Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 312 (2015) 143–160

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Practical Extraction of Evidence Terms From Common-knowledge Reasoning

Natalia Novak

Mathematics and Computer Science Department,
Bronx Community College, CUNY,
Bronx NY 10453, USA
Natalia.Novak@bcc.cuny.edu

Abstract

Knowledge, belief, and evidence are fundamental notions which appear in a wide range of areas. Over the last decade epistemic reasoning with justifications has broadened even more the scope of applications of epistemic logic as agents gained the ability to not only reason about epistemic states of knowledge and belief of agents, but also to track their justifications and to sort those which are pertinent to given facts and sufficient for epistemic conclusions.

This paper extends realization algorithm for S4-to-LP case to $\mathsf{S4}^{\mathsf{J}}_n$-to-$\mathsf{S4}_n\mathsf{LP}$ case. It converts cut-free derivations in $\mathsf{S4}^{\mathsf{J}}_n$ into derivations in the corresponding Justification Logic $\mathsf{S4}_n\mathsf{LP}$ where witnesses of knowledge, the justification terms, are recovered for all instances of justified common knowledge. The algorithm was implemented in the MetaPRL framework and was tested on several well-known epistemic puzzles, such as Muddy Children, Surprise Examination Paradox, etc.

Keywords: logic of proofs, logical puzzles, evidence terms, metaprl

Introduction

The study of epistemic reasoning, reasoning about knowledge and belief, is one of the core areas of Computer Science and Artificial Intelligence. The traditional systems of formal epistemology are based on modal logics and have been the subjects of intense research activity during the past decades [10; 15]. There are several computer-aided systems of modal and epistemic reasoning available (for an incomplete list, see [17]).

A foundational effort in this area has enriched modal epistemic logic with the internalized notion of justification, which became part of the language of epistemic logic. This development substantially broadens the scope of applications of epistemic logic. We now have the capability to not only reason about epistemic states of knowledge and belief of agents, but also to track their justifications and to sort those which are pertinent to given facts and sufficient for epistemic conclusions. The very notion of *evidence* has become the subject of rigorous studies.

http://dx.doi.org/10.1016/j.entcs.2015.04.009

The Artemov's Realization Theorem [2; 3] is the fundamental result that reveals the robust evidence system behind traditional epistemic modal logic reasoning. It recovers evidence terms for each occurrence of epistemic modality in a given theorem. We started with the implementation of improved Artemov's Realization Theorem within the framework of the MetaPRL computer-aided reasoning system, then proceeded with test runs on a wide range of well-known epistemic problems.

1 Translation of $\mathsf{S4}_n^{\mathsf{J}}$ cut-free proofs into $\mathsf{S4}_n\mathsf{LP}$ proof

1.1 Overview of $\mathsf{S4}_n\mathsf{LP}$ *logic*

$\mathsf{S4}_n\mathsf{LP}$ [4] is a multi-agent logic of evidence-based knowledge, with knowledge operators of n agents $K_1, K_2, K_3, \ldots, K_n$, acting as $\mathsf{S4}$ modalities [10], and evidence assertions of the form $t : A$, where t is an evidence term and A is a formula, as in LP [3]. Evidence term t is built from constants $a, b, c, \ldots$ and variables $x, y, z, \ldots$ with the help of binary operators '$\cdot$' (application), '$+$' (union), and unary operator '!' (inspection).

Formulas of $\mathsf{S4}_n\mathsf{LP}$ are defined by the following grammar:
$\bot \mid S \mid A \to B \mid A \wedge B \mid A \vee B \mid \neg A \mid K_i A \mid t : A$, where t is an evidence and S is a sentence variable.

Evidence operation has highest precedence and all other connectives have standard precedence order.

Hilbert-style axioms and rules of $\mathsf{S4}_n\mathsf{LP}$ contain classical propositional logic axioms with the *Modus Ponens* rule along with

Knowledge principles

$\mathrm{B1}_i$. $K_i(A \to B) \to (K_i A \to K_i B)$
$\mathrm{B2}_i$. $K_i A \to A$
$\mathrm{B3}_i$. $K_i A \to K_i K_i A$ (*positive introspection*)
$\mathrm{R2}_i$. $A \vdash K_i A$ (*knowledge generalization*)
for each individual knowledge operator K_i.

Evidence Principles

E1. $s : (A \to B) \to (t : A \to (s \cdot t) : B)$ (*application*)
E2. $t : A \to !t : (t : A)$ (*inspection*)
E3. $s : A \to (s + t) : A, \quad t : A \to (s + t) : A$ (*union*)
E4. $t : A \to A$ (*reflexivity*)
R3. $\vdash c : A$, where A is an $\mathsf{S4}_n\mathsf{LP}$ axiom and c is a proof constant (*evidence for axioms*).

Principle connecting evidence and knowledge

C1. $t : A \to K_i A$ (*undeniability of evidence*).

All axioms are schemas in the language of $\mathsf{S4}_n\mathsf{LP}$. Rules are applied across all sections. The system is closed under substitutions of evidence terms for evidence variables and formulas for propositional variables. Deduction theorem $\Gamma, A \vdash B \Rightarrow \Gamma \vdash A \to B$ holds, where Γ is a finite set of $\mathsf{S4}_n\mathsf{LP}$ formulas.

The following two lemmas will be used in the proof of realization algorithm presented later in the text:

Lemma 1.1 (Lifting Lemma) *[3; 4]* If $A_1, \ldots, A_n, y_1 : B_1, \ldots, y_m : B_m \vdash F$, then for some evidence term $t = t(x_1, \ldots, x_n, y_1, \ldots, y_m)$,

$$x_1 : A_1, \ldots, x_n : A_n, y_1 : B_1, \ldots, y_m : B_m \vdash t(x_1, \ldots, x_n, y_1, \ldots, y_m) : F.$$

Lemma 1.2 *[5] For any proof variables $\boldsymbol{x_i}$ and any formulas $\boldsymbol{B_i}$, there exists a proof term $s = s(x_1, \ldots, x_n)$ such that*

$$\mathsf{LP} \vdash x_1 : B_1 \wedge \ldots \wedge x_n : B_n \to s(x_1, \ldots, x_n) : (x_1 : B_1 \wedge \ldots \wedge x_n : B_n).$$

1.2 Overview of $\mathsf{S4}^{\mathsf{J}}_n$ system

$\mathsf{S4}^{\mathsf{J}}_n$ [4; 5] is a forgetful evidence-based logic with $n+1$ modalities $K_1, \ldots, K_n, J$. JA reads as 'A is justified' and is a forgetful projection of evidence assertion $t : A$.

The dummy $n+1^{th}$ agent corresponding to J plays the role of a skeptical and not logically omniscient $\mathsf{S4}$-agent who accepts facts only if they are supplied with checkable evidence. This agent is trusted by all other agents and is capable of internalizing and inspecting any fact actually proven in the system.

The forgetful version of *undeniability of evidence* principle for $\mathsf{S4}^{\mathsf{J}}_n$ is $JA \to K_i A$, for all $i = 1, \ldots, n$.

Below we present a Gentzen-style formulation [18] of $\mathsf{S4}^{\mathsf{J}}_n$ called $\mathsf{S4}^{\mathsf{J}}_n\mathsf{G}$.

A *sequent* is a pair of finite sets of $\mathsf{S4}^{\mathsf{J}}_n$ formulas presented as $\Gamma \Rightarrow \Delta$. Axioms of Gentzen-style $\mathsf{S4}^{\mathsf{J}}_n$ are the sequents $S, \Gamma \Rightarrow \Delta, S$ and $\bot, \Gamma \Rightarrow \Delta$, where S is a propositional variable.

$\mathsf{S4}^{\mathsf{J}}_n$ is strictly weaker than system $\mathsf{S4}^{\mathsf{C}}_n$ [1], where common knowledge C is defined using traditional fixpoint common knowledge [10]. Nevertheless, $\mathsf{S4}^{\mathsf{J}}_n$ seems sufficient for all practical applications. At the same time, $\mathsf{S4}^{\mathsf{J}}_n$ axiomatics allows standard methods of proof theory, and allows to draw parallels with logic of explicit proofs LP.

Gentzen style rules of $\mathsf{S4}_n^{\mathsf{J}}$:

$$\frac{\Gamma \Rightarrow \Delta, A}{\neg A, \Gamma \Rightarrow \Delta}(\neg \Rightarrow) \qquad \frac{A, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg A}(\Rightarrow \neg)$$

$$\frac{A, B, \Gamma \Rightarrow \Delta}{A \wedge B, \Gamma \Rightarrow \Delta}(\wedge \Rightarrow) \qquad \frac{\Gamma \Rightarrow \Delta, A \quad \Gamma \Rightarrow \Delta, B}{\Gamma \Rightarrow \Delta, A \wedge B}(\Rightarrow \wedge)$$

$$\frac{A, \Gamma \Rightarrow \Delta \quad B, \Gamma \Rightarrow \Delta}{A \vee B, \Gamma \Rightarrow \Delta}(\vee \Rightarrow) \qquad \frac{\Gamma \Rightarrow \Delta, A, B}{\Gamma \Rightarrow \Delta, A \vee B}(\Rightarrow \vee)$$

$$\frac{\Gamma \Rightarrow \Delta, A \quad B, \Gamma \Rightarrow A}{A \to B, \Gamma \Rightarrow \Delta}(\to \Rightarrow) \qquad \frac{A, \Gamma \Rightarrow \Delta, B}{\Gamma \Rightarrow \Delta, A \to B}(\Rightarrow \to)$$

and $n+1$ pairs of modal rules:

$$\frac{A, \Box A, \Gamma \Rightarrow \Delta}{\Box A, \Gamma \Rightarrow \Delta}(\Box \Rightarrow) \qquad \frac{J\Gamma, \Box\Delta \Rightarrow A}{J\Gamma, \Box\Delta, \Pi \Rightarrow \Sigma, \Box A}(\Rightarrow \Box)$$

where $\Box \in \{K_1, \ldots, K_n, J\}$ and $\Box\{A_1, \ldots, A_m\} = \{\Box A_1, \ldots, \Box A_m\}$.

1.3 Main Definitions and Facts for the Realization Procedure

The goal of this work was to replace justifiable knowledge J with explicit justifications, i.e. proof polynomials (proof terms). In other words, instead of knowing that something is 'justified,' we would like to have its actual justification. An algorithm that accomplishes this is called *realization procedure*.

Definition 1.3 To *realize* a modal formula F from $\mathsf{S4}_n^{\mathsf{J}}$ in $\mathsf{S4}_n\mathsf{LP}$ means to substitute proof polynomials for all occurrences of J in F.

Definition 1.4 A realization r is called *normal* if all negative occurrences of J are realized by proof variables.

Realization theorem.

The first version of the realization theorem for $\mathsf{S4}$ and LP, producing a Hilbert-style derivation, was established in [2]. Another version of the realization theorem, producing a Gentzen-style derivation, was presented in [3]. In both versions the produced evidence terms could be exponential in the length of a given cut-free proof of the theorem. A modified algorithm in [5] lowers the bound of produced evidence terms' length to quadratic in the size of a given cut-free proof of an epistemic modal theorem.

In this work, we extend realization algorithm from [5] to $\mathsf{S4}_n^{\mathsf{J}}$ and $\mathsf{S4}_n\mathsf{LP}$, implement it, and test its performance on a number of paradigmatic epistemic problems. These tests show robust behavior of realization terms in which complexity stays firmly within theoretically predicted polynomial (quadratic) bounds.

1.4 Realization Algorithm

The realization procedure works by induction on the depth of the $\mathsf{S4}^{\mathsf{J}}_n$ derivation tree. It runs through the Gentzen-style proof of a formula F in $\mathsf{S4}^{\mathsf{J}}_n$ and simultaneously constructs a realization and Hilbert-style proof of the realized formula. We also keep track of all instances of the evidence for axiom rule R3 used in this Hilbert-style proof, i.e., of constant specification.

We start with definitions of positive and negative occurrences of modality J in a formula and in a sequent, as adapted from [2]:

- An outer occurrence of J in $\mathsf{J}F$ is positive;
- A corresponding occurrence of J in F and $G \to F$, $G \vee F$, $G \wedge F$, $\mathsf{J}F$, and $\Gamma \Rightarrow \Delta, F$ has the same polarity;
- Corresponding occurrences of J in F and $\neg F$, $F \to G$, and $F, \Gamma \Rightarrow \Delta$ have opposite polarities.

In a cut-free derivation, the rules respect polarities. Occurrences of J introduced by $(\Rightarrow \mathsf{J})$ are positive:

$$\frac{J\Gamma \Rightarrow A}{J\Gamma, \Pi \Rightarrow \Sigma, \mathbf{J}\mathbf{A}} (\Rightarrow \mathsf{J}).$$

All occurrences of J-modality in a given derivation tree of $\Rightarrow F$ are divided into families of related occurrences. Each occurrence of J in a side formula G (i.e., from Γ and Δ) in the premise of the rule is related only to the corresponding occurrence of J in G in the conclusion of the rule. Similarly, each occurrence of J in an active formula of the rule, i.e., in a formula in the premise that is transformed by the rule, is related only to the corresponding occurrence of J in the principal formula of the rule, i.e. in the result of transformation. For example, in the $(\mathsf{J} \Rightarrow)$-rule, formulas A and $\mathsf{J}A$ in the premise sequent are the active formulas, and formula $\mathsf{J}A$ in the conclusion sequent is the principal formula. This relationship is extended by reflexivity and transitivity. Therefore all related occurrences are naturally split into *families of related occurrences.*

Since rules in the cut-free Gentzen system respect polarities, each family consists of J's of the same polarity. We call a family *positive* if it consists of positive J's, and *negative* if it consists of negative J's.

J modalities from the same family correspond to the same occurrence of J in the proof, so we realize them by the same proof polynomial that explicates this J. In addition, due to the normality condition, all J's from a negative family have to be realized by the same proof variable.

Proofs (derivations) of formulas in a Gentzen system, $\mathsf{S4}^{\mathsf{J}}_n\mathsf{G}$ in particular, can be viewed as derivation trees. Nodes are triples: the current sequent, name of the rule and the principal formulas, and axioms as leaves. It imposes a tree structure on each family of J's, with leaves as those nodes where J's of a particular family are first introduced (at a leaf of the derivation tree or in a $(\Rightarrow \Box)$ rule).

Comment: It is not necessary to carry sequents in every node - they can be

reconstructed from rule names, principle formulas, and full sequents at the root. Although for the realization algorithm full sequents in every node are needed.

A positive family of J's is *essential* if at least one of its leaves corresponds to a principal J in a $(\Rightarrow \Box)$ rule, and is *non-essential* otherwise.

Realization algorithm:
(by recursion on the derivation tree structure)
In our system $\mathsf{S4}_n^{\mathsf{J}}\mathsf{G}$, there are only three ways of introducing new J-modalities:

- by an axiom;
- inside a formula by which a sequent is 'weakened' in a $(\Rightarrow \Box)$ rule;
- the outer J in the principal formula of a $(\Rightarrow \Box)$ rule.

Let us enumerate all $(\Rightarrow \mathsf{J})$ rules in the derivation tree and associate provisional variable u_i with the principal J of the i-th rule. All of the provisional variables will be replaced with proof polynomials by the end of the algorithm.

Stage 1 *Every negative family and non-essential family of* J*'s is realized by a fresh proof variable. All* J*'s from such a family will be realized by a proof variable corresponding to that family.*

Stage 2 *Pick an essential positive family of* J*'s. Enumerate all the occurrences of* $(\Rightarrow \Box)$ *rules that introduce* J*'s from this family as the principles:* $i_1 < i_2 < \ldots < i_k$*. All such* J*'s are initially realized by provisional term* $u_{i_1} + u_{i_2} + \ldots u_{i_k}$*, where addition is associated to the left and* u_i*'s are fresh provisional variables.*

We also initialize a substitution σ*, which acts on these provisional variables, to be the empty substitution. At the end of the realization procedure, this substitution will assign a certain proof polynomial to each provisional variable. As a result, essential positive* J*'s will also be realized by proof polynomials that contain no provisional variables.*

Next, each $\mathsf{S4}_n^{\mathsf{J}}$ *formula* G *occurring in the sequent derivation is translated into an* $\mathsf{S4}_n\mathsf{LP}$ *formula* G_r *as follows: each occurrence of* J *in* G *is replaced by a proof polynomial* $t\sigma$ *that possibly contains provisional variables, where* t *is the term realizing the family of that* J*, and* σ *is the current state of the substitution acting on provisional variables. This substitution is appended during the realization procedure, namely, during processing of* $(\Rightarrow \Box)$ *rules.*

Stage 3 *For each sequent in the initial derivation we will construct*

- *an* $\mathsf{S4}_n\mathsf{LP}$ *formula* C *that corresponds to that sequent,*
- *a proof polynomial* t *that contains no provisional variables, and*
- *a Hilbert-style derivation of* $t : C$

recursively on the structure of the derivation tree of $\Rightarrow F$*.*

Kuznets and Brezhnev had the idea to use the polynomials t. They are used while processing the $(\Rightarrow \Box)$ rules of the initial $\mathsf{S4}_n^{\mathsf{J}}$ derivation, and are a vital part of eliminating exponential blow-up.

Base: Let C be a sequent formula. Any sequent formula $\Gamma \Rightarrow \Delta$, where $\Gamma = \{A_1, \ldots, A_n\}$, and $\Delta = \{B_1, \ldots, B_m\}$, is translated into a formula $(\ldots(A_1^r \wedge A_2^r) \wedge \ldots) \wedge A_n^r \rightarrow (\ldots(B_1^r \vee B_2^r) \vee \ldots) \vee B_m^r$.

The antecedent and the consequent of a sequent are multisets, so the order of formulas is irrelevant in both, but normal Hilbert-style operations do not possess such freedom. Thus we need to force some order on A_i^r's and on B_i^r's, so we can use any ordering that allows efficient sorting. This ordering should be uniform for all sequents. This is important for Cook and Reckhow's idea [9] of implementing each step of Gentzen-style derivation by several steps of the corresponding Hilbert-style derivation, otherwise the formulas on different branches of the tree might not match. The lexicographical order is a natural one.

An empty consequent constitutes empty disjunction and is translated as $\bot$. An empty antecedent constitutes empty conjunction and is translated as $\top$. Therefore $\Rightarrow F$ is translated as $\top \rightarrow F^r$.

Translation of two axioms of $\mathsf{S4}_n^{\mathsf{J}}\mathsf{G}$:

(A) $A_1, \ldots, A_{i-1}, S, A_i, \ldots, A_n \Rightarrow B_1, \ldots, B_{j-1}, S, B_j, \ldots, B_m$
is translated as
$A_1^r \wedge \ldots \wedge A_{i-1}^r \wedge S \wedge A_i^r \wedge \ldots \wedge A_n^r \rightarrow B_1^r \vee \ldots \vee B_{j-1}^r \vee S \vee B_j^r \vee \ldots \vee B_m^r$,
in particular, $S \Rightarrow S$ is translated as $S^r \rightarrow S^r$;

(B) $\bot, A_1, \ldots, A_n \Rightarrow B_1, \ldots, B_m$
is translated as
$\bot \wedge A_1^r \wedge \ldots \wedge A_n^r \rightarrow B_1^r \vee \ldots \vee B_m^r$;
in particular, $\bot \Rightarrow$ is translated as $\bot \rightarrow \bot$.

(Assuming that A_k's and B_l's, A_k^r's and B_l^r's are already ordered alphabetically, and $\bot$ is the first symbol of the alphabet, disjunctions and conjunctions are associated to the left.)

Each translated implication C of this type is clearly derivable in $\mathsf{S4}_n\mathsf{LP}$. After application of the Lifting Lemma to this derivation, we get a ground proof polynomial s and a derivation of $s : C$.

Induction Step:

Propositional rule with one premise:

$$\frac{\Gamma \Rightarrow \Delta}{\Gamma' \Rightarrow \Delta'} \ .$$

Let C and C' be translations of $\Gamma \Rightarrow \Delta$ and $\Gamma' \Rightarrow \Delta'$ respectively. By induction hypothesis, there is a term t_C and a derivation l_C of $t_C : C$. By propositional reasoning, there is a derivation of $C \rightarrow C'$. Using the Lifting Lemma, we get a ground term t_R and a derivation l_R of $t_R : (C \rightarrow C')$. Concatenating l_C with l_R and appending the result with the following sequence ... (*derivation* l_R)

`n.` $t_R : (C \rightarrow C')$
... (*derivation* l_C)
`m.` $t_C : C$
`m+1.` $t_R : (C \rightarrow C') \rightarrow (t_C : C \rightarrow t_R \cdot t_C : C')$ (axiom $E1$)

m+2. $t_C : C \to t_R \cdot t_C : C'$ (MP from n and m+1)
m+3. $t_R \cdot t_C : C'$ (MP from m and m+2) ,

we obtain the term $t_{C'} = t_R \cdot t_C$ and the derivation $l_{C'}$ of $t_R \cdot t_C : C'$.

A case of a propositional rule with two premises are handled in a similar way.

Let us consider $(\Box \Rightarrow)$ rule for $\Gamma = \{B_1, \ldots, B_n\}$, and $\Delta = \{D_1, \ldots, D_m\}$:

$$\frac{A, \mathsf{J}A, B_1, \ldots, B_n \Rightarrow D_1, \ldots, D_m}{\mathsf{J}A, B_1, \ldots, B_n \Rightarrow D_1, \ldots, D_m}(\Box \Rightarrow) \ .$$

Without loss of generality, let's assume that the translation of the premise is

$$C = B_1^r \wedge \ldots \wedge B_{i-1}^r \wedge A^r \wedge B_i^r \wedge \ldots \wedge B_{j-1}^r \wedge x : A^r \wedge B_j^r \wedge \ldots \wedge B_n^r \to D,$$

where $D = D_1^r \vee \ldots \vee D_m^r$ and x is the proof variable associated with the negative family of the outer J-modality in $\mathsf{J}A$. Then the translation of the conclusion is

$$C' = B_1^r \wedge \ldots \wedge B_{j-1}^r \wedge x : A^r \wedge B_j^r \wedge \ldots \wedge B_n^r \to D.$$

Since $\mathsf{S4}_n\mathsf{LP} \vdash x : A^r \to A^r$ (reflexivity principle E4), it is easy to derive $C \to C'$. Then, using the Lifting Lemma, we obtain a ground term $t_{(\Box\Rightarrow)}$ and a derivation $l_{(\Box\Rightarrow)}$ of $t_{(\Box\Rightarrow)} : (C \to C')$. The rest is the same as with the one-premise propositional rules.

The only rule that is treated differently is $(\Rightarrow \Box)$, which includes two cases: $(\Rightarrow K_i)$ and $(\Rightarrow \mathsf{J})$. Let's consider the first one: for $\Gamma = \{B_1, \ldots, B_n\}$, $\Delta = \{D_1, \ldots, D_m\}$, $\Sigma = \{E_1, \ldots, E_o\}$, and $\Pi = \{A_1, \ldots, A_r\}$

$$\frac{J\Gamma, K_i\Delta \Rightarrow A}{J\Gamma, K_i\Delta, \Pi \Rightarrow \Sigma, K_iA}(\Rightarrow K_i) \ .$$

Without loss of generality, let's assume that the translation of the premise is

$$C = \boldsymbol{x} : \Gamma^r \wedge K_i\Delta^r \to A^r$$

and $\boldsymbol{x} = (x_1, \ldots, x_n)$, where x_i are distinct proof variables associated with the negative families of the outer J-modality in $\mathsf{J}\Gamma$.

Then the translation of the conclusion is

$$C' = \boldsymbol{x} : \Gamma^r \wedge K_i\Delta^r \wedge \Pi^r \to \Sigma^r \vee K_iA^r \ .$$

By induction hypothesis, we have a term t_C and a derivation l_C of

$$t_C : (\boldsymbol{x} : \Gamma^r \wedge K_i : \boldsymbol{D^r} \to A^r)$$

... (*derivation* l_C)

n. $t_C : (\boldsymbol{x} : \Gamma^r \wedge K_i\Delta^r \to A^r)$

n+1. $t_C : (\boldsymbol{x} : \Gamma^r \wedge K_i\Delta^r \to A^r) \to K_i(\boldsymbol{x} : \Gamma^r \wedge K_i\Delta^r \to A^r)$(axiom $C1$)

n+2. $K_i(\boldsymbol{x} : \Gamma^r \wedge K_i\Delta^r \to A^r)$ (MP from n and n+1)

n+3. $K_i(\boldsymbol{x} : \Gamma^r \wedge K_i\Delta^r \to A^r) \to (K_i(\boldsymbol{x} : \Gamma^r \wedge K_i\Delta^r) \to K_iA^r)$ (axiom $B1_i$)

n+4. $K_i(\boldsymbol{x} : \Gamma^r \wedge K_i\Delta^r) \to K_iA^r$ (MP from n+2 and n+3)

n+5. $\left(\bigwedge K_ix_j : B_j^r\right) \wedge (\bigwedge K_iK_iD_l^r) \to K_i(\boldsymbol{x} : \Gamma^r \wedge K_i\Delta^r)$(simple $\mathsf{S4}_n\mathsf{LP}$ reasoning)

n+6. $\left(\bigwedge K_ix_j : B_j^r\right) \wedge (\bigwedge K_iK_iD_l^r) \to K_iA^r$ (*syllogism* from n+4 and n+5)

n+7. $x_j : B_j^r \to K_ix_j : B_j^r$ (an easy $\mathsf{S4}_n\mathsf{LP}$ fact)

n+8. $K_iD_l^r \to K_iK_iD_l^r$ (axiom $B3_i$)

n+9. $\left(\bigwedge x_j : B_j^r\right) \wedge (\bigwedge K_iD_l^r) \to \left(\bigwedge K_ix_j : B_j^r\right) \wedge (\bigwedge K_iK_iD_l^r)$

(propositional reasoning from n+7 and n+8)

n+10. $\left(\bigwedge x_j : B_j^r\right) \wedge (\bigwedge K_iD_l^r) \to K_iA^r$ (*syllogism* from n+6 and n+9)

n+11. $\left(\bigwedge x_j : B_j^r\right) \wedge (\bigwedge K_iD_l^r) \wedge \Pi \to \Sigma \vee K_iA^r$

(propositional reasoning from n+10)

n+12. $t_{n+11} : (\left(\bigwedge x_j : B_j^r\right) \wedge (\bigwedge K_iD_l^r) \wedge \Pi \to \Sigma \vee K_iA^r)$

(Lifting Lemma from n+11).

We obtained the ground term t_{n+11} and a derivation of $t_{n+11} : (C \to C')$. Now, let us consider the $(\Rightarrow \mathsf{J})$ rule:

for $\Delta = \{D_1, \ldots, D_m\}$, $\Sigma = \{E_1, \ldots, E_o\}$, and $\Pi = \{A_1, \ldots, A_r\}$

$$\frac{\mathsf{J}D_1, \ldots, \mathsf{J}D_m \Rightarrow A}{\mathsf{J}D_1, \ldots, \mathsf{J}D_m, A_1, \ldots, A_r \Rightarrow E_1, \ldots, E_o, JA}(\Rightarrow \mathsf{J}) \ .$$

All J's in $\mathsf{J}D_i$'s are negative and belong to different families, so they are realized by distinct proof variables x_i's. Let k be the number of this $(\Rightarrow \mathsf{J})$ rule and let its family be realized by $u_{s_1} + \ldots + u_k + \ldots u_{s_l}$. By induction hypothesis, we have a term t_C and a derivation l_C of

$$t_C : (x_1 : D_1^r \wedge \ldots \wedge x_m : D_m^r \to A^r).$$

By Lemma 2, we construct a term $s = s(x_1, \ldots, x_m)$ and a derivation l_1 of

$$x_1 : D_1^r \wedge \ldots \wedge x_m : D_m^r \to s : (x_1 : D_1^r \wedge \ldots \wedge x_m : D_m^r).$$

Note that s does not contain any provisional variables. It is now easy to append derivations l_C and l_1 (we'll use vector notation for conjunction):

... (*derivation* l_C)
n. $t_C : (\boldsymbol{x} : \boldsymbol{D^r} \to A^r)$
... (*derivation* l_1)
m. $\boldsymbol{x} : \boldsymbol{D^r} \to s : (\boldsymbol{x} : \boldsymbol{D^r})$
m+1. $t_C : (\boldsymbol{x} : \boldsymbol{D^r} \to A^r) \to (s : (\boldsymbol{x} : \boldsymbol{D^r}) \to t_C \cdot s : A^r)$ (axiom $E1$)
m+2. $s : (\boldsymbol{x} : \boldsymbol{D^r}) \to t_C \cdot s : A^r$ (MP from n and m+1)
m+3. $\boldsymbol{x} : \boldsymbol{D^r} \to t_C \cdot s : A^r$ (*syllogism* from m and m+2)
...(`using axiom` $E3$ several times)
k. $\boldsymbol{x} : \boldsymbol{D^r} \to (u_{s_1}\sigma + \ldots + t_C \cdot s + \ldots u_{s_l}\sigma) : A^r$.

Moreover this derivation is easy to append to obtain derivation l_2 of a formula C' that (modulo permutations) looks like

$$\boldsymbol{x} : \boldsymbol{D^r} \wedge \boldsymbol{A} \to \boldsymbol{E} \vee (u_{s_1}\sigma + \ldots + t_C \cdot s + \ldots u_{s_l}\sigma) : A^r$$

(here $\boldsymbol{x} : \boldsymbol{D^r}$ and $\boldsymbol{A}$ stand for conjunctions, and $\boldsymbol{E}$ for disjunctions).

We then use the Lifting Lemma to reproduce a ground term $t_{C'}$ and a derivation $l_{C'}$ of

$$t_{C'} : (\boldsymbol{x} : \boldsymbol{D^r} \wedge \boldsymbol{A} \to \boldsymbol{E} \vee (u_{s_1}\sigma + \ldots + t_C \cdot s + \ldots u_{s_l}\sigma) : A^r) \ .$$

While lifting l_2 [5], there is no need to lift its initial part l_C since the only formula we use for the second part is $t_C : (\boldsymbol{x} : \boldsymbol{D^r} \to A^r)$; this formula is easily lifted by adding to l_C the following two formulas:

$t_C : (\boldsymbol{x} : \boldsymbol{D^r} \to A^r) \to !t_C : t_C : (\boldsymbol{x} : \boldsymbol{D^r} \to A^r)$, and

$!t_C : t_C : (\boldsymbol{x} : \boldsymbol{D^r} \to A^r)$,

the latter being the desired lifted version. This procedure produces a ground term because t_C is ground. Also, this modification renders the whole procedure polynomial in the size of the original $\mathsf{S4}^{\mathsf{J}}_n$-derivation. In the original algorithm [2], each time a $(\Rightarrow \Box)$ rule is processed, most formulas in the initial derivation are replaced by three formulas in the lifted one, which leads to exponential growth in the number of $(\Rightarrow \Box)$ rules. We then append σ by a new substitution: $\sigma = \sigma + \{u_k \leftarrow t_C \cdot s\}$, and apply this substitution throughout the derivation ($\mathsf{S4}_n\mathsf{LP}$ is known to be closed under substitutions). After that, there are no occurrences of u_k remaining in the derivation. As a result, we eliminated one provisional variable.

Final Touch: At the end of the procedure, the entire derivation tree of $\Rightarrow F$ is translated – all $(\Rightarrow \Box)$ rules have been processed and there are no provisional variables remaining. Thus, F^r is simply an $\mathsf{S4}_n\mathsf{LP}$ formula. Moreover, we have a Hilbert-style derivation l_t of $t : (\top \to F^r)$ for some ground term t. To acquire F^r, we do the following:

... (derivation l_t)
n. $t : (\top \to F^r)$
n+1. $t : (\top \to F^r) \to (\top \to F^r)$ (axiom $E4$)
n+2. $(\top \to F^r)$ (MP from n and n+1)

n+3. $\top$

n+4. F^r (MP from n+2 and n+3).

1.5 Notes on implementation

The procedure was implemented in OCaml. Evaluation of box families, considered to be trivial on paper, was the biggest hurdle. Each box was assigned a unique family identifier. Proofs have branches, and families grow by transitive extension, thus the disjoint set of sets (families) of box identifiers that are related (belong to the same family) needs to be maintained. What actually occurs is that each box is replaced with $\Pr(Provisional, F)$ where *Provisional* is this unique identifier. The algorithm recursively walks over the proof tree, assigns these identifiers, and collects information about which identifiers fall into which family. At each step representing application of a rule, we have to track how each formula above the line (of the rule) was transformed and then transform the formulas below the line accordingly.

For all classical reasoning that justifies a shift from rule assumptions to rule conclusion, we say that there is a (fresh) proof constant justifying the implication, and deduce such implications using the reflexivity axiom $t : A \to A$.

At some point, we realized that the produced proofs were way too long, both in the number of steps and the length of produced formulas. We partially address both.

First of all, as suggested by Melvin Fitting, we introduced two one-step rules: Deduction and Lifting. After each application of the Lifting lemma or Deduction Theorem, we retain the full chain of reasoning for validation purposes, but collapse them for purposes of display. Therefore, one sees the following:

$$k. \quad A_1, \ldots, A_m, \ldots, A_n \vdash B$$

$$k+1. \quad A_{m+1}, \ldots, A_n \vdash A_m \to \ldots \to A_1 \to B \text{ (by Deduction)}$$

and

$$k. \quad x_1 : A_1, \ldots, x_n : A_n \vdash B$$

$$k+1. \; x_1 : A_1, \ldots, x_n : A_n \vdash c(x_1, \ldots, x_n) : B \text{ (by Lifting).}$$

Second, we assigned fresh (shortcut) constants to each proof term appearing in the proof and with length more than 5. We list these assignments at the end of the proof and provide a hyperlink from each occurrence of such a constant to its definition.

We also inserted dummy steps to mark the boundaries between individual Gentzen proof steps and certain stages of $\Rightarrow \Box$ rule realization. Such dummy steps are labeled with Gentzen rules or realization stage names instead of Hilbert rules or axioms. These dummy rules are also rendered with normal font size, and all intermediate steps with a smaller font.

This implementation was connected with the automatic prover for multi-agent logic with justified knowledge $\mathsf{S4}^{\mathsf{J}}_n$ in the MetaPRL logical framework [6; 7]. The

output of the $\mathsf{S4}^{\mathsf{J}}_n$ prover is a Gentzen-style cut-free proof, exactly what polynomial realization procedures need.

2 Experiments

The realization procedure was ran on several simple examples, as well as on several classical puzzles: the Wise Girls puzzle and the Wise Men Puzzle [16], which we will not cover in this article.

We present here the Muddy Children Puzzle and Surprise Examination paradox.

2.1 Graphs

Some of the experiments are accompanied with the three types of graphs.

The first graph shows the growth of the number of steps in a Hilbert-style proof as a function of the number of steps in the original Gentzen-style proof. According to [5] we should observe $O(n^2)$ - dependency on this graph.

The second graph shows the growth of the total length of all formulas in a Hilbert-style proof as a function of the total length of all formulas in the original Gentzen-style proof. According to [5] we should observe $O(n^6)$ - dependency on this graph.

And the last graph shows the growth of the external (outer) terms' sizes in a Hilbert-style proof as a function of the number of steps in the original Gentzen-style proof. According to [5] we should observe $O(n^2)$ - dependency on this graph.

One can see the bumps on the graphs. The bumps are the result of $(\Rightarrow \Box)$ rule realization. We observe that all other rules are linear in the number of steps. Therefore, it is possible that $O(n^6)$ bound can be improved by counting "expensive" and "inexpensive" rules separately.

2.2 Realization of the Muddy Children Puzzle

N children are playing together [10]. Their mother tells them that if they get dirty, there will be severe consequences. Now it so happens that during their play, some of the children, say k of them, get mud on their foreheads. Each can see mud on the others, but not on his own forehead, so no one says anything. Along comes their father who says, "At least one of you has mud on your forehead," thus expressing a fact known to each of them before he speaks (if $k > 1$). The father then asks the following question repeatedly: "Do any of you know whether or not you have mud on your own forehead?" Assuming that the children are all perceptive, intelligent, truthful, and that they answer simultaneously, what will happen? There is a 'proof' that the first $k - 1$ times he asks the question, they will all answer "No," but the k^{th} time, the children with muddy foreheads will all answer "Yes."

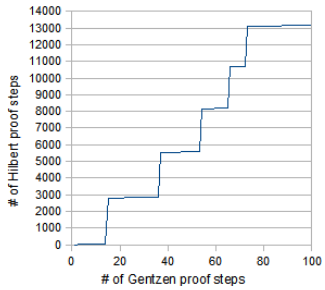
For this puzzle, a straightforward epistemic logic formalization can lead to inconsistent sets of hypotheses [7; 11; 12]. For example, let $c_1, \ldots, c_n$ be propositional variables reserved for N children, where c_i encodes that i^{th} child is muddy. J stand

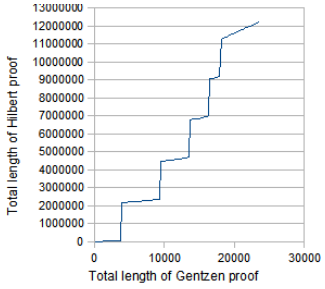
for "it is a common knowledge", and kw i a stand for "i knows whether a or $\neg a$". Then, the set of formulae

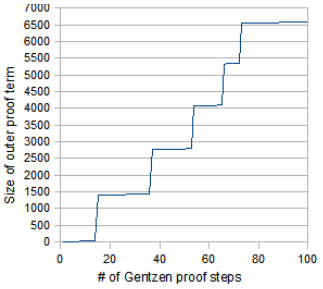
$$J(c_1 \vee c_2 \vee \ldots \vee c_n), \qquad J(\text{kw } i\ C_j) \text{ for all } i \neq j, \qquad J\neg(\text{kw } i\ C_i) \text{ for all } i,$$

is inconsistent for all $n \geq 2$.

For the sake of observing the complexity of the realization process, one can find the complexities of this contradictory formalization for five children in the following graphs.







In [14] a version of logic similar to $\mathsf{S4}^{\mathsf{J}}_n$ was used, with all modalities graded by time, to present a model-based solution of Muddy Children. This solution has a model, hence it avoids introduction of a contradiction. For reference, it is presented it the Appendix of [16].

Here we present three of the possible formalizations of the Muddy Children puzzle for three children, using McCarthy's idea. Graphs for each complexity measure are given after the last formalization, we merged them for ease of comparison.

We use the following notation:

- K-modality has two indices now: time and agent, i.e. $K_{t,a}A$ stands for "At time t, agent a knows A."
- $E_tA \ ::= \ K_{t,a_1}A \wedge \ldots \wedge K_{t,a_n}A$ stands for "Everybody knows A at time t."
- $\mathtt{kw}_{t,a}\ A ::= \ K_{t,a}A \vee K_{t,a}\neg A$, i.e. "At the moment t agent a knows whether or not A holds."

We use McCarthy's idea only, not his system. We stay in $\mathsf{S4}^{\mathsf{J}}_n$, no new axioms or rules related to time are introduced, and $K_{t,a}$ is just syntactic sugar for $K_{(t-1)n+a}$, where n is the number of agents.

Without loss of generality, let us assume that the first and third children are dirty, and the second one is clean, i.e., $c_1, \neg c_2, c_3$.

(A) Longer version

Denotations: Child 1 stands for the first child, etc.

Hypotheses: (denotation: *wmn* stands for 'were/was muddy or not')

(1) $c_1 \wedge \neg c_2 \wedge c_3$

(2) $\mathtt{kw}_{3,2}\ c_1$; *at moment 3, Child 2 knows if Child 1 is muddy or not*

(3) $\mathtt{kw}_{3,2}\ c_3$; *at moment 3, Child 2 knows if Child 3 is muddy or not*

(4) $\mathtt{kw}_{3,2}\ (\mathtt{kw}_{2,1}\ c_1)$; *at moment 3, Child 2 knows if Child 1 at moment 2 knew he wmn*

(5) $K_{3,2}($ *at moment 3, the second child knows that*

(6) $(\mathtt{kw}_{2,1}\ c_3) \wedge (\mathtt{kw}_{2,1}\ c_2) \wedge$ *at moment 2, Child 1 knew if Child 2 and Child 3 wmn, and*

(7) $K_{2,1}($ *at moment 2, Child 1 knows that*

(8) $\neg(\mathtt{kw}_{1,1}\ c_1) \wedge \neg(\mathtt{kw}_{1,2}\ c_2) \wedge \neg(\mathtt{kw}_{1,3}\ c_3) \wedge$ *at the 1st moment, nobody knew if they wmn, and*

(9) $(E_1(c_1 \vee c_2 \vee c_3)) \wedge$ *at 1st moment, everybody knew that at least one of them was muddy, and*

(10) $(\mathtt{kw}_{1,1}\ c_2) \wedge (\mathtt{kw}_{1,1}\ c_3) \wedge$ *at 1st moment, Child 1 knew if Child 2 and Child 3 wmn, and*

(11) $(\mathtt{kw}_{1,2}\ c_1) \wedge (\mathtt{kw}_{1,2}\ c_3) \wedge$ *at 1st moment, Child 2 knew if Child 1 and Child 3 wmn, and*

(12) $(\mathtt{kw}_{1,3}\ c_1) \wedge (\mathtt{kw}_{1,3}\ c_2)\ ))$ *at 1st moment, Child 3 knew if Child 1 and Child 2 wmn*

(13) $K_{3,2}((c_1 \wedge c_2 \wedge c_3) \rightarrow \neg\ \mathtt{kw}_{2,1}\ \ c_1)$ *at moment 3, Child 2 knows that if all of them are dirty, Child 1 at moment 2 did not know if he was muddy or not*

Conclusion: $\mathtt{kw}_{3,2}\ c_2$ *at moment 3, Child 2 knew if he was muddy or not*

The last hypothesis cannot be relaxed because in situations where a child says that (s)he does not know if (s)he is muddy, (s)he cannot really derive it in $\mathsf{S4}^{\mathsf{J}}_n$. So when the system performs case analysis, we have to help it with such implications.

(B) A short version:

If we make the second, third, and fourth premises stronger, we then obtain a shorter version:

Hypotheses:

(1) $c_1 \wedge \neg c_2 \wedge c_3$
(2) $K_{3,2}\ c_1$
(3) $K_{3,2}\ c_3$
(4) $K_{3,2}\ (K_{2,1} c_1)$
(5) $K_{3,2}\ ((c_1 \wedge c_2 \wedge c_3) \rightarrow \neg \mathtt{kw}_{2,1}\ c_1)$

Conclusion: $\mathtt{kw}_{3,2}\ c_2$.

(C) A short version with J:

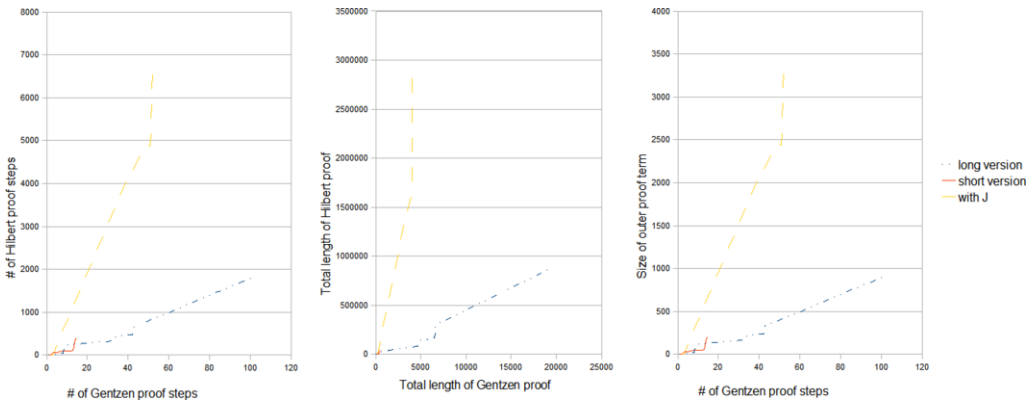
We would like to see some modalities realized as proof terms so we replaced all modalities that represent facts that everyone knows (due to public announcements or general conditions of the puzzle) with J. We also strengthen the conclusion to state that at the third moment, everyone knows that everyone knows about themselves, so we have to add lines (6), (7) to bring the knowledge of the first and third children from step 2 to step 3, and we add (9) to reflect the fact that if the second child learns about himself, he will announce it. We did not perform this transformation with the longer version because the prover was overwhelmed by the complexity.

Hypotheses:

(1) $c_1 \wedge \neg c_2 \wedge c_3$
(2) $J(\mathtt{kw}_{3,2}\ c_1)$
(3) $J(\mathtt{kw}_{3,2}\ c_3)$
(4) $J(K_{2,1}\ c_1)$
(5) $J(K_{2,3}\ c_3)$

(6) $J(K_{2,1}c_1 \to K_{3,1}c_1)$
(7) $J(K_{2,3}c_3 \to K_{3,3}c_3)$
(8) $J((c_1 \wedge c_2 \wedge c_3) \to \neg(\mathtt{kw}_{2,1}\ c_1))$
(9) $J(K_{3,2}c_2 \to J(K_{3,2}\ c_2))$
Conclusion: $J(K_{3,2}\neg c_2 \wedge K_{3,1}c_1 \wedge K_{3,3}c_3)$.

Here are the graphs for all three cases:



2.3 Formalization of the Surprise Examination Paradox

There is a famous epistemic Surprise Examination Paradox (SEP): A professor tells students in his class that there will be a surprise in-class exam during the next week. There are 5 weekdays and class meets every day. Can the professor give an exam? We assume that professor and students are truthful and smart. By the usual backward induction argument, students figure that the exam cannot be given at all. Indeed, the exam cannot be given on the last day, since it would not be a surprise. Therefore, the exam cannot be given the day before, etc. The paradox occurs when the professor gives an exam on day two, and it is a complete surprise for the students!

The are 12 pages of references on the subject in [8], and it was not our goal to become experts in this area. Here is a naive, straightforward formalization of SEP conditions: consider a very trustful student who, despite his or her intelligence, is very confused on the last day if the test has not yet happened. He/she "knows" that the professor never lies or makes mistakes, and at the same time it seems that there is no alternative option and the exam has to happen on this last day. Any certain answer, positive or negative, to the question "Do you think that the test will be given today?" will lead to a contradiction. Therefore, the natural answer is "I do not know," which implies (just as a negative answer) that the student considers the possibility that there will be no test. This implicit additional outcome resolves the paradox.

In the formalization below we take a conservative approach and simply prove that full set of assumptions is contradictory or that the exam cannot happen on the last day of the week. We use different agent indices to represent different times

and different students, if needed. Knowledge relation between different moments of time for the same student have to be explicitly stated for individual formulae, as $\mathsf{S4}_n^{\mathsf{J}}$ has no built-in support for time.

d_i stands for "the exam took place on ith day." For the case of one student, $\mathtt{K}_i A$ means that the student will know A just before day i. With this setup, $\mathtt{K}_1$ has a somewhat special meaning because it describes what we know up front about this puzzle.

2.3.1 2-day week

$$\frac{1.\ \mathsf{J}\,((d_1 \wedge \neg d_2) \vee (\neg d_1 \wedge d_2)) \quad 2.\ \mathtt{K}_1(\mathtt{K}_2 d_1 \vee \mathtt{K}_2 \neg d_1) \quad 3.\ \mathtt{K}_1 \neg \mathtt{K}_2 d_2 \quad 4.\ \neg \mathtt{K}_1 d_1}{\bot}$$

The first assumption states that the exam will happen on either of the days. The second assumption states that after the first day, it will be known if the exam took place already. The third assumption is effectively our formalization of "surprise" - prior to the second day, we will not know if the exam will happen on second day. And the last assumption is also "surprise", this time, for the first day. MetaPRL finds a proof for this theorem in a few seconds.

2.3.2 2-day week, many students

The same problem with 2 students is formulated trivially by duplicating relevant formulas and replacing modality indices:

$$\frac{\begin{array}{ll} 1.\ \mathsf{J}\,((d_1 \wedge \neg d_2) \vee (\neg d_1 \wedge d_2)) & 5.\ \mathtt{K}_3(\mathtt{K}_4 d_1 \vee \mathtt{K}_4 \neg d_1) \\ 2.\ \mathtt{K}_1(\mathtt{K}_2 d_1 \vee \mathtt{K}_2 \neg d_1) & 6.\ \mathtt{K}_3 \neg \mathtt{K}_4 d_2 \\ 3.\ \mathtt{K}_1 \neg \mathtt{K}_2 d_2 & 7.\ \neg \mathtt{K}_3 d_1 \\ 4.\ \neg \mathtt{K}_1 d_1 & \end{array}}{\bot}$$

Extending this theorem for 40 students has some, but little effect on proof search timing, still keeping it under one second; it seems to be linear or polynomial of low degree. This is expected, as conclusion of the theorem is independent of the number of students, and the first 4 assumptions are sufficient anyway.

2.3.3 3-day week

The first assumption states that exam will happen on either of the days. The 2nd, 3rd, and 4th assumptions say that we know the results of the previous days. The

5th to 8th assumptions are "surprise" conditions.

1. $\mathsf{J}\left((d_1 \wedge \neg d_2 \wedge \neg d_3) \vee (\neg d_1 \wedge d_2 \wedge \neg d_3) \vee (\neg d_1 \wedge \neg d_2 \wedge d_3)\right)$
2. $\mathsf{J}(\mathtt{K}_2 d_1 \vee \mathtt{K}_2 \neg d_1)$
3. $\mathsf{J}(\mathtt{K}_3 d_1 \vee \mathtt{K}_3 \neg d_1)$
4. $\mathsf{J}(\mathtt{K}_3 d_2 \vee \mathtt{K}_3 \neg d_2)$
5. $\mathtt{K}_1 \neg \mathtt{K}_2 d_2$
6. $\mathtt{K}_1 \neg \mathtt{K}_3 d_3$
7. $\mathtt{K}_2 \neg \mathtt{K}_3 d_3$
8. $\neg \mathtt{K}_1 d_1$

$\neg d_3$

Note that this theorem only states the impossibility of the exam on the last day of the week and does not involve backward induction. So this is a simplified version, basically the first step in establishing the paradox.

If we unfold all disjunctions, MetaPRL has no problems completing the proof. But it was unable to find the proof in fully automatic mode in an hour. In this sense, this theorem is harder than Muddy Children puzzle for 4 children, which can be solved in 150-250 seconds. The reason why fully automatic mode fails is because all J-boxed assumptions have to be used twice in the reasoning and the prover works by first exhausting all proof matrices with modalities used at most once, then expanding the search space by allowing each modality to have two prefix interpretations; the resulting search space is big and the solution does not appear quickly. The Muddy Children puzzle also needs this search space expansion, but we get luckier there, although it is not clear exactly why.

The Muddy Children puzzle for 4 children can be sped up by an order of magnitude by manually duplicating assumptions that will be used twice in the reasoning and avoiding automatic expansion of all modalities. We tried to apply the same technique to our problem at hand but it produced too big of a search space for the prover, and the search simply did not finish within a reasonable time.

This straightforward formalization of SEP results in contradiction as expected. For more on SEP see [13].

3 Conclusion

Following [5] and [4], a polynomial realization algorithm for $\mathsf{S4}_n\mathsf{LP}$ was implemented in MetaPRL Logical Framework and connected with $\mathsf{S4}_n^{\mathsf{J}}$ prover [7]. This procedure was run on several interesting examples. Realization algorithm performed better than expected, the bottleneck was always on the $\mathsf{S4}_n^{\mathsf{J}}$ prover's side. On the other hand, even small $\mathsf{S4}_n^{\mathsf{J}}$ Gentzen-style proofs result in long $\mathsf{S4}_n\mathsf{LP}$ Hilbert-style proofs which are beyond human comprehension.

Though we abbreviate certain trivial segments of a proof, the result is too long to be readable. Therefore the instructive parts are the length of the resulting proof, the length of the outer term as functions of an incoming proof, and the realized formula itself.

The algorithm has been known for more than a decade, but this work is a first time it was applied to proofs longer than just few steps. We used famous Muddy

Children Puzzle and Surprise Exam Paradox for our experiments.

4 Future Work

Considering the complications that were met, it seems instrumental to modify or extend the current implementation of $\mathsf{S4}_n\mathsf{LP}$ realizer to accept interactively/manually constructed proofs. This will allow to evaluate realization procedure performance on SEP for three and more days.

References

[1] E. Antonakos. Comparing justified and common knowledge. *The Bulletin of Symbolic Logic*, 12, 2006. in: *2005 Summer Meeting of the ASL*.

[2] Sergei Artemov. Operational modal logic. Technical Report MSI 95-29, Cornell University, 1995.

[3] Sergei Artemov. Explicit provability and constructive semantics. *The Bulletin for Symbolic Logic*, 6(1):1–36, 2001.

[4] Sergei Artemov. Evidence-based common knowledge. Technical Report TR-2004018, CUNY Ph.D. Program in Computer Science Technical Reports, November 2004.

[5] Vladimir Brezhnev and Roman Kuznets. Making knowledge explicit: how hard it is. *Theoretical Computer Science*, 357(1):23–34, 2006.

[6] Yegor Bryukhov. Automatic proof search in logic of justified common knowledge. In Holger Schlingloff, editor, *Proceedings of Methods for Modalities Workshop 2005*. Humboldt University, 2005.

[7] Yegor Bryukhov. *Integration of Decision Procedures into High-Order Interactive Provers*. PhD thesis, The Graduate School and University Center, CUNY, 2006.

[8] Timothy Y. Chow. The surprise examination or unexpected hanging paradox. *The American Mathematical Monthly*, 105:41–51, 1998.

[9] Stephen Cook and Robert Reckhow. On the lengths of proofs in the propositional calculus (preliminary version). In *STOC '74: Proceedings of the sixth annual ACM symposium on Theory of computing*, pages 135–148, New York, NY, USA, 1974. ACM.

[10] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning About Knowledge*. Massachusetts Institute of Technology, 1995.

[11] Melvin Fitting. Modal logic notes, January 2006. First Indian Winter School on Logic and Its Relationship with Other Dsciplines, IIT Bombay.

[12] Melvin Fitting. Modal proof theory. In P. Blackburn, J. F. K. van Benthem, and F. Walter, editors, *Handbook of modal logic*, pages 85–138. Elsevier, 2007.

[13] Paul Franceschi. A dichotomic analysis of the surprise examination paradox, 2002. http://www.univ-corse.fr/~franceschi/sep(gb).htm.

[14] John McCarthy, Masahiko Sato, Takeshi Hayashi, and Shigeru Igarashi. On the model theory of knowledge. Technical report, Stanford, CA, USA, 1978.

[15] John-Jules Ch Meyer and Wiebe Van Der Hoek. *Epistemic Logic for AI and Computer Science*. Cambridge University Press, New York, NY, USA, 1995.

[16] Natalia Novak. Computer-aided reasoning about knowledge and justifications. Technical Report TR-2014001, CUNY Ph.D. Program in Computer Science Technical Reports, January 2014.

[17] Renate Schmidt. A list of computational tools useful for modal logics, and related logics, 2009. http://www.cs.man.ac.uk/~schmidt/tools/.

[18] A. S. Troelstra and H. Schwichtenberg. *Basic proof theory (2nd ed.)*. Cambridge University Press, New York, NY, USA, 2000.