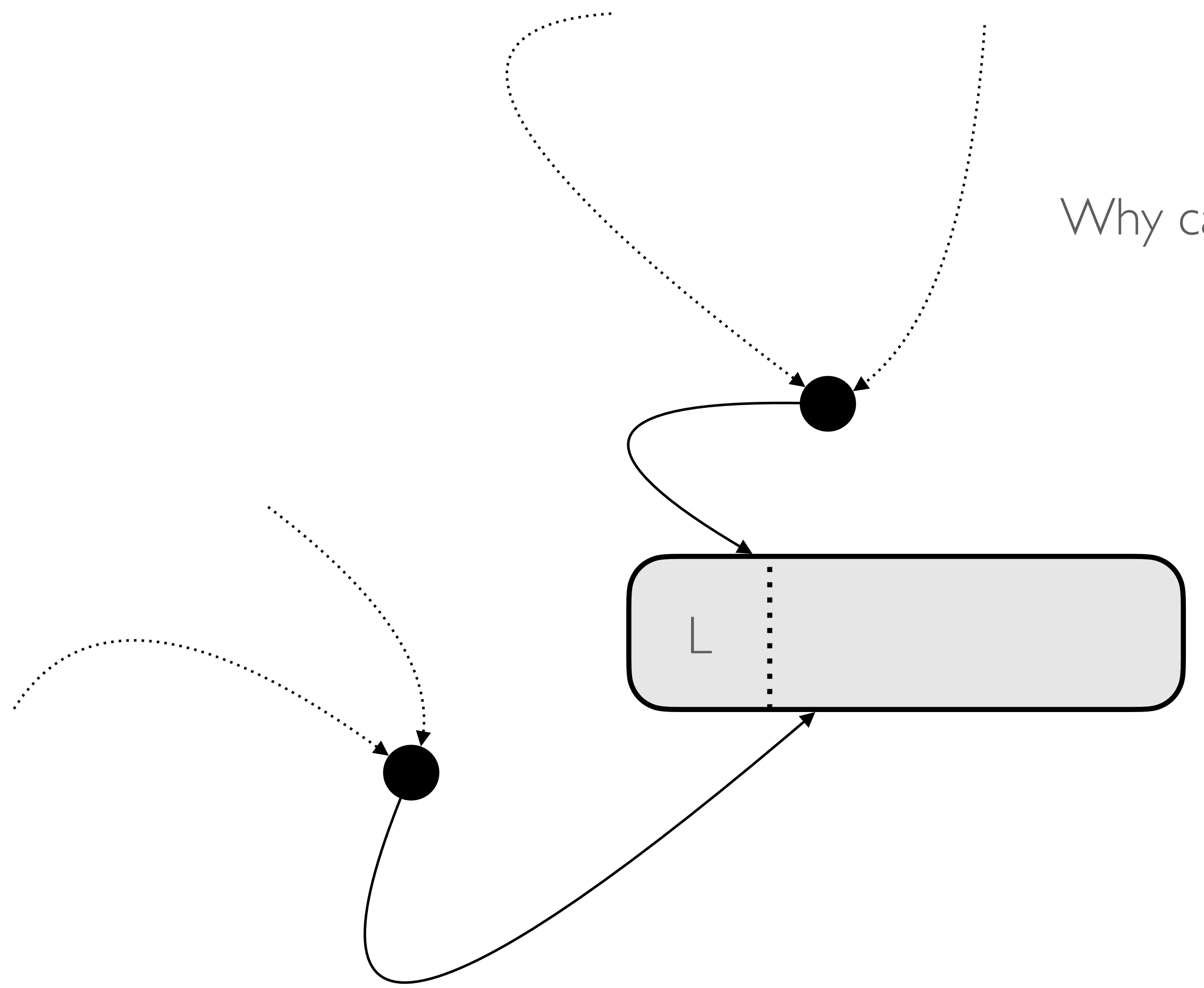
What about (oft vaunted) quantum computers?

What about (oft vaunted) quantum computers?



What about (oft vaunted) quantum computers?





Why can't a "neural" node be like tour nodes?