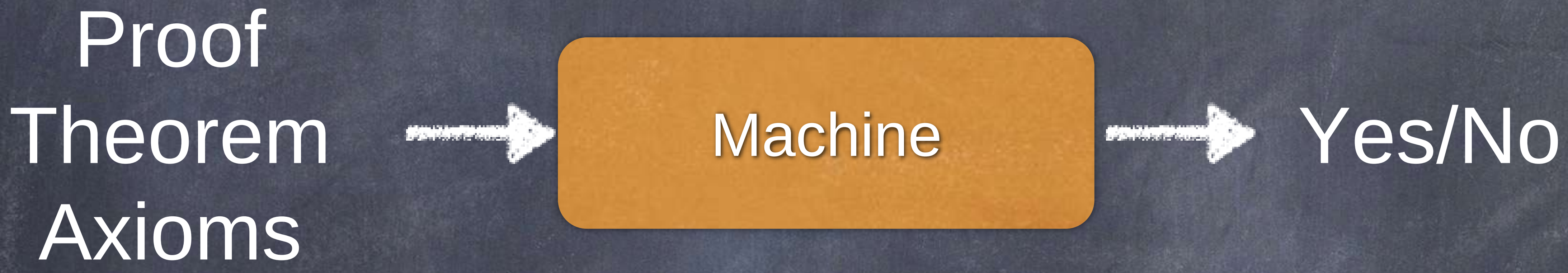
Toward a Computational Study of Time Travel

Atriya Sen*, Naveen Sundar G., Selmer Bringsjord
Rensselaer Polytechnic Institute
Troy, NY, USA

Hard, Harder, Hardest…

Hard: Proof Verification



Example

- Four Colour Theorem (1976, 2005)[1]
- Flyspeck Project (2014)
  - Verifies proof of the Kepler Conjecture
  - Uses HOL Light and Isabelle assistants

[1]Arkoudas, K. & Bringsjord, S. (2007) “Computers, Justification, and Mathematical Knowledge” Minds and Machines 17.2: 185–202.


Computers, Justification, and Mathematical Knowledge

Konstantine Arkoudas · Selmer Bringsjord

Published online: 23 June 2007

Abstract The original proof of the four-color theorem by Appel and Haken sparked a controversy when Tymoczko used it to argue that the justification provided by unsurveyable proofs carried out by computers cannot be a priori. It also created a lingering impression to the effect that such proofs depend heavily for their soundness on large amounts of computation-intensive custom-built software. Contra Tymoczko, we argue that the justification provided by certain computerized mathematical proofs is not fundamentally different from that provided by surveyable proofs, and can be sensibly regarded as a priori. We also show that the aforementioned impression is mistaken because it fails to distinguish between proof search (the context of discovery) and proof checking (the context of justification). By using mechanized proof assistants capable of producing certificates that can be independently checked, it is possible to carry out complex proofs without the need to trust arbitrary custom-written code. We only need to trust one fixed, small, and simple piece of software: the proof checker. This is not only possible in principle, but is in fact becoming a viable methodology for performing complicated mathematical reasoning. This is evinced by a new proof of the four-color theorem that appeared in 2005, and which was developed and checked in its entirety by a mechanical proof system.

Keywords A priori · Justification · Proofs · Certificates · Four-color theorem · Mathematical knowledge

K. Arkoudas (✉) · S. Bringsjord
Cognitive Science Department, Computer Science Department, RPI, Troy, NY, USA 12180
e-mail: arkouk@rpi.edu

S. Bringsjord
e-mail: brings@rpi.edu

Harder: Proof Discovery

Examples (first proofs)

- Semi-automated
  - Gödel's First Incompleteness Theorem[1] (2013, RAIRL)
- Fully automated
  - Robbins Conjecture in Otter (1997)

[1]Licato, John, et al. "Analogico-deductive generation of Gödel's first incompleteness theorem from the liar paradox." Proceedings of the Twenty-Third international joint conference on Artificial Intelligence. AAAI Press, 2013.

Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence

Analogico-Deductive Generation of Gödel's First Incompleteness Theorem from the Liar Paradox*

John Licato, Naveen Sundar Govindarajulu,
Selmer Bringsjord, Michael Pomeranz, Logan Gittelson
Rensselaer Polytechnic Institute (RPI)
Troy NY USA
{licatj,govinn,selmer,pomerm,gittel}@rpi.edu

Abstract

Gödel's proof of his famous first incompleteness theorem (**G1**) has quite understandably long been a tantalizing target for those wanting to engineer impressively intelligent computational systems. After all, in establishing **G1**, Gödel did something that by any metric must be classified as stunningly intelligent. We observe that it has long been understood that there is some sort of analogical relationship between the Liar Paradox (**LP**) and **G1**, and that Gödel himself appreciated and exploited the relationship. Yet the exact nature of the relationship has hitherto not been uncovered, by which we mean that the following question has not been answered: Given a description of **LP**, and the suspicion that it may somehow be used by a suitably programmed computing machine to find a proof of the incompleteness of Peano Arithmetic, can such a machine, provided this description as input, produce as output a complete and verifiably correct proof of **G1**? In this paper, we summarize engineering that entails an affirmative answer to this question. Our approach uses what we call *analogico-deductive reasoning* (ADR), which combines analogical and deductive reasoning to produce a full deductive proof of **G1** from **LP**. Our engineering uses a form of ADR based on our META-R system, and a connection between the Liar Sentence in **LP** and Gödel's Fixed Point Lemma, from which **G1** follows quickly.

1 Introduction

Gödel's proofs of his incompleteness theorems are among the greatest intellectual achievements of the 20th century. Even armed with the suggestion that the Liar Paradox (**LP**) might somehow serve as a guide to proving the incompleteness of Peano Arithmetic (**PA**),[1] the level of creativity and philosophical clarity required to actually tie the two concepts together and produce a valid proof is staggering; it certainly should not be controversial to claim that no computational reasoning system can, at present, achieve this sort of feat without significant human assistance.

1.1 Automating the Proof of G1

Prior work devoted to producing computational systems able to prove **G1** have yielded systems that manage to prove this theorem only when the distance between this result and the starting point is quite small. This for example holds for the first (and certainly seminal) foray; i.e., for [?], as explained in [?], where it's shown that the proof of **G1**, because the set of premises includes an ingenious human-devised encoding scheme, is very easy—to the point of being at the level of proofs requested from students in introductory mathematical logic classes.

Likewise, [?] is an exact parallel of the human-devised proof given by [?]. Finally, in much more recent and truly impressive work by [?], there is a move to natural-deduction formats, which we applaud—but the machine essentially begins its processing at a point exceedingly close to where it needs to end up. As Sieg and Field concede: "As axioms we take for granted the representability and derivability conditions for the central syntactic notions as well as the diagonal lemma for constructing self-referential sentences." If one takes for granted such things, finding a proof of **G1** is effortless for a computing machine.[2] In sum, while a lot of commendable work has been done to build the foundation for our prospective work, the daunting formal and engineering challenge of producing a computational system able to produce **G1** without clever seeding from a human remains entirely unmet.

2 The Analogico-Deductive Approach

2.1 Conjecture Generation

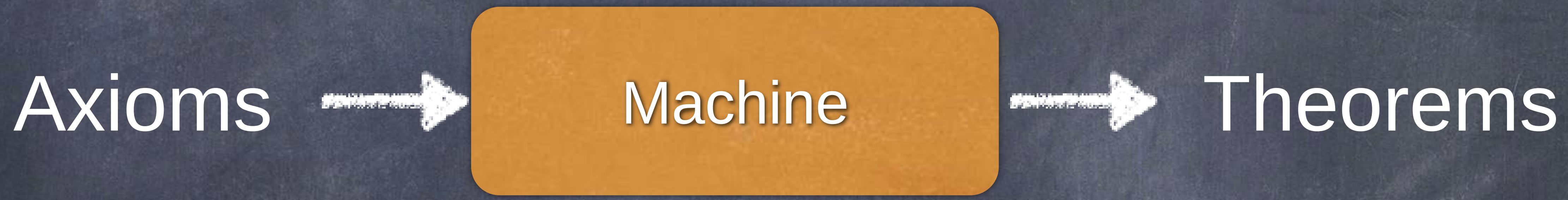
The problem with the purely deductive method is simply that it does not allow us to come close to the type of model-based reasoning that great thinkers are known to have used. Gödel himself has been described as having a "fine

*We are deeply grateful for penetrating feedback provided by three anonymous referees, and for financial support from AFOSR and the John Templeton Foundation.

[1]**G1** of course applies to any axiom system meeting the standard conditions (Turing-decidability, representability, consistency), but we tend to refer to **PA** for economization.

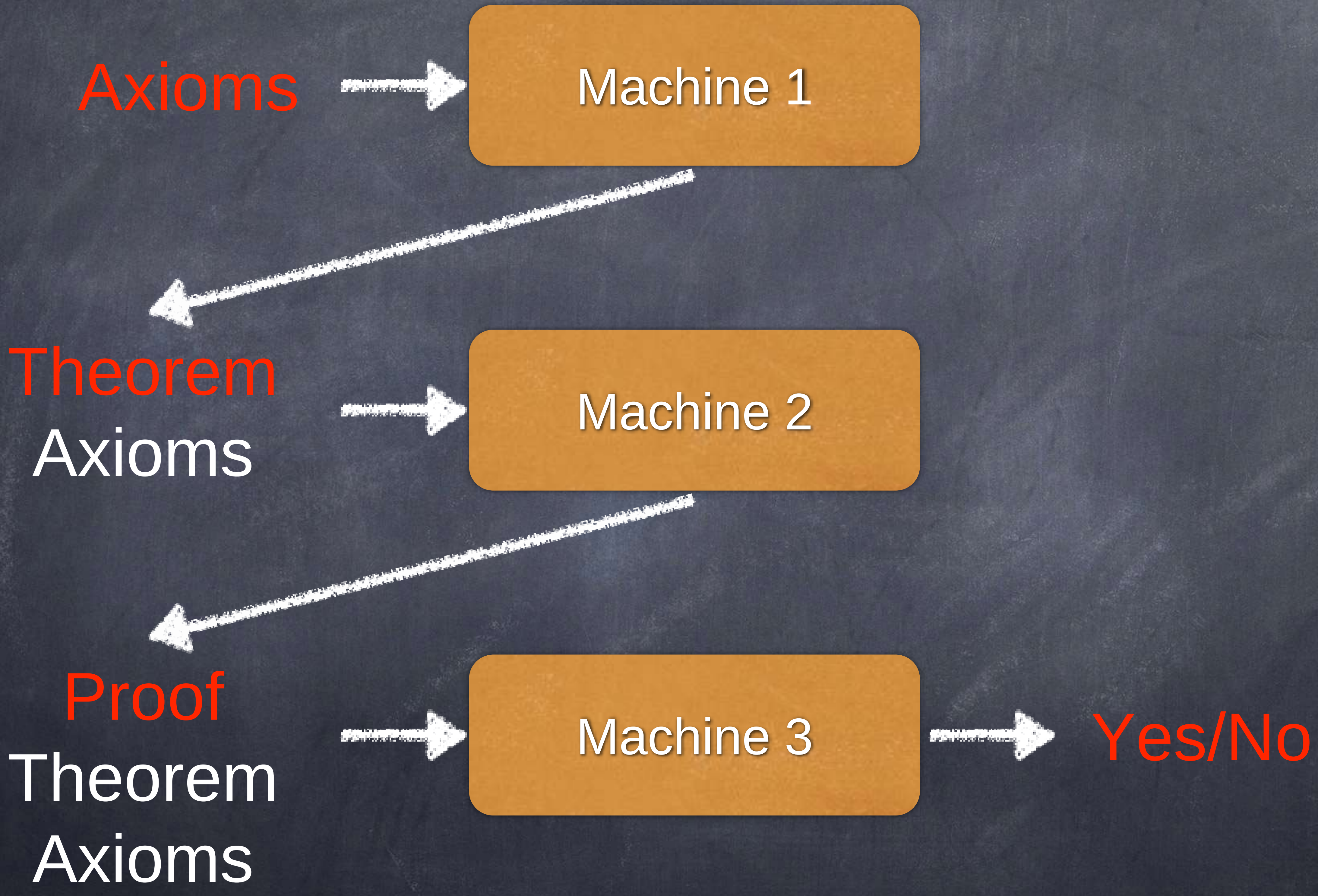
[2]A video demonstration of the small-distance process can be found at http://kryten.mm.rpi.edu/Godel1.abstract.in.Slate.mov.

Hardest: Theorem Discovery



Examples

- :(



Axioms
Machine 1
Theorem
Axioms
Machine 2
Proof
Theorem
Axioms
Machine 3
Yes/No

An Update since LRB15: Diagrammatic Reasoning

$$
\begin{array}{rcl}
D & ::= & \mathit{RuleApp} \\
 & | & \textbf{assume}\ F\ D \\
 & | & F\ \textbf{by}\ D \\
 & | & \mathfrak{D}; D \\
 & | & \textbf{pick-any}\ x\ \ D \\
 & | & \textbf{pick-witness}\ w\ \ \textbf{for}\ \exists\, x\, .\, F\ \ D \\
 & | & \textbf{specialize}\ \forall\, x_1 \cdots x_n\, .\, F\ \textbf{with}\ t_1, \ldots, t_n \\
 & | & \textbf{ex-generalize}\ \exists\, x\, .\, F\ \textbf{from}\ t \\
 & | & \textbf{cases by}\ F_1, \ldots, F_k\text{:}\ (\sigma_1; \rho_1) \rightarrow\ D_1 \mid \cdots \mid (\sigma_n; \rho_n) \rightarrow\ D_n \\
 & | & \textbf{observe}\ \ F
\end{array}
$$

$$
\begin{array}{rcl}
\Delta & ::= & \mathfrak{D}; \Delta \\
 & | & \textbf{claim}\ (\sigma; \rho) \\
 & | & (\sigma; \rho)\ \textbf{by thinning with}\ F_1, \ldots, F_n \\
 & | & (\sigma; \rho)\ \textbf{by widening} \\
 & | & (\sigma; \rho)\ \textbf{by absurdity} \\
 & | & \textbf{cases by}\ F_1, \ldots, F_k\text{:}\ (\sigma_1; \rho_1) \rightarrow\ \Delta_1 \mid \cdots \mid (\sigma_n; \rho_n) \rightarrow\ \Delta_n \\
 & | & \textbf{cases}\ F_1\ \vee\ F_2\text{:}\ F_1\ \rightarrow\ \Delta_1 \mid F_2 \rightarrow\ \Delta_2 \\
 & | & \textbf{pick-witness}\ w\ \ \textbf{for}\ \exists\, x\, .\, F\ \ \Delta
\end{array}
$$

$$
\mathfrak{D} \ ::= \ D \mid \Delta
$$

$$\frac{}{(\beta \cup \{F_1, \ldots, F_n\}; (\sigma; \rho)) \vdash (\sigma'; \rho') \textbf{ by thinning with } F_1, \ldots, F_n \leadsto (\sigma'; \rho')} \quad [\textit{Thinning}]$$

provided $(\sigma; \rho) \Vdash_{\{F_1, \ldots, F_n\}} (\sigma'; \rho')$

$$\frac{}{(\beta; (\sigma; \rho)) \vdash (\sigma'; \rho') \textbf{ by widening } \leadsto (\sigma'; \rho')} \quad [\textit{Widening}]$$

provided $(\sigma; \rho) \sqsubseteq (\sigma'; \rho')$

$$\frac{}{(\beta \cup \{\textbf{false}\}; (\sigma; \rho)) \vdash (\sigma'; \rho') \textbf{ by absurdity } \leadsto (\sigma'; \rho')} \quad [\textit{Absurdity}]$$

$$\frac{}{(\beta; (\sigma; \rho)) \vdash \textbf{claim } (\sigma; \rho) \leadsto (\sigma; \rho)} \quad [\textit{Diagram-Reiteration}]$$

$$\frac{\begin{array}{c}(\beta \cup \{F_1, \ldots, F_k\}; (\sigma_1; \rho_1)) \vdash \Delta_1 \leadsto (\sigma'; \rho') \\ \vdots \\ (\beta \cup \{F_1, \ldots, F_k\}; (\sigma_n; \rho_n)) \vdash \Delta_n \leadsto (\sigma'; \rho')\end{array}}{(\beta \cup \{F_1, \ldots, F_k\}; (\sigma; \rho)) \vdash \textbf{cases by } F_1, \ldots, F_k\text{: } (\sigma_1; \rho_1) \rightarrow \Delta_1 \mid \cdots \mid (\sigma_n; \rho_n) \rightarrow \Delta_n \leadsto (\sigma'; \rho')} \quad [C_1]$$

provided $(\sigma; \rho) \Vdash_{\{F_1, \ldots, F_k\}} \{(\sigma_1; \rho_1), \ldots, (\sigma_n; \rho_n)\}$

$$\frac{(\beta \cup \{F_1 \vee F_2, F_1\}; (\sigma; \rho)) \vdash \Delta_1 \leadsto (\sigma'; \rho') \qquad (\beta \cup \{F_1 \vee F_2, F_2\}; (\sigma; \rho)) \vdash \Delta_2 \leadsto (\sigma'; \rho')}{(\beta \cup \{F_1 \vee F_2\}; (\sigma; \rho)) \vdash \textbf{cases } F_1 \vee F_2\text{: } F_1 \rightarrow \Delta_1 \mid F_2 \rightarrow \Delta_2 \leadsto (\sigma'; \rho')} \quad [C_2]$$

$$\frac{(\beta; (\sigma; \rho)) \vdash D \leadsto F \qquad (\beta \cup \{F\}; (\sigma; \rho)) \vdash \Delta \leadsto (\sigma'; \rho')}{(\beta; (\sigma; \rho)) \vdash D; \Delta \leadsto (\sigma'; \rho')} \quad [D; \Delta]$$

$$\frac{(\beta; (\sigma; \rho)) \vdash \Delta \leadsto (\sigma'; \rho') \qquad (\beta; (\sigma'; \rho')) \vdash D \leadsto F}{(\beta; (\sigma; \rho)) \vdash \Delta; D \leadsto F} \quad [\Delta; D]$$

$$\frac{(\beta; (\sigma; \rho)) \vdash \Delta_1 \leadsto (\sigma_1; \rho_1) \qquad (\beta; (\sigma_1; \rho_1)) \vdash \Delta_2 \leadsto (\sigma_2; \rho_2)}{(\beta; (\sigma; \rho)) \vdash \Delta_1; \Delta_2 \leadsto (\sigma_2; \rho_2)} \quad [\Delta; \Delta]$$

$$\frac{(\beta; (\sigma; \rho)) \vdash D_1 \leadsto F_1 \qquad (\beta \cup \{F_1\}; (\sigma; \rho)) \vdash D_2 \leadsto F_2}{(\beta; (\sigma; \rho)) \vdash D_1; D_2 \leadsto F_2} \quad [D; D]$$

$$\frac{(\beta \cup \{\exists\, x\,.\, F, F\,[z/x]\}; (\sigma; \rho)) \vdash \Delta\,[z/w] \leadsto (\sigma'; \rho')}{(\beta \cup \{\exists\, x\,.\, F\}; (\sigma; \rho)) \vdash \textbf{pick-witness } w \textbf{ for } \exists\, x\,.\, F \;\; \Delta \leadsto (\sigma'; \rho')} \quad [\textit{EI}/\Delta]$$

provided z is fresh

Theorem Clocks: Moving Clocks Go Out of Sync

$\mathbf{\Delta}_0$ ➔ $\mathbf{\Delta}_1$ by [Diagram − Reiteration]:
observe (meets(i1,m1,m2) ∧ ¬in same frame(v1,v2))
observe in same frame(v2,v3)
observe in same frame(v2,v4)
observe in same frame(v3,v4)

$\mathbf{\Delta}_1$ ➔ $\mathbf{\Delta}_2$ by [Diagram − Reiteration]:
observe (meets(p, c1, m3) ∧ speed of light(c1)

$\mathbf{\Delta}_2$ ➔ $\mathbf{\Delta}_3$ by [Thinning] with can observe photon at definition

$\mathbf{\Delta}_3$ ➔ $\mathbf{\Delta}_4$ by [Diagram − Reiteration]:
observe (meets(q, c2, m4) ∧ speed of light(c2)

$\mathbf{\Delta}_4$ ➔ $\mathbf{\Delta}_5$ by [Thinning] with the can observe photon at definition:
observe not same location(q, q′) ()where q′ = (2, 0))
observe not same location(p, q)
observe (meets(i1, m2, c1) ∧ meets(p, c1, c2) ∧ meets(i2, c2, m4) ∧ speed of light(c1) ∧ speed of light(c2)) observe clocks unequal(i1, i2)

Verified

The Event Calculus

Abductive:
What happens when
(Explanation, planning)

Logical
Machinery

Deductive:
What’s true
when
(Prediction)

Inductive:
What actions do
(Learning)

- Formalism for reasoning about action and change
- Logical mechanism for inferring what’s true given what happens when, and what actions do
- Single time-line on which events occur
- Can be first-order or modal
- Ontology: events, fluents, time points
- (Can support context-sensitive effects of events, indirect effects, action preconditions, partially-ordered events, etc.)

Example

- The effects of events:
  - If the light's switch is flipped up, then the light will be on
  - If the light's switch is flipped down, then the light will be off
- A specific scenario:
  - The light was off at time 0
  - Then the switch was flipped up at time 5
  - Then the switch was flipped down at time 8
- We can now conclude:
  - At time 3, the light was off
  - At time 7, it was on
  - At time 10, it was off

The Relativistic Event Calculus (REC)

REC: Sorts

- Quantities
- Bodies
  - Observers
  - Photons
- Fluents
- Events

REC: Predicates

- Initially(m,f): Observer m observes fluent f at origin
- HoldsAt(m,f,x): Fluent f holds at x in m’s FOR
- Happens(m,e,x): Event e happens at x in m’s FOR
- Initiates(m,e,x,f,x’): Event e happens at x in m’s FOR, initiating a fluent f at x’ in m’s FOR
    - x and x’ constrained by STR and causality
- Terminates(m,e,x,f,x’): Event e happens at x in m’s FOR, terminating a fluent f at x’ in m’s FOR
- StoppedIn(m,x,f,x’): The fluent f is stopped somewhere between x and x’ in m's FOR

REC: Axioms

- REC1: $(\text{Initially}(m,f) \land \neg\text{StoppedIn}(m,0,f,x)) \lor \exists e,x',x'' (\text{Happens}(m,e,x') \land \text{Initiates}(m,e,x',f,x'') \land \neg\text{StoppedIn}(m,x'',f,x))) \equiv \text{HoldsAt}(m,f,x)$
- REC2: $\text{StoppedIn}(m,x_1,f,x_2) \equiv \exists e,x (\text{Happens}(m,e,x_1) \land \text{Terminates}(e,f,x) \land \text{Between}(m,x_1,x,x_2))$

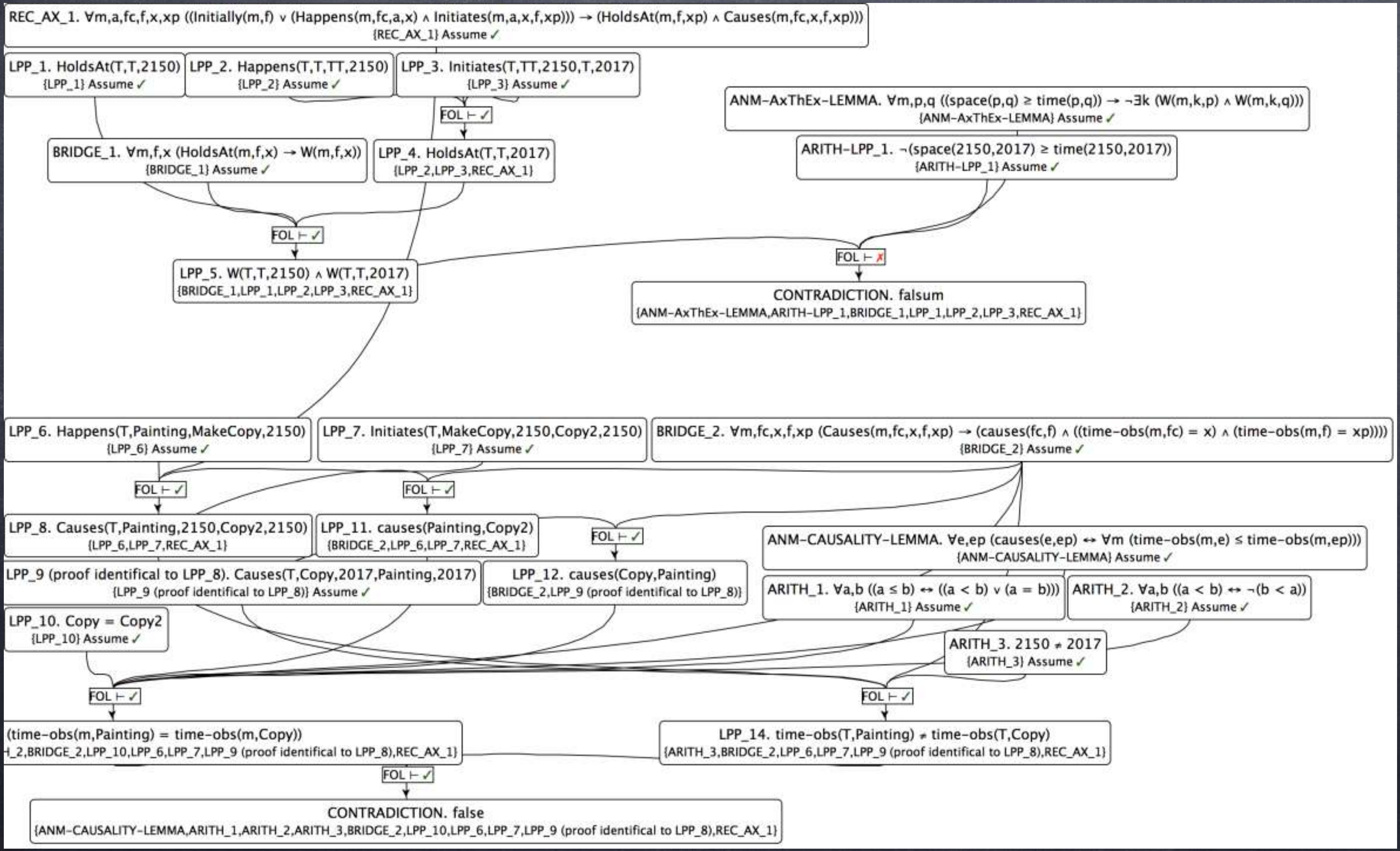
REC: STR

- Initially(m,f) → W(m,f,0); Holds(m,f,x) → W(m,f,x); Happens(m,e,x) → W(m,e,x)
- Lines connecting events with fluents:
  - Are straight (AxLine-REC)
  - Have slope lesser or equal to 1 (AxThEx-REC, AxLine-REC)

REC: Causality

- REC1-C: (Initially(m,f) ∧ ¬StoppedIn(m,0,f,x)) ∨ ∃e,x',x'' (Happens(m,f_p,e,x') ∧ Initiates(m,e,x',f,x'') ∧ ¬StoppedIn(m,x'',f,x))) ≡ HoldsAt(m,f,x) ∧ Causes(m,f_p,x',f,x)
- In $Specrel_0$ + Ax↑, event e casually precedes event e' iff (∀m∈Ob)($time_m$(e) ≤ $time_m$(e'))

Specrel + REC + LP is inconsistent



Köszönöm szépen