2nd International Conference

Logic and Applications 2013

(LAP 2013)

September 16-20, 2013, Dubrovnik, Croatia

- Book of abstracts -

Course directors

- Zvonimir Šikić, University of Zagreb
- Andre Scedrov, University of Pennsylvania
- Silvia Ghilezan, University of Novi Sad
- Zoran Ognjanović, Mathematical Institute SANU, Belgrade

Combinatorial Aspects of Interpretability Logic

Vedran Čačić, University of Zagreb, Croatia

Gödel's theorems were a big breakthrough for matematical logic. With time, mathematicians started to wonder how they can be generalized, and what else, based on some simple facts we knew, could be deduced about provability predicates. Formalizing provability over some base theory T as a unary modal operator $\Box$, led to the theory GL (named after Gödel and Löb) which we know today is the provability logic of many base theories.

Provability is great for judging absolute strength of some formula against a theory. But what about relative strength? For some base theory T and two formulas F and G, is $T+F$ interpretable in $T+G$? That is, can we find a way of reinterpreting symbols of T, preserving provability of whole T, but such that (reinterpreted) formula F becomes a theorem, if we add G as an axiom? Here we don't just divide formulas into black and white, but try to order them in various shades of gray. In fact, various colors would be a better analogy, since the ordering is usually not total.

We can do something quite analogous here. Formalizing interpretability in the above sense as a binary modal operator $\rhd$, we are led to various interpretability logics, most basic of which is probably IL. Unfortunately, IL itself is, unlike GL, just a "lowest common intersection" of those interpretability logics, and different base theories add to IL different principles of interpretability, extending it in diverse ways.

However, we still can consider properties of GL, and ask ourselves if IL has something analogous. One well-known property of GL is that its closed formulas have very regular normal forms: every GL formula without variables is equivalent to a Boolean combination of formulas $\bot$, $\Box\bot$, $\Box\Box\bot$, and so on. That Boolean combination can be further normalized, taking into account that $\Box^n\bot \to \Box^m\bot$ whenever $n \leq m$.

Do IL formulas have something similar? The first hard question is: what are the basic blocks here? In GL, it was easy—repeating $\Box$ before $\bot$ gives a natural single-parameter countable family of "propositional variables", to be connected into Boolean combinations. There is not anything analogous in IL, except that family itself. Namely, it can be easily seen that $\Box$ can be emulated in IL, $\Box A$ being equivalent to $\neg A \rhd \bot$ ($\bot$ is invariant under interpretation, and $T + \neg A$ is inconsistent iff $T \vdash A$). So the same family $\{\Box^n\bot : n \in \mathbb{N}\}$ is available in IL, too.

In [1], we have shown that many IL formulas have GL equivalents, and by that, the same normal forms as GL formulas. Here we count those formulas, and find the exact share they have in the whole closed fragment of IL.

By studying the general forms of such families of IL formulas, we become aware of many interesting combinatorial problems. First, how to define "share" in the first place? Set of all closed IL formulas is infinite, but we use asymptotics based on complexity of formulas. Second, how to count formulas given recursively? Those recurrences often don't have closed form solutions, but asymptotic behaviour can be obtained via generating functions. And third, many of classes we have to count collectively aren't disjoint: exclusion-inclusion formula helps here.

References

[1] V. Čačić, M. Vuković, *A note on normal forms for the closed fragment of system IL*. Mathematical Communications, **17** (2012), 195–204

Reducibility method: an overview[1]

Silvia Ghilezan, Faculty of Technical Sciences, University of Novi Sad, Serbia

This is an overview of the development and application of the reducibility method in different aspects of logic and computation. The results are obtained over the last twenty years in collaboration with Henk Barendregt, Mariangiola Dezani-Ciancaglini, Daniel Dougherty, Jelena Ivetić, Pierre Lescanne, Silvia Likavec and Viktor Kunčak.

The reducibility method is a well known framework for proving reduction properties of terms typeable in different type systems. It was introduced by Tait in [10] for proving the strong normalization property for the simply typed lambda calculus. The main idea of this method is to relate terms typeable in a certain type system and terms satisfying certain reduction properties such as strong normalisation, head normalisation etc. Emerging from these proofs, the reducibility method became a widely accepted technique for proving various reduction properties of terms typeable in different type systems. This method was used to prove strong normalization of polymorphic (second-order) lambda calculus, intersection type systems, calculus with explicit substitutions and various other type systems. The reader is referred to [1], a comprehensive reference book on type systems.

The basic concept of the method can be represented by a unary predicate $P_\alpha(t)$, which means that a term t typeable by α satisfies the property P. To this aim types are interpreted as suitable sets of terms called saturated or stable sets. Then, the soundness of type assignment is obtained with respect to these interpretations. A consequence of soundness is that every term typeable in the type system belongs to the interpretation of its type. This is an intermediate step between the terms typeable in the given type system and terms satisfying the considered property P. In general, the principal notions of the reducibility method are:

- type interpretations (based on the considered property P);
- term valuations;
- saturation and closure conditions;
- soundness of the type assignment.

In [4] the reducibility method is applied to completely characterise the strongly normalizing lambda terms in the lambda calculus with intersection types. Suitable modifications of the reducibility method lead to uniform proofs of other reduction properties. An overview can be found in [6]. In [2] the reducibility method is applied to characterise normalising, head normalising and weak head normalising terms as well as their persistent versions. In [5] the reducibility is developed for a resource aware term calculus.

In the setting of classical logic, the reducibility method is not well suited to prove strong normalization for $\lambda\mu$-calculus, the simply typed classical term calculus. The symmetric candidates technique used to prove strong normalisation employs a fixed-point technique to define the reducibility candidates in [3].

Extending on the reducibility method, *logical relations* were introduced by Statman in [9] to proof the confluence (the Church-Rosser property) of $\beta\eta$-reduction of the simply typed λ-terms. It became a well-known method for proving confluence and

[1]Partially supported by the Serbian Ministry of Education, Science, and Technological Development (projects ON174026 and III044006).

standardisation in various type systems. Similarly to the reducibility method, the key notions are type interpretations. Logical relations in turn is a method based on binary relations $R_\alpha(t, t')$, which relate terms t and t' typeable by the type α that satisfy the relation R. Types are then interpreted as admissible relations.

In programming languages it is often necessary to relate terms either from the same language or from different languages in order to show their equivalence. To this aim logical relations became a powerful tool in programming languages, see Pierce [8]. Some of the most important applications.

- Observational equivalence: logical relations prove that terms obtained by optimisation are equivalent.
- Compiler correctness: logical relations are employed to relate the source and target language.
- Security information flow: logical relations prove that the system prevents high security data to leak in low security output.

References

[1] H.P. Barendregt, W. Dekkers, R. Statman, Lambda Calculus with Types, Cambridge University Press, 2013.

[2] M. Dezani-Ciancaglini, S. Ghilezan and S. Likavec, Behavioural inverse limit models, Theoretical Computer Science 316:49-74 (2004).

[3] D. Doughert, S. Ghilezan and P. Lescanne, Characterizing strong normalization in the Curien-Herbelin symmetric lambda calculus: extending the Coppo-Dezani heritage, Theoretical Computer Science 398:114-128 (2008).

[4] S. Ghilezan, Strong normalization and typability with intersection types. Notre Dame Journal of Formal Logic 37:44–52 (1996).

[5] S. Ghilezan, J. Ivetić, P. Lescanne, S. Likavec, Intersection Types for the Resource Control Lambda Calculi. ICTAC 2011, Lecture Notes in Computer Science 6916:116-134 (2011).

[6] S. Ghilezan and S. Likavec, Reducibility: A Ubiquitous Method in Lambda Calculus with Intersection Types, Electronic Notes in Theoretical Computer Science 70 (2003).

[7] S. Ghilezan, V. Kuncak, Confluence of Untyped Lambda Calculus via Simple Typesi. ICTCS 2001, Lecture Notes in Computer Science 2206:38-49 (2001).

[8] B. Pierce, Types and Programming Languages, MIT Press, 2002.

[9] R. Statman, Logical relations and the typed λ-calculus. Information and Control 65:85-97 (1985).

[10] W. W. Tait, Intensional interpretations of functionals of finite type I. Journal of Symbolic Logic 32:198–212 (1967).

$\mathsf{LF}_{\mathcal{P}}$ - A Logical Framework with External Predicates[2]

Furio Honsell, Università di Udine, Italy
Marina Lenisa, Università di Udine, Italy
Petar Maksimović, INRIA Sophia Antipolis Méditerranée, France and
Mathematical Institute of the Serbian Academy of Sciences and Arts, Serbia
Ivan Scagnetto, Università di Udine, Italy
Luigi Liquori, INRIA Sophia Antipolis Méditerranée, France

The Edinburgh Logical Framework LF, presented in [2], is a first-order constructive type theory featuring dependent types. It was first introduced as a *general meta-language for logics*, as well as a specification language for *generic proof-checking/proof-development environments*. In this abstract, we present an extension of LF with *predicates* and *oracle calls*, which is accomplished by defining a mechamism serving for the *locking* and *unlocking* of types and terms.

Following the standard specification paradigm in Constructive Type Theory, we define locked types using *introduction*, *elimination*, and *equality rules*. We introduce a lock *constructor* for building objects $\mathcal{L}^{\mathcal{P}}_{N,\sigma}[M]$ of type $\mathcal{L}^{\mathcal{P}}_{N,\sigma}[\rho]$, via the *introduction rule* $(O{\cdot}Lock)$, presented below, and a corresponding unlock *destructor*, $\mathcal{U}^{\mathcal{P}}_{N,\sigma}[M]$, and an *elimination rule* $(O{\cdot}Unlock)$ which allows elimination of the locked type constructor, under the condition that a specific predicate $\mathcal{P}$ is verified, possibly *externally*, on an appropriate *correct*, *i.e.* derivable, judgement.

$$\frac{\Gamma \vdash_\Sigma M : \rho \quad \Gamma \vdash_\Sigma N : \sigma}{\Gamma \vdash_\Sigma \mathcal{L}^{\mathcal{P}}_{N,\sigma}[M] : \mathcal{L}^{\mathcal{P}}_{N,\sigma}[\rho]}\ (Lock)$$

$$\frac{\Gamma \vdash_\Sigma M : \mathcal{L}^{\mathcal{P}}_{N,\sigma}[\rho] \quad \Gamma \vdash_\Sigma N : \sigma \quad \mathcal{P}(\Gamma \vdash_\Sigma N : \sigma)}{\Gamma \vdash_\Sigma \mathcal{U}^{\mathcal{P}}_{N,\sigma}[M] : \rho}\ (Unlock)$$

The *equality rule* for locked types amounts to a new form of reduction we refer to as *lock-reductor* ($\mathcal{L}$-reduction), $\mathcal{U}^{\mathcal{P}}_{N,\sigma}[\mathcal{L}^{\mathcal{P}}_{N,\sigma}[M]] \to_{\mathcal{L}} M$, which allows elimination of a *lock*, in the presence of an *unlock*. The $\mathcal{L}$-reduction combines with standard β-reduction into $\beta\mathcal{L}$-reduction.

$\mathsf{LF}_{\mathcal{P}}$ is parametric over a potentially unlimited set of predicates $\mathcal{P}$, which are defined on derivable typing judgements of the form $\Gamma \vdash_\Sigma N : \sigma$. The syntax of $\mathsf{LF}_{\mathcal{P}}$ predicates is not specified, with the main idea being that their truth is to be verified via a *call* to an *external validation tool*; one can view this externalization as an *oracle call*. Thus, $\mathsf{LF}_{\mathcal{P}}$ allows for the invocation of external "modules" which, in principle, can be executed elsewhere, and whose successful verification can be acknowledged in the system via $\mathcal{L}$-reduction. Pragmatically, locked types allow for the factoring out of the complexity of derivations by delegating the {checking, verification, computation} of such predicates to an external proof engine or tool. The proof terms themselves do not contain explicit evidence for external predicates, but just record that a verification {has to be (lock), has been successfully (unlock)} carried out. In this manner, we combine the reliability of formal proof systems based on constructive type theory with the efficiency of other computer tools, in the style of the *Poincaré Principle* [5].

[2]This work was supported by the Serbian Ministry of Education, Science, and Technological Development (projects ON174026 and III044006).

In this abstract, we only outline the main results on $\mathsf{LF}_{\mathcal{P}}$, which have been treated in detail in [3, 4]. As far as meta-theoretic properties are concerned, strong normalization and confluence of the system have been proven without imposing any additional assumptions on the predicates. However, for subject reduction, we require the predicates to be *well-behaved*, *i.e.closed under weakening and permutation of the signature and context*, *substitution*, and $\beta\mathcal{L}$*-reduction* in the arguments. $\mathsf{LF}_{\mathcal{P}}$ is decidable, if the external predicates are decidable. Furthermore, a *canonical* presentation of $\mathsf{LF}_{\mathcal{P}}$ is constructed, in the style of [1, 6], based on a suitable extension of the notion of $\beta\eta$-long normal form, allowing for simple proofs of the adequacy of the encodings.

When it comes to illustrating the main benefits of $\mathsf{LF}_{\mathcal{P}}$ in practice, we have provided a number of relevant encodings. We have encoded in $\mathsf{LF}_{\mathcal{P}}$ the call-by-value λ-calculus, as well as its extension supporting the *design-by-contract* paradigm. We also provide smooth encodings of side conditions in the rules of Modal Logics, both in Hilbert and Natural Deduction styles, and also show how to encode sub-structural logics, *i.e.*non-commutative Linear Logic. We also illustrate how $\mathsf{LF}_{\mathcal{P}}$ can naturally support *program correctness* systems and Hoare-like logics. We show that other related systems can be embedded into $\mathsf{LF}_{\mathcal{P}}$ via locked types, and provide pseudo-code for some of the used predicates.

As far as expressiveness is concerned, $\mathsf{LF}_{\mathcal{P}}$ is a stepping stone towards a general theory of *shallow vs. deep encodings*, with our encodings being shallow by definition. Clearly, by Church's thesis, all external decidable predicates in $\mathsf{LF}_{\mathcal{P}}$ can be encoded, possibly with very deep encodings, in standard LF. It would be interesting to state in a precise categorical setting the relationship between such deep internal encodings and the encodings in $\mathsf{LF}_{\mathcal{P}}$.

$\mathsf{LF}_{\mathcal{P}}$ can also be viewed as a neat methodology for separating the logical-deductive contents from, on one hand, the verification of structural and syntactical properties, which are often needlessly cumbersome but ultimately computable, or, on the other hand, from more general means of validation.

From a philosophical point of view, the mechanism of *locking and unlocking types* in the presence of *external oracles*, which we are introducing in $\mathsf{LF}_{\mathcal{P}}$, effectively opens up the Logical Framework to alternate means of providing evidence for judgements. In standard LF, there are only two ways of providing this evidence, namely discovering types to be inhabited or postulating that types are inhabited by introducing appropriate constants. The *locked/unlocked types* of $\mathsf{LF}_{\mathcal{P}}$ open the door to an intermediate level, one provided by external means, such as computation engines or automated theorem proving tools. However, among these, one could also think of graphical tools based on neural networks, or even intuitive visual arguments, as were used in ancient times for giving the first *demonstrations* of the Pythagoras' theorem, for instance. In a sense, $\mathsf{LF}_{\mathcal{P}}$, by allowing formal accommodation of any alternative proof method to pure axiomatic deduction, vindicates all of the "proof cultures" which have been used pragmatically in the history of mathematics, not only in the Western tradition.

The traditional LF answer to the question "What is a Logic?" was: "A signature in LF". In $\mathsf{LF}_{\mathcal{P}}$, we can give the homologue answer, namely "A signature in $\mathsf{LF}_{\mathcal{P}}$", since external predicates can be read off the types occurring in the signatures themselves. But, we can also use this very definition to answer a far more intriguing question:

"What is a Proof Culture?".

References

[1] R. Harper and D. Licata. Mechanizing metatheory in a logical framework. *Journal of Functional Programming*, 17:613–673, 2007.

[2] Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. *Journal of the ACM*, 40:143–184, January 1993.

[3] F. Honsell, M. Lenisa, L. Liquori, P. Maksimovic, and I. Scagnetto. LF_P – a logical framework with external predicates. In *Proceedings of LFMTP 2012*, pages 13–22. ACM Digital Library, 2013.

[4] Furio Honsell, Marina Lenisa, Luigi Liquori, Petar Maksimović, and Ivan Scagnetto. An open logical framework. *Journal of Logic and Computation*, 2013. DOI:10.1093/logcom/ext028.

[5] H. Poincaré. *La Science et l'Hypothèse*. Flammarion, Paris, 1902.

[6] K. Watkins, I. Cervesato, F. Pfenning, and D. Walker. A Concurrent Logical Framework I: Judgments and Properties. Tech. Rep. CMU-CS-02-101, 2002.

Problems in formulating the consecution calculus of contraction–less relevant logics

Mirjana Ilić and Branislav Boričić
Faculty of Economics, University of Belgrade, Serbia

The contraction–less logic $RW^{\circ t}$ is the best known relevant system R ([1], p. 341) with co–tenability $\circ$ and t, but without the contraction axiom:

$$(W) \quad (\alpha \to .\alpha \to \beta) \to .\alpha \to \beta$$

The first problem in formulating a consecution calculus of $RW^{\circ t}$ is common to all relevant systems: how to enable the inference of $\alpha \wedge (\beta \vee \gamma). \to .(\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$, in the absence of thinning. This problem is solved by Dunn [4] and Minc [5], by allowing two types of sequences of formula: intensional (usually denoted by $(\Gamma_1; \ldots; \Gamma_n)$) and extensional ones (usually denoted by $(\Gamma_1, \ldots, \Gamma_n)$), which must be allowed to be nested within another. Due to the presence of these two types of sequences, every Gentzen structural rule can be formulated either as the intensional or as the extensional one. The contraction axiom (W) corresponds to Intensional Contraction. The missing thinning rule corresponds to Intensional Thinning structural rule, therefore in a sequent system of $RW^{\circ t}$, in addition to the structural rule Intensional Thinning, we also lack Intensional Contraction. The extensional variants of those rules should be present. The proof of the above distribution rule, in the single conclusion sequent system, is then:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\alpha \vdash \alpha}{\alpha, \beta \vdash \alpha}\,(\mathrm{KE}\vdash) \quad \dfrac{\beta \vdash \beta}{\alpha, \beta \vdash \beta}\,(\mathrm{KE}\vdash)}{\alpha, \beta \vdash \alpha \wedge \beta}\,(\vdash \wedge) \qquad \vdots}{\alpha, \beta \vdash (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)}\,(\vdash \vee) \qquad \alpha, \gamma \vdash (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)}{\alpha, \beta \vee \gamma \vdash (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)}\,(\vee \vdash)}{\alpha \wedge (\beta \vee \gamma), \alpha \wedge (\beta \vee \gamma) \vdash (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)}\,(\wedge \vdash)}{\alpha \wedge (\beta \vee \gamma) \vdash (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)}\,(\mathrm{WE}\vdash)}{\vdash \sim\sim \alpha \wedge (\beta \vee \gamma). \to .(\alpha \wedge \beta) \vee (\alpha \wedge \gamma)}\,(\vdash \to)$$

Following that idea, Giambrone [3] established the (single–conclusion) cut–free Gentzenisation of $RW^{\circ t}_{+}$ (positive $RW^{\circ t}$).

The second problem is how to enable the inference of $\sim\sim \alpha \to \alpha$. Originally, Gentzen allowed multiple-conclusion sequents:

$$\dfrac{\dfrac{\dfrac{\alpha \vdash \alpha}{\vdash \sim \alpha; \alpha}\,(\vdash \sim)}{\sim\sim \alpha \vdash \alpha}\,(\sim \vdash)}{\vdash \sim\sim \alpha \to \alpha}\,(\vdash \to)$$

But, for RW it won't work. Brady [2] found that the Gentzenisation of RW requires careful addition of negation to Gentzenisation of $RW^{\circ t}_{+}$. Really, in the multiple–conclusion sequent system, in the style of Gentzen, but with intensional and extensional sequences of formula, we would have:

$$\dfrac{\dfrac{\pi_1 \qquad \pi_2}{\Gamma_1 \vdash \alpha; \Delta_1 \qquad \Gamma_1 \vdash \beta \vee \gamma; \Delta_1}\,(\vdash \wedge)}{\Gamma_1 \vdash \alpha \wedge (\beta \vee \gamma); \Delta_1} \qquad \dfrac{\dfrac{\pi_3}{\Gamma_2; (\alpha, \beta \vee \gamma) \vdash \Delta_2}\,(\wedge \vdash)}{(\Gamma_2; \alpha \wedge (\beta \vee \gamma)), (\Gamma_2; \alpha \wedge (\beta \vee \gamma)) \vdash \Delta_2}\,(\mathrm{WE}\vdash)}{\Gamma_2; \alpha \wedge (\beta \vee \gamma) \vdash \Delta_2}\,(\mathrm{cut})$$

$$\Gamma_1; \Gamma_2 \vdash \Delta_1; \Delta_2$$

Our attempt to transform this proof to a cut–free proof, would lead to:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\pi_2}{\Gamma_1 \vdash \beta \vee \gamma; \Delta_1} \quad \dfrac{\dfrac{\pi_1}{\Gamma_1 \vdash \alpha; \Delta_1} \quad \dfrac{\pi_3}{\Gamma_2; (\alpha, \beta \vee \gamma) \vdash \Delta_2}}{\Gamma_2; (\Gamma_1, \beta \vee \gamma) \vdash \Delta_1; \Delta_2}\,(\text{cut})}{\Gamma_2; (\Gamma_1, \Gamma_1) \vdash \Delta_1; \Delta_1; \Delta_2}\,(\text{cut})}{\Gamma_2; \Gamma_1 \vdash \Delta_1; \Delta_1; \Delta_2}\,(\text{WE} \vdash)}{\cdots}\,\text{permutations}}{\Gamma_1; \Gamma_2 \vdash \Delta_1; \Delta_1; \Delta_2}$$

However, in the absence of extensional thinning, from here it is not possible to derive a sequent $\Gamma_1; \Gamma_2 \vdash \Delta_1; \Delta_2$, for non–empty Δ_1.

Brady solved this problem by formulating the single–conclusion sequent system of $RW^{\circ t}$ based on *signed* formulae $T\alpha$ and $F\alpha$, instead of just formulae α, with logical rules for both types of signed formulae. In his system, instead of the above derivation, we would have the following ($S\gamma$ stands in either for $T\gamma$ or $F\gamma$, or is empty):

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\pi_2}{\Gamma' \vdash T\beta \vee \gamma} \quad \dfrac{\dfrac{\pi_1}{\Gamma' \vdash T\alpha} \quad \dfrac{\pi_3}{\Gamma''; (T\alpha, T\beta \vee \gamma) \vdash S\gamma}}{\Gamma''; (\Gamma', T\beta \vee \gamma) \vdash S\gamma}\,(\text{cut})}{\Gamma''; (\Gamma', \Gamma') \vdash S\gamma}\,(\text{cut})}{\Gamma''; \Gamma' \vdash S\gamma}\,(\text{WE} \vdash)}{\cdots}\,\text{permutations}}{\Gamma'; \Gamma'' \vdash S\gamma}$$

However, Brady's system is very complicated (e. g., there are 8 different forms of the rule for $\rightarrow$, i. e., it has four shemata for the rule $(T \rightarrow \vdash)$, two for $(F \rightarrow \vdash)$, one for $(\vdash F \rightarrow)$ and one rule for $(\vdash T \rightarrow)$, which is, unusually, two–premise rule).

We propose another solution. Namely, we change the vocabulary. Instead of $\sim$, we take a propositional constant f as primitive (and we define negation, as usual, via $\sim \alpha =_{\text{df}} \alpha \rightarrow f$). We define a cut–free right–handed sequent system of $RW^{\circ tf}$, based on (just) formulae. $RW^{\circ tf}$ is obtained by adding co–tenability $\circ$ and propositional constants t and f to positive RW via the axioms given in [1] pp. 343.-344.

References

[1] A. ANDERSON, N. BELNAP JR., Entailment: the logic of relevance and necessity, vol. 1, Princeton University Press, Princeton, New Jersey, 1975.

[2] R. T. BRADY, *The Gentzenization and decidability of RW,* Journal of Philosophical Logic, 19, 35-73, 1990.

[3] S. GIAMBRONE, TW_+ *and* RW_+ *are decidable,* Journal of Philosophical Logic, 14, 235-254, 1985.

[4] J. M. DUNN, *A 'Gentzen system' for positive relevant implication,* The Journal of Symbolic Logic 38, pp. 356-357, 1973.

[5] G. MINC, *Cut elimination theorem for relevant logics,* Journal of Soviet Mathematics 6, 422-428, 1976.

Timed Collaborative Systems with Real Time

Max Kanovich, Queen Mary, University of London, UK
Tajana Ban Kirigin, University of Rijeka, HR
Vivek Nigam, Federal University of Paraíba, João Pessoa, Brazil
Andre Scedrov, University of Pennsylvania, Philadelphia, USA

Often one needs to reason about time when setting the rules and goals of a collaboration. For example, agents need to comply with deadlines and react quickly under unexpected events. Although in many situations it is enough to represent time discretely [3], such as in days or hours, sometimes one needs real time.

For instance, in many situations, an agent A needs to know whether another agent B is near, within a radius. A way to determine this is by calculating the round time of a message: A sends a challenge message m to B and remembers the time t_0 when the message was sent. Then once B receives the message m, it computes a response message $f(m)$ and sends it as quickly as possible back to A. When this response message reaches agent A, at some time t_1, A checks whether the round time $t_1 - t_0$ is less than the given threshold. If this is the case, then A can assume that B is within some radius. Otherwise A cannot conclude anything about how distant B is. In fact, this is the basic principle of Distance Bounding Protocols [1].

In order to formally specify and verify collaborative systems involving real time, such as the scenario above, and also to verify whether it is possible for an intruder to appear to be someone else or to appear closer than he actually is, one needs formal models that can mention real time and can *generate fresh values*. Fresh values, also called nonces in protocol security literature [2], are used so that messages sent in previous interaction between agents are not mixed up with current ones. They are used for instance in the authentication of agents.

This paper proposes a rewriting framework that can be used to specify collaborative systems equipped with real time and where agents may create fresh values. It extends our previous work on Timed Local State Transition Systems (TLSTSes) with explicit time [3] where only discrete time was allowed. Modelling real time in TLSTS was left as future work in [3]. Here, we outline the extension of the model and describe the fragment for which the reachability problem is PSPACE-complete.

Rewriting Model TLSTSes are multiset rewriting systems. The state of the system or a system configuration is represented by a multiset of facts. Facts are atomic formulas with a positive real number called *timestamp* associated to each fact. Agents change the state of the system by applying actions. Sequences of actions or *plans* are compliant if, starting from a given initial configuration, they lead to a goal configuration without reaching any configuration that is considered critical. The main problem when studying TLSTSes is the *planning problem*: Given a timed local state transition system $\mathcal{T}$, an initial configuration W and a finite set of goal and critical configurations, is there a compliant plan? In this paper we address the complexity of the planning problem for TLSTSes with real time.

In TLSTSes time is modelled through timestamps attached to facts, through a special fact *Time* representing global time, and through time constrains that can be associated with actions and with configurations. More precisely, a *timestamp* is a positive real number attached to a fact. It can represent time in various ways, for example it can denote the time when the fact was created, or the time time until the fact is valid etc.

Time constraints are arithmetic comparisons involving exactly two timestamps:

$$T_1 = T_2 \pm a,\ T_1 > T_2 \pm a,\ \text{or}\ T_1 \geq T_2 \pm a, \tag{1}$$

where a is a *natural number* and T_1 and T_2 are time variables, which may be instantiated by the timestamps of any fact including the global time.

Action application can also have time conditions. Time constraints can be attached to actions, to act as a guard of the rule, that is, an action can only be applied if the attached time constraints are all satisfied. Only two types of actions are allowed. The first is the following type of action that increments the global time of a configuration by a *positive real number* t:

$\mathit{Time}@T \mid \{\} \rightarrow_{clock} \mathit{Time}@(T+t)$.

This is the only action that can modify the global time of a configuration. Actions of the second type are instantaneous and have necessarily the following form:

$\mathit{Time}@T, W \mid \Upsilon \ \longrightarrow_A \ \exists \vec{x}.\mathit{Time}@T, W'$

where Υ is the guard of the action containing a finite set of constraints. We restrict actions so that all variables appearing in Υ are contained in the set of time variables $\{T_1, \ldots, T_n, T\}$ from the pre-condition. We further impose the following condition on these actions: if $\mathit{Time}@T$ is in the pre-condition W, then all facts created in the post-condition W' are of the form $P@(T+d)$, where d is a *natural number*, possibly zero. That is, all the created facts have timestamps greater or equal to the global time. The existentially quantified variables in a rewrite rule specify the creation of fresh values.

Goals and critical configurations are specified similarly, by allowing one to attach time constraints to them, exactly as in [3].

Complexity When considering the complexity of the planning problem with real time, one has to deal with the unboundedness of time and with the density of time. In our previous work with discrete time [3], we show how to tackle the unboundedness of time, *i.e.*, an internally infinite space of configurations, by using a finite number of δ-representations of configurations. Instead of the actual values of timestamps, δ-representations contain only relative time differences truncated by an upper bound D_{max} deduced from the system specification. It is necessary to assume that the size of facts is bounded and that the timed local state transition system is balanced, that is pre and post-conditions of all actions have the same number of facts.

This approach alone does not work when timestamps are real numbers, as there is an infinite number of possible values for relative time differences. To address the density of time, we introduce the novel equivalence relation among configurations, inspired by [4]. This provides a bounded number of classes called *circle-abstractions*. Similar to [3], we define the δ-configuration of a configuration $\mathcal{S}$ and a natural number D_{max} as follows: it is the list $[F_1, \delta_1, F_2, \ldots, F_{n_1}, \delta_{n-1}, F_n]$ of its facts, $F_1, \ldots, F_n$, ordered according to the values of their timestamps, interleaved by the value δ_i obtained by the truncated time differences w.r.t. D_{max} of the corresponding two neighbouring facts F_i and F_{i+1}.

Definition 1. *Two configurations $\mathcal{S}_1$ and $\mathcal{S}_2$ are equivalent for a given natural number D_{max} if the following two conditions are satisfied: (1 - δ-configurations) Their δ-configurations w.r.t. D_{max} are the same when considering only the integer part of the time differences; and (2 - Circle) when their facts are ordered using only the decimal part of timestamps, one obtains the same list of facts. (If they have the same value, then we indicate this in the list by using the symbol $=$.)*

For instance, the following two configurations are equivalent when $D_{max} = 2$: $\{P_0@0.4, P_1@1.5, \mathit{Time}@5.4, P_2@6.6\}$ and $\{P_0@3.2, P_1@4.5, \mathit{Time}@8.2, P_2@9.6\}$ as they have the same integer parts of truncated relative times, represented by the following δ-configuration $[P_0, 1, P_1, \infty, \mathit{Time}, 1, P_2]$ and the same circle configuration, namely $[\mathit{Time} = P_0, P_1, P_2]$.

We show that circle-abstractions are well-defined with respect to the planning problem. This allows us to represent plans using circle-abstractions only. In particular, we show that all configurations that have the same circle-abstraction satisfy the same time constraints. We extend action application to circle-abstractions. Since abstractions do not contain the information of exact time differences between timestamps and the current time, we cannot apply time incrementing action for a concrete value t to circle-abstractions. In order to model time advancement we add a special action $next$. Application of $next$ results in the circle-abstraction in which the time has shifted just enough to change the abstraction, and hasn't shifted too much to jump over some abstractions as time advances.

We show that our formalization of circle-abstractions is sound and complete and, therefore, we conclude that any given planning problem can be conceived as a planning problem over circle-abstractions. For the complexity proofs it is essential to show that for a given planning problem we obtain only a finite number of circle-abstractions with which we are able to represent an infinite space of configurations.

We finally show that in balanced TLSTSes with real time, when the size of facts is bounded and actions are balanced, the planning problem is PSPACE-complete.

Acknowledgments: We thank Catherine Meadows, John Mitchell, and Carolyn Talcott for helpful discussions. This material is based upon work supported by the MURI program under AFOSR Grant No: FA9550-08-1-0352 and upon work supported by the MURI program under AFOSR Grant No. FA9550-11-1-0137. Additional support for Scedrov from NSF Grant CNS-0830949 and from ONR grant N00014-11-1-0555. Nigam was partially supported by the Alexander von Humboldt Foundation and CNPq. Kanovich was partially supported by the EPSRC.

References

[1] S. Brands and D. Chaum. Distance-bounding protocols. In EUROCRYPT, 1993.

[2] N. A. Durgin, P. Lincoln, J. C. Mitchell, and A. Scedrov. Multiset rewriting and the complexity of bounded security protocols. *Journal of Computer Security*, 12(2):247–311, 2004.

[3] M. I. Kanovich, T. B. Kirigin, V. Nigam, A. Scedrov, C. L. Talcott, and R. Perovic. A rewriting framework for activities subject to regulations. In RTA, 2012.

[4] M. I. Kanovich, M. Okada, and A. Scedrov. Specifying real-time finite-state systems in linear logic. *Electr. Notes Theor. Comput. Sci.*, 16(1):42–59, 1998.

Multiple conclusion deductions in classical logic

Marcel Maretić, University of Zagreb, Croatia

Natural deduction systems are, unlike Gentzen's sequent calculus, not related to semantic trees. Natural deductions arise from syntactic approach to logic – from proof search and inference rules. In that sense they are adequate for minimal and intuitionistic logic. Addition of *tertium non datur* (TND) or *reduction ad absurdum* (RAA) yields deduction calculus for classical logic.

Kneale in [2] proposes multiple conclusion deductions as an elegant and symmetrical version of deduction calculus that provides a good fit for classical logic. Kneale's inference rules are local – hypotheses are never discharged. Proofs, which Kneale calls *developments*, are formula trees branching downward *and upward*.

Kneale's calculus of developments is not complete. Shoesmith and Smiley in [1] propose adjustments for completion of the calculus. Our approach to multiple conclusion calculus is simple and better motivated. Unlike Shoesmith and Smiley, who in [1] motivate multiple conlcusion deductions syntactically, we relate multiple conclusion deductions to semantic trees. We present an elegant and analytic proof search for multiple conclusion deductions.

Essential steps of the algorithm are:

(1) analysis: construction of analytic deductions;

(2) synthesis: matching of analytic deductions that completes the proof search.

Thus, multiple conclusion deductions are analytic in the sense that they yield a simple analytic proof search (as in [3]).

Steps of the proof search algorithm can be motivated semantically. Analysis (step 1) corresponds to semantic analysis and branching of a clausal semantic tree, whereas synthesys (step 2) corresponds to branch closing on the clausal semantic tree. Therefore, proof search for multiple conclusion deductions is algorithmically equivalent to Beth's semantic trees.

References

[1] Shoesmith D.J., Smiley T.J., *Multiple Conclusion Logic*, Cambridge University Press, 1978.

[2] Kneale W., Kneale M., *Development of Logic*, Clarendon Press, 1956, pp. 538–548.

[3] Smullyan R. M., *First Order Logic*, Courier Dover Publications, 1995.

On Certain Problems of Cryptology and Computational Complexity of Processing over Uncertain Data

Miodrag J. Mihaljević, Mathematical Institute of the Serbian Academy of Sciences and Arts, Belgrade, Serbia and Chuo University, Tokyo, Japan
Hideki Imai, Chuo University, Tokyo, Japan

Abstract. A link between certain problems of cryptology and mathematical logic is pointed out. Computational complexity of the Learning Parity in Noise (LPN) problem is discussed as an example of the reasoning over uncertain data.

Introduction

This paper links a real-life problem of growing importance and a topic of mathematical logic. We consider certain issues of security within cyber-space and a topic of mathematical logic dedicated to reasoning over uncertain data and corresponding computational complexity. Accordingly, an application of the topics of mathematical logic to cryptology is pointed out. Section 2 summarizes some emergency issues of cyber-security which imply request for employment of low-complexity cryptographic techniques. A mathematical problem called Learning Parity in Noise (LPN) relevant for design of low-complexity and highly secure cryptographic primitives is summarized in Section 3. Finally, section 4 addresses some issues of the LPN problem complexity which are also challenges regarding reasoning over uncertain data.

Preliminaries

Overheads Implied by Cryptographic Techniques. Our society strongly depends on information-communications technologies (ICT) and the security of ICT has been recognized as one of the top priorities in order to minimize impacts of potential attempts regarding misuse of ICT with disastrous consequences. Accordingly, we face an extensive employment of the security mechanisms and as a consequence we face significant overheads to the main functionality of the systems implied by the employed security mechanisms. Reduction of these security related overheads is of a top interest because cumulative effect of all these overheads has a (very) significant cost. A part of these overheads corresponds to the cryptographic techniques employed in the security mechanisms. Accordingly, reduction of the security overheads implied by cryptographic algorithms appear as an issue of very high importance. On the other hand, minimization of the "cryptographic overheads" should not jeopardize the cryptographic security, and design of highly secure cryptographic algorithms and protocols which minimize the overheads is still a challenge and an emergency issue.
Lightweight and Provably Secure Cryptographic Primitives. The main overheads implied by cryptographic techniques correspond to: (i) implementation overheads (required additional software/hardware); (ii) computational overheads for performing cryptographic operations; (iii) power-consumption overheads regarding cryptographic processing. Cryptographic techniques which provide minimization of the overheads are called light-weight cryptographic techniques. On the other hand side, a claim that a cryptographic primitive is provably secure means that assumption of its insecurity implies that certain hard mathematical problem can be solved (employing certain algorithm for cryptanalysis) implying a contradiction and a justification of the security. Note that When instead of a provably secure construction a heuristically secure approach is employed we could face iterative improvements of exploring the vulnerabilities with serious security consequences (as an illustration see [7]-[9])

Learning Parity in Noise (LPN) Problem

LPN problem has been recognized as an underlying approach for constructions of light-weight and provably secure cryptographic primitives (see [10], for example). Informally, the LPN problem a problem of solving a probabilistic overdefined consistent system of linear equations over GF(2) where the right side of each equation is true with the known probability $p > 1/2$ (typically $p < 0.25$). One of its incarnations is the problem of decoding of a random linear binary block code.

Definition: LPN Search Problem. Let s be a random binary string of length l. We consider the Bernoulli distribution $\mathcal{B}_\theta$ with parameter $\theta \in (0, 1/2)$. Let $\mathcal{Q}_{s,\theta}$ be the following distribution:

$$\{(a, < s, a > \oplus e)| \leftarrow \{0,1\}^l, e \leftarrow \mathcal{B}_\theta\} \ .$$

For an adversary $\mathcal{A}$ trying to discover the random string s, we define its advantage as

$$\mathrm{Adv}_{\mathrm{LPN}_{\theta,\mathcal{A}}}(l) = \Pr[\mathcal{A}^{\mathcal{Q}_{s,\theta}} = s | s \leftarrow \{0,1\}^l] \ .$$

The LPN_θ problem with parameter θ is hard if the advantage of adversaries $\mathcal{A}$ that make a polynomial number of oracle queries is negligible.

In [5] a distinguishing variant of the problem has been introduced, which is more useful in the context of encryption schemes. Roughly speaking, the decisional LPN problem asks to distinguish a number of noisy samples of a linear function (specified by a secret vector x) from uniform random. The problem is, given $\mathbf{A}$ and $\mathbf{y}$, to decide whether $\mathbf{y}$ is distributed according to $\mathbf{A} \cdot \mathbf{x} \oplus \mathbf{e}$ or chosen uniformly at random.

Definition: LPNDP - LPN Decisional (Distingushing) Problem. Let s, a be binary strings of length l. Let further $\mathcal{Q}_{s,\theta}$ be as in Definition of the LPN search problem. Let $\mathcal{A}$ be a adversary. The distinguishing-advantage of $\mathcal{A}$ between $\mathcal{Q}_{s,\theta}$ and the uniform distribution $\mathcal{U}_{l+1}$ is defined as

$$\mathrm{Adv}_{\mathrm{LPNDP}_{\theta,\mathcal{A}}}(l) = \Pr[\mathcal{A}^{\mathcal{Q}_{s,\theta}} = s | s \leftarrow \{0,1\}^l] - \Pr[\mathcal{A}^{\mathcal{U}_{l+1}} = 1] \ .$$

The LPNDP_θ with parameter θ is hard if the advantage of adversaries $\mathcal{A}$ is negligible.

It has been shown in [5] that the distinguishing-problem is as hard as the search-problem with similar parameters.

Complexity of Reasoning over Uncertain Data

According to the definitions of the LPN search and distinguishing problems, they belong to a wider class of problems related to the reasoning over uncertain data.

It has been proved that in the worst-case, the problem of decoding a random liner binary block code is NP-complete [1] as well as the LPN problem in the worst case.

On the other hand side, it should be noted that the average case hardness of the LPN problems, cannot be reduced to the worst-case hardness of a NP-hard problem. The confidence on the hardness of solving LPN problems in average case appears from the lack of efficient solutions despite the efforts over the years. Currently, the best known algorithms for solving the LPN search problems are the one reported in [2] and its improvements/alternatives (see [6]. [3] and [4]). The BKW algorithm [2] has the complexity $2^{O(l/\log_2 l)}$ and its improvements/alternatives can provide further reduction of the exponent for a factor $\Delta(l, \theta)$ (see the Definitions of the LPN problems). The talk discuses complexity of the above mentioned algorithms and points out to the open challenges.

References

[1] E.R. Berlekamp, R.J. McEliece, and H.C.A. van Tilborg, "On the Inherent Intractability of Certain Coding Problems", *IEEE Trans. Info. Theory*, vol. 24, pp. 384-386, 1978.

[2] A. Blum, A. Kalai and H. Wasserman, "Noise-Tolerant Learning, the Parity Problem, and the Statistical Query Model", *Journal of the ACM*, vol. 50, no. 4, pp. 506-519, July 2003.

[3] M. Fossorier, M.J. Mihaljević, H. Imai, Y. Cui and K. Matsuura, "An Algorithm for Solving the LPN Problem and its Application to Security Evaluation of the HB Protocols for RFID Authentication", INDOCRYPT 2006, *Lecture Notes in Computer Science*, vol. 4329, pp. 48-62, Dec. 2006.

[4] M. Fossorier, M.J. Mihaljević and H. Imai, "Modeling Block Encoding Approaches for Fast Correlation Attack", *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4728-4737, Dec. 2007.

[5] J. Katz and J. Shin, "Parallel and Concurrent Security of the HB and HB+ Protocols" EUROCRYPT 2006, *Lecture Notes in Computer Science*, vol. 4004, pp. 73-87, 2006.

[6] E. Levieil and P.-A. Fouque, "An Improved LPN Algorithm", SCN 2006, *Lecture Notes in Computer Science*, vol. 4116, pp. 348-359, 2006.

[7] M.J. Mihaljević, S. Gangopadhyay, G. Paul and H. Imai, "State Recovery of Grain-v1 Employing Normality Order of the Filter Function", *IET Information Security*, vol. 6, no. 2, pp. 55-64, June 2012

[8] M.J. Mihaljević, S. Gangopadhyay, G. Paul and H. Imai, "Internal State Recovery of Keystream Generator LILI-128 Based on a Novel Weakness of the Employed Boolean Function", *Information Processing Letters*, vol. 112, no. 21, pp. 805-810, November 2012.

[9] M.J. Mihaljević, S. Gangopadhyay, G. Paul and H. Imai, "Generic Cryptographic Weakness of k-normal Boolean Functions in Certain Stream Ciphers and Cryptanalysis of Grain-128", *Periodica Mathematica Hungarica* (Selected Papers of 2011 Central European Conference on Cryptology), vol. 65, no. 2, pp. 205-227, Dec. 2012.

[10] K. Pietrzak, "Cryptography from Learning Parity with Noise", SOFSEM 2012, *Lecture Notes in Computer Science*, vol. 7147, pp. 99-114, 2012.

Hierarchies of probability logics

Zoran Ognjanović, Mathematical Institute of the Serbian Academy of Sciences and Arts, Belgrade, Serbia
Aleksandar Perović, Faculty of Transport and Traffic Engineering, University of Belgrade, Serbia
Miodrag Rašković, Mathematical Institute of the Serbian Academy of Sciences and Arts, Belgrade, Serbia
Nebojša Ikodinović, Faculty of Mathematics, University of Belgrade, Serbia

Abstract. Our aim is to present what we call the lower and the upper hierarchies of the real valued probability logics with probability operators of the form $P_{\geqslant s}$ and Q_F, where $s \in [0,1]_{\mathbb{Q}} = [0,1] \cap \mathbb{Q}$ and F is a recursive subset of $[0,1]_{\mathbb{Q}}$. The intended meaning of $P_{\geqslant s}\alpha$ is that the probability of α is at least s, while the intended meaning of $Q_F\alpha$ is that the probability of α is in F.

Introduction

The modern probability logics arose from the work of Jerome Keisler on generalized quantifiers and hyperfinite model theory in the mid seventies of the twentieth century [8]. Another branch of research that was involved with automatization of reasoning under uncertainty have led to development of numerous Hilbert style formal systems with modal like probability operators, see for instance [5, 2, 11, 13, 14, 17, 18, 20, 23, 24]. The simplest form of such representation of uncertainty does not allow iteration of probability operators, so formulas are Boolean combinations of the basic probability formulas, i.e. formulas of the form

$$\texttt{ProbOp}(\alpha_1, \ldots, \alpha_n),$$

where $\alpha_1, \ldots, \alpha_n$ are classical (propositional or predicate) formulas and `ProbOp` is an n-ary probability operator. Weighted probability formulas used by Fagin, Halpern and Megiddo in [2] can be treated as n-ary probability operators. For instance,

$$w(\alpha) + 3w(\beta) - 5w(\gamma) \geqslant 1$$

is example of a ternary probability operator.

The vast majority of those formal systems have unary or binary probability operators. The unary operators are used for statements about probability of classical formulas: for example we use

$$P_{\geqslant 3/4}(p \vee q)$$

to express "the probability of $p \vee q$ is at least 3/4", while

$$Q_{\{\frac{n}{n+1} \mid n \in \mathbb{N}\}}(p \vee q)$$

in our notation reads "the probability of $p \vee q$ is an element of the set $\{\frac{n}{n+1} \mid n \in \mathbb{N}\}$". The binary operators are usually used for the expression of conditional probability: for instance, we use

$$CP_{\geqslant 1/3}(p, q)$$

to express that the conditional probability of p given q is at least $1/3$.

Over the course of two decades we have developed various probability logics with the mentioned types of probability operators - an extensive survey including a uniform notation for logics is presented in [17]. The aim of this paper is to put the certain class of probability logics into the wider context of mathematical phenomenology - to compare mathematical concepts according to some natural criterion (expressive power, class of models, consistency strength and so on).

Here we will focus on the classification of two sorts of probability logics: $LPP_{2,P,Q,O}$ logics introduced in [12] and $LPP_2^{\mathrm{Fr}(n)}$ logics introduced in [3, 13, 17, 20, 24] (L for logic, the first P for propositional, and the second P for probability). Independently, several authors in [4, 6] have developed the fuzzy logics $FP(Ł_n)$ that extend Łukasiewicz logic. The $LPP_2^{\mathrm{Fr}(n)}$ logics can be embedded into those logics. For the $LPP_{2,P,Q,O}$ logics we introduce the comparison criterion with respect to the classes of models, while the $LPP_2^{\mathrm{Fr}(n)}$ logics we compare in terms of the interpretation method. We show that both criteria can be joined in a single one. Thus we have obtained the hierarchy of probability logics where the lattice of $LPP_{2,P,Q,O}$ logics is the end extension of the lattice of $LPP_2^{\mathrm{Fr}(n)}$ logics.

Acknowledgements

The authors are partially supported by Serbian ministry of education and science through grants III044006, III041103, ON174062 and TR36001.

References

[1] R. Djordjević, M. Rašković, Z. Ognjanović. Completeness theorem for propositional probabilistic models whose measures have only finite ranges. Archive for Mathematical Logic 43, 557 –563, 2004.

[2] R. Fagin, J. Halpern, N. Megiddo. A logic for reasoning about probabilities. Information and Computation 87(1–2), pp 78–128, 1990.

[3] M. Fattorosi-Barnaba and G. Amati. Modal operators with probabilistic interpretations I. Studia Logica 46(4), 383–393, 1989.

[4] T. Flaminio, L. Godo. A logic for reasoning about the probability of fuzzy events. Fuzzy Sets and Systems 158(6), 625–638, 2007.

[5] L. Godo, E. Marchioni. Coherent conditional probability in a fuzzy logic setting. Logic Journal of the IGPL, Vol. 14 No. 3, pp 457–481, 2006.

[6] P. Hajek, L. Godo, F. Esteva, Fuzzy Logic and Probability. In Proc. of UAI'95, Morgan–Kaufmann, 237–244, 1995.

[7] N. Ikodinović,, M. Rašković, Z. Marković, Z. Ognjanović. Measure logic. ECSQARU 2007: 128–138.

[8] H. J. Keisler. Probability quantifiers. In J. Barwise and S. Feferman, editors, Model–Theoretic Logics, Perspectives in Mathematical Logic, Springer–Verlag 1985.

[9] H. J. Keisler. Elementary calculus. An infinitesimal approach. 2nd edition, Prindle, Weber and Schmidt, Boston, Massachusets, 1986.

[10] D. Lehmann, M. Magidor. What does a conditional knowledge base entail? *Artificial Intelligence*, 55, 1–60, 1992.

[11] N. Nilsson. Probabilistic logic. Artif. Intell. 28, 71–78, 1986.

[12] Z. Ognjanović, M. Rašković. Some probability logics with new types of probability operators, J. Logic Computat., Vol 9 No. 2, pp 181–195, 1999.

[13] Z. Ognjanović, M. Rašković. Some first-order probability logics. Theoretical Computer Science 247(1–2), pp 191–212, 2000.

[14] Z. Ognjanović, Z. Marković, M. Rašković. Completeness Theorem for a Logic with imprecise and conditional probabilities. Publications de L'Institute Matematique (Beograd), ns. 78 (92) 35 – 49, 2005.

[15] Z. Ognjanović. Discrete linear-time probabilistic logics: completeness, decidability and complexity. J. Log. Comput. 16(2), pp 257–285, 2006.

[16] Z. Ognjanović, A. Perović, M. Rašković. Logics with the qualitative probabilty operator. Logic journal of the IGPL 16(2), 105–120, 2008.

[17] Z. Ognjanović, M. Rašković, Z. Marković. Probability Logics. Zbornik radova. Logic in Computer Science (edited by Z. Ognjanović), 12(20), 35–111, Mathematical Institute of Serbian Academy of Sciences and Arts, 2009. http://elib.mi.sanu.ac.rs/files/journals/zr/20/n020p035.pdf

[18] Z. Ognjanović, M. Rašković, Z. Marković, and A. Perović, On probability logic, The IPSI BgD Transactions on Advanced Research, 2– 7, Volume 8 Number 1, 2012. (ISSN 1820-4511)

[19] A. Perović, Z. Ognjanović, M. Rašković, Z. Marković. A probabilistic logic with polynomial weight formulas. FoIKS 2008, pp 239–252.

[20] M. Rašković. Classical logic with some probability operators. Publications de l'institut mathematique, Nouvelle série, tome 53(67), 1–3, 1993.

[21] M. Rašković, Z. Ognjanović. A first order probability logic LP_Q. Publications de l'institut mathematique, Nouvelle série, tome 65(79), pp 1–7, 1999.

[22] M. Rašković, Z. Ognjanović, Z. Marković. A logic with Conditional Probabilities. In J. Leite and J. Alferes, editors, 9th European Conference Jelia'04 Logics in Artificial Intelligence, volume 3229 of Lecture notes in computer science, pages 226–238, Springer-Verlag 2004.

[23] M. Rašković, Z. Ognjanović, Z. Marković. A logic with approximate conditional probabilities that can model default reasoning. *Int. J. Approx. Reasoning* 49(1): 52–66, 2008.

[24] W. van der Hoek. Some considerations on the logic $P_F D$: a logic combining modality and probability. Journal of Applied Non-Classical Logics, 7(3), 287–307, 1997.

Collaborative Systems

Andre Scedrov, University of Pennsylvania, USA

We discuss a model of collaboration, introduced in a joint work with Kanovich and Rowe, in which the participants are unwilling to share all their informationwith each other, but some information sharing is unavoidable when achieving a common goal. The need to share information and the desire to keep it confidential are two competing notions which affect the outcome of a collaboration. Our model is based on the notion of a plan which originates in the AI literature. We also consider an extension of the model which allows for updates of values with fresh ones, such as updating a password.

All the players inside our system, including potential adversaries, have similar capabilities. They have bounded storage capacity, that is, they can only remember a bounded number of facts. This is technically imposed by allowing only the so-called balanced actions, that is, actions that have the same number of facts in their pre and post conditions. We investigate the complexity of the planning problem, whether the players can reach a goal while avoiding certain critical configurations along the way. We show that this problem is PSPACE-complete. The complexity is lowered to NP-completeness for the class of so-called progressing collaborative systems, intended to describe administrative processes, which normally have a progressing nature: once an item in an activity to-do list is checked, that activity is not repeated.

As an application we turn to network security protocol analysis and demonstrate that when an adversary has enough storage capacity, then many known protocol anomalies can also occur in the presence of a bounded memory intruder. We believe that precisely this is a theoretical reason for the successful use in the past years of model checkers in security protocol verification. In particular, the known anomalies arise for bounded memory protocols, where there is only a bounded number of concurrent sessions and the honest participants of the protocol cannot generate an unbounded number of facts nor an unbounded number of fresh values. This led us to the question of whether it is possible to infer an upper-bound on the memory required by the adversary to carry out an anomaly from the memory restrictions of the bounded protocol. We answer this question negatively. This is joint work with Max Kanovich, Tajana Ban Kirigin, and Vivek Nigam.

Cut-elimination for modal fixed point logics

Thomas Studer, Institut für Informatik und angewandte Mathematik, Universität Bern, Switzerland

Modal fixed point logics with additional constructors for fixed points occur in many different places in computer science. For instance, there are temporal logics with an always operator, epistemic logics with a common knowledge operator, program logics with an iteration operator, and the propositional modal μ-calculus with fixed points for arbitrary positive formulas.

While the model-theoretic side of modal fixed point logics is very well investigated, we do not know much about the proof theory of these logics. In this talk we will survey syntactic cut-elimination results for modal logics with fixed points.

Most of these results make use of deep inference where rules may not only be applied to outermost connectives but also deeply inside formulas. The first result of this kind has been obtained by Pliuskevicius [10] who presents a syntactic cut-elimination procedure for linear time temporal logic. Brünnler and Studer [1] employ nested sequents to develop a cut-elimination procedure for the logic of common knowledge. Hill and Poggiolesi [6] use a similar approach to establish effective cut-elimination for propositional dynamic logic. A generalization of this method is studied in [2] where, however, it is also shown that it cannot be extended to fixed points that have a $\Box$-operator in the scope of a μ-operator. Fixed points of this kind occur, for instance, in CTL in the form of universal path quantifiers.

Thus we need a more general approach to obtain syntactic cut-elimination for the modal μ-calculus. A standard proof-theoretic technique to deal with inductive definitions and fixed points is Buchholz' Ω-rule [3, 5]. Jäger and Studer [7] present a formulation of the Ω-rule for non-iterated modal fixed point logic and they obtain cut-elimination for positive formulas of this logic. In order to overcome this restriction to positive formulas, Mints [8] introduces an Ω-rule that has a wider set of premises, which enables him to obtain full cut-elimination for non-iterated modal fixed point logic.

Mints' cut-elimination algorithm makes use of, in addition to ideas from [4], a new tool presented in [8]. It is based on the distinction, see [11], between implicit and explicit occurrences of formulas in a derivation with cut. If an occurrence of a formula is traceable to the endsequent of the derivation, then it is called explicit. If it is traceable to a cut-formula, then it is an implicit occurrence.

Implicit and explicit occurrences of greatest fixed points are treated differently in the translation of the induction rule to the infinitary system. An instance of the induction rule that derives a sequent $\nu X.A, B$ goes to an instance of the ω-rule if $\nu X.A$ is explicit. Otherwise, if $\nu X.A$ is traceable to a cut-formula, the induction rule is translated to an instance of the Ω-rule that is preserved until the last stage of cut-elimination. At that stage, called collapsing, the Ω-rule is eliminated completely. Recently, Mints and Studer [9] showed that this method can be extended to a μ-calculus with iterated fixed points. Hence they obtain complete syntactic cut-elimination for the one-variable fragment of the modal μ-calculus.

References

[1] Kai Brünnler and Thomas Studer. Syntactic cut-elimination for common knowledge. *Annals of Pure and Applied Logic*, 160(1):82–95, 2009.

[2] Kai Brünnler and Thomas Studer. Syntactic cut-elimination for a fragment of the modal mu-calculus. *Annals of Pure and Applied Logic*, 163(12):1838–1853, 2012.

[3] Wilfried Buchholz. The $\Omega_{\mu+1}$-rule. In Wilfried Buchholz, Solomon Feferman, Wolfram Pohlers, and Wilfried Sieg, editors, *Iterated Inductive Definitions and Subsystems of Analysis: Recent Proof Theoretic Studies*, volume 897 of *Lecture Notes in Mathematics*, pages 189–233. Springer, 1981.

[4] Wilfried Buchholz. Explaining the Gentzen-Takeuti reduction steps: a second-order system. *Archive for Mathematical Logic*, 40(4):255–272, 2001.

[5] Wilfried Buchholz and Kurt Schütte. *Proof Theory of Impredicative Subsystems of Analysis*. Bibliopolis, 1988.

[6] Brian Hill and Francesca Poggiolesi. A contraction-free and cut-free sequent calculus for propositional dynamic logic. *Studia Logica*, 94(1):47–72, 2010.

[7] Gerhard Jäger and Thomas Studer. A Buchholz rule for modal fixed point logics. *Logica Universalis*, 5:1–19, 2011.

[8] Grigori Mints. Effective cut-elimination for a fragment of modal mu-calculus. *Studia Logica*, 100(1–2):279–287, 2012.

[9] Grigori Mints and Thomas Studer. Cut-elimination for the mu-calculus with one variable. In *Fixed Points in Computer Science 2012*, volume 77 of *EPTCS*, pages 47–54. Open Publishing Association, 2012.

[10] Regimantas Pliuskevicius. Investigation of finitary calculus for a discrete linear time logic by means of infinitary calculus. In *Baltic Computer Science, Selected Papers*, pages 504–528. Springer, 1991.

[11] Gaisi Takeuti. *Proof Theory*. North-Holland, 1987.

Gödel's Incompletness Theorem and Man-Machine Non-equivalence

Zvonimir Šikić, University of Zagreb, Croatia

For many, it is still hard to conceive how the world of subjective experiences spring out of merely physical events. This problem of qualia is the hardest and the main part of the mind-body problem. The problem is often summed up in the following question: "How matter (i.e. body and brain) becomes mind." All sorts of dualists think it never does and some of them, like Lucas [2] and Penrose [3], think that Gödel's incompleteness theorem proves that. Their main argument is that Gödel's theorem implies man-machine non-equivalence in the following sense:

There is no machine which could capture all our mathematical intuitions.

Hence, we are not just machines, there is something beyond that.

I'll start with the bare bones of Gödel's incompleteness. Let **M** be a machine which is programmed to print finite sequences of three symbols: -, P, D. (In what follows these sequences will be simply called sequences.) At each stage one sequence is printed into a square and each square is part of the tape unending in one direction.

We say that **M** prints a sequence if **M** prints it at some stage. We say that **M** does not print a sequence if **M** does not print it at any stage. Some of the sequences are meaningfull and we call them sentences. Here is the definition.

A sentence is a sequence of the form PX, -PX, PDY or -PDY, where X is any sequence not starting with D and Y is any sequence.

*(i) The meaning of a sentence of the form PX is "**M** prints X".*

*(ii) The meaning of a sentence of the form -PX is "**M** does not print X".*

*(iii) The meaning of a sentence of the form PDY is "**M** prints YY".*

*(iv) The meaning of a sentence of the form -PDY is "**M** does not print YY".*

Remark: It helps to think of -, P and D as words meaning "not", "prints" and "double". Sentences have meanings, so they are true or false. All other sequences are neither true nor false, they are meaningless. We deal with the machines that print only sentences. Our main interest concerning machines are their **correctness** and **completeness**. These notions are now easily defined.

*Machine M is correct if it prints only true sentences. (It is not necessary that **M** prints all of them).*

*Machine M is complete if it prints all true sentences. (It is possible that **M** prints some false sentences.)*

The most interesting machines would be those which are correct and complete. Unfortunately, there is no such machine (i.e. it is impossible to construct such a machine).

Theorem on correctness and completeness impossibility:
Every machine is either incorrect or incomplete.
Proof:
Take a look at sentence -PD-PD. It means "**M** does not print -PD-PD".
Of course, **M** either prints -PD-PD or not.
If **M** prints -PD-PD then it prints a false sentence i.e. **M** is not correct.
If **M** does not print -PD-PD then it does not print a true sentence i.e.**M** is incomplete.

Now, what is the link between this very simple theorem, Gödel's incompleteness theorem and mechanisation of our mathematical intuitions? To mechanize our mathematical intuitions means to construct a machine which would prove (and then print!) all mathematical theorems which are normally derived using these intuitions. A formalized mathematical theory with explicitly defined language, axioms and deductive rules is such a machine. Hence, here is the machine to which we may apply our simple theorem.

But there is one serious problem. Mathematical machines are not self reflective in the sense that the machines from our theorem are. These machines produce sentences which assert something about the machines themselves. Our mathematical theories (machines) assert many things about various mathematical objects, but nothing about the theories (machines) themselves. If we are interested in a theory (machine) itself we usually construct another theory (machine), called meta-theory (meta-machine), to deal with it. But here comes Gödel. In his famous [1] he proved that a mathematical theory (machine), which includes an appropriate amount of arithmetic, may represent its own meta-theory (meta-machine) so that our simple theorem applies. (This is the hardest part of Gödel's proof.) Hence, we have:

If mechanized mathematical theory includes an appropriate amount of arithmetic it is either incorrect or incomplete, i.e. if it is correct then it is incomplete. Even more, we can explicitly define Gödel's sentence (corresponding to our -PD-PD) which is true but not provable in the theory.

Now, the dualists argument is as follows. Any attempt to mechanize our mathematical intuitions is doomed to fail because the very fact of mechanization yields new intuitive knowledge, e.g. Gödel's sentence, which is not captured by the mechanization.

What is wrong with this argument? The problem is that you have to now quite a lot about the specific mechanization to conclude that its corresponding Gödel's sentence is true. In the specific case that Gödel analysed and that we partially presented above we know just enough to conclude that. On the other hand, it remains possible that there may exist (and even be empirically discoverable) a mathematical machine which in fact is equivalent to our mathematical intuitions. For example, we could be such machines.

So, dualists like Lucas and Penrose confused the incorrect argument:

There is no machine which could capture all our mathematical intuitions,

with the correct argument:

There is no machine which could capture all our mathematical intuitions and which we could understand well enough to see that its Gödel's sentence is true.

We may conclude. As far as Gödel's incompleteness theorem is concerned we could well be machines. But if we are then we are definitely not capable of the complete knowledge of the machines, i.e. of the complete knowledge of ourselves.

References

[1] Gödel, K. 1931, Über formal unentscheidbäre Sätze I. Monatshefte für Mathematik und Physik, 38, 173-198.

[2] Lucas, J.R. 1961, Minds, machines and Gödel. Philosophy, 36, 112-137.

[3] Penrose, R. 1989, The Emperor's New Mind. Oxford University

Interpretability Logic

Mladen Vuković, Department of Mathematics University of Zagreb, Croatia

This is an overview a study of interpretability logic in Zagreb for the last twenty years: a brief history and some planes for further research. The idea of treating a provability predicate as a modal operator goes back to Gödel. The same idea was taken up later by Kripke and Montague, but only in the mid–seventies was the correct choice of axioms, based on Löb's theorem, seriously considered by several logicians independently: G. Boolos, D. de Jongh, R. Magari, G. Sambin and R. Solovay. The system **GL** (Gödel, Löb) is a modal propositional logic. R. Solovay 1976. proved arithmetical completeness of modal system **GL**. Many theories have the same provability logic - **GL**. It means that the provability logic **GL** cannot distinguish some properties, as e.g. finite axiomatizability, reflexivity, etc. Some logicians considered modal representations of other arithmetical properties, for example interpretability, Π_n-conservativity, interpolability ... Roughly, a theory S interprets a theory T if there is a natural way of translating the language of S into the language of T in such a way that the translations of all the axioms of T become provable in S. We write $S \geq T$ if this is the case. A derived notion is that of relative interpretability over a base theory T. Let A and B be arithmetical sentences. We say that A interprets B over T if $T + A \geq T + B$.

Modal logics for relative interpretability were first studied by P. Hájek (1981) and V. Švejdar (1983). A. Visser (1990) introduced the binary modal logic **IL** (interpretability logic). The interpretability logic **IL** results from the provability logic **GL**, by adding the binary modal operator $\rhd$. The language of the interpretability logic contains propositional letters $p_0,\ p_1, \ldots,$ the logical connectives $\wedge,\ \vee, \rightarrow$ and $\neg$, and the unary modal operator $\Box$ and the binary modal operator $\rhd$. The axioms of the interpretability logic **IL** are: all tautologies of the propositional calculus, $\Box(A \rightarrow B) \rightarrow (\Box A \rightarrow \Box B),\ \Box A \rightarrow \Box\Box A,\ \Box(\Box A \rightarrow A) \rightarrow \Box A,\ \Box(A \rightarrow B) \rightarrow (A \rhd B),\ (A \rhd B\ \wedge B \rhd C) \rightarrow (A \rhd C),\ ((A \rhd C) \wedge (B \rhd C)) \rightarrow ((A \vee B) \rhd C),$ $(A \rhd B) \rightarrow (\Diamond A \rightarrow \Diamond B)$, and $\Diamond A \rhd A$, where $\Diamond$ stands for $\neg\Box\neg$ and $\rhd$ has the same priority as $\rightarrow$. The deduction rules of **IL** are modus ponens and necessitation. Arithmetical semantics of interpretability logic is based on the fact that each sufficiently strong theory S has arithmetical formulas $Pr(x)$ and $Int(x, y)$. Formula $Pr(x)$ expressing that "x is provable in S" (i.e. formula with Gödel number x is provable in S). Formula $Int(x, y)$ expressing that "$S + x$ interprets $S + y$." An arithmetical interpretation is a function $*$ from modal formulas into arithmetical sentences preserving Boolean connectives and satisfying $(\Box A)^* = Pr(\lceil A^* \rceil)$ and $(A \rhd B)^* = Int(\lceil A^* \rceil, \lceil B^* \rceil)$ ($\lceil A^* \rceil$ denote Gödel number of formula A^*). The system **IL** is natural from the modal point of view, but arithmetically incomplete. Various extensions of **IL**are obtained by adding some new axioms. These new axioms are called the principles of interpretability. We denote by **ILX** the system obtained by adding a principle X to the system **IL**. System **ILM** is the interpretability logic of Peano arithmetic. The arithmetical completeness of system **ILM** is proved in [1]. Visser (in [8]) proved the arithmetical completeness of the system **ILP**.

There are several kinds of semantics for the interpretability logic. The basic semantics is Veltman models. D. de Jongh and F. Veltman proved the completeness of **IL** w.r.t. Veltman models (see [5]). We think that there are two main reasons for other semantics. First, the proofs of arithmetical completeness of interpretability logic are very complex. Second, the characteristic classes Veltman frames of some principles of interpretability are equal. Generalized Veltman models were defined by de Jongh. We use generalized Veltman models in [12] to prove independence between principles of interpretability. A question is which kind of connection exists between generalized Veltman models and general Kripke models.

If we want to study a correspondence between Kripke models K and K' we consider an isomorphism or an elementarily equivalence. If we want to study "weaker" correspondence we can consider a bisimulation. Van Benthem defined bisimulations of Kripke models. Visser in [8] defined a notion of bismulation between two Veltman models. We defined a notion of bisimulation between two generalized Veltman models in [13], and proved Hennessy–Milner theorem for generalized Veltman semantics. We study various kinds of bisimulations of Veltman

models in [11]. In [10] bisimulation quotients of generalized Veltman models are considered. We proved in [14] that there is a bisimulation between Veltman model and generalized Veltman model. The existence of a bisimulation in general setting is an open problem.

P. Hájek and V. Švejdar in 1990. determined normal forms for the system **ILF**. The existence of the normal forms for system **IL** is an open problem. In [3] are determined normal forms in **IL** for some special classes of formulas.

The correspondence theory is the systematic study of the relationship between modal and classical logic. Bisimulations and the standard translation are two of the tools we need to understand modal expressivity. Van Benthem's characterization theorem (cf. [7]) shows that modal languages are the bisimulation invariant fragment of first–order languages, and it is established by classical methods of first-order model theory. The preservation theorems (cf. [6]) characterise a correspondence between semantic conditions of a class of models and logical formulas, too. However, the preservation property is usually much less significant than the corresponding expressive completeness property that any formula satisfying the semantic invariance condition is equivalent to one of the restricted syntactic form. D. Janin and I. Walukiewicz prove that a formula of monadic second–order logic is invariant under bisimulations if, and only if, it is logically equivalent to a formula of the μ–calculus. E. Rosen prove that the characterization theorem holds even in restriction to finite structures. A. Dawar and M. Otto in [4] investigate ramifications of van Benthem's characterization theorem for specific classes of Kripke structures. They study in particular Kripke modal classes defined through conditions on the underlying frames. Classical model theoretic arguments as saturated models and ultrafilter extensions do not apply to many of the most interesting classes. In the proofs the game–based analysis is used. V. Čačić and D. Vrgoč defined in [2] a bisimulation game between Veltman models and they proved the basic properties. We are interested in corresponding characterizations of modal fragments of first–order formula over Veltman models. The main problem when we prove van Benthem's theorem for interpretability logic is the existence of saturated Veltman model. We considered ultraproduct of Veltman models in [15].

References

[1] A. BERARDUCCI, *The Interpretability Logic of Peano Arithmetic*, Journal of Symbolic Logic, 55(1990), 1059-1089

[2] V. ČAČIĆ, D. VRGOČ, *A Note on Bisimulation and Modal Equivalence in Provability Logic and Interpretability Logic*, Studia Logica 101(2013), 31–44

[3] V. ČAČIĆ, M. VUKOVIĆ, *A note on normal forms for closed fragment of system IL*, Mathematical Communications, 17(2012), 195–204

[4] A. DAWAR, M. OTTO, *Modal characterization theorems over special classes of frames*, Annals of Pure and Applied Logic 161(2009), 1–42

[5] D. DE JONGH, F. VELTMAN, *Provability Logics for Relative Interpretability*, In: *Mathematical Logic*, (P. P. Petkov, Ed.), Proceedings of the 1988 Heyting Conference, Plenum Press, New York, 1990, 31–42

[6] T. PERKOV, M. VUKOVIĆ, *Some characterization and preservation theorems in modal logic*, Annals of Pure and Applied Logic 163(2012), 1928–1939

[7] J. VAN BENTHEM, *Modal Logic and Classical Logic*, Bibliopolis, Napoli, 1983.

[8] A. VISSER, Interpretability logic, In: P. P. Petkov (ed.), Mathematical Logic, Proceedings of the 1988 Heyting Conference, Plenum Press, New York, 1990, 175–210

[9] A. VISSER, *An overview of interpretability logic*, In: K. Marcus (ed.) et al., Advances in modal logic. Vol. 1. Selected papers from the 1st international workshop (AiML'96), Berlin, Germany, October 1996, Stanford, CA: CSLI Publications, CSLI Lect. Notes. 87(1998), 307–359

[10] D. VRGOČ, M. VUKOVIĆ, *Bisimulations and bisimulation quotients of generalized Veltman models*, Logic Journal of the IGPL, 18(2010), 870–880

[11] D. VRGOČ, M. VUKOVIĆ, *Bismulation quotients of Veltman models*, Reports on Mathematical Logic, 46(2011), 59–73

[12] M. VUKOVIĆ, *The principles of interpretability*, Notre Dame Journal of Formal Logic, 40(1999), 227–235

[13] M. VUKOVIĆ, *Hennessy–Milner theorem for interpretability logic*, Bulletin of the Section of Logic, 34(2005), 195–201

[14] M. VUKOVIĆ, *Bisimulations between generalized Veltman modles and Veltman models*, Mathematical Logic Quarterly, 54(2008), 368–373

[15] M. VUKOVIĆ, *A note on ultraproducts of Veltman models*, Glasnik matematički, 46(2011), 7–10

The unessential in classical logic and computation

Dragiša Žunić, Faculty of Economics and Engineering Management, Fimek, Novi Sad, Serbia
Pierre Lescanne, Ecole Normale Supérieure de Lyon, France

Abstract. We present a congruence relation on classical proofs, which identifies proofs up to trivial rule permutation. The study is performed in the framework of ${}^*\mathcal{X}$ calculus, designed to provide a Curry-Howard correspondence for classical logic, therefore the terms can be seen as proofs. Each congruence class has a single diagrammatic representation.

Introduction

We first present a higher order rewrite system, the ${}^*\mathcal{X}$ calculus, which represents a computational interpretation of standard Gentzen's formulation for classical logic (the sequent system G1 [1]). This system is characterized by the presence of structural rules, namely weakening and contraction. In this calculus, which encompasses all the details of classical computation, we define which syntactically different terms are in essence the same.

The history of computational interpretations of classical logic is recent. The first one relying on sequents was presented by Herbelin [2], while a more direct correspondence with a standard sequent formulation of classical logic was presented in [3]. This research first lead to the $\mathcal{X}$ calculus [4] which served as a base to implement explicit erasure and duplication, yielding the ${}^*\mathcal{X}$ calculus [5].

${}^*\mathcal{X}$ calculus, the syntax

Intuitively when we speak about ${}^*\mathcal{X}$-terms we speak about classical proofs in sequent system with explicit structural rules. Terms are built from *names*. This concept differs from that applied in λ-calculus, where *variable* is the basic notion. The difference lies in the fact that a variable can be substituted by an arbitrary term, while a name can be only *renamed* (that is, substituted by another name). The presences of hats over certain names denotes the binding of a name.

Definition 1 (${}^*\mathcal{X}$-syntax) *The syntax of ${}^*\mathcal{X}$-calculus is presented in Figure 1, where $x, y, z...$ range over an infinite set of in-names and $\alpha, \beta, \gamma...$ range over an infinite set of out-names.*

P, Q	$::=$	$\langle x.\alpha \rangle$	*capsule*	(axiom rule)
	$\mid$	$\widehat{x}\, P\, \widehat{\beta} \cdot \alpha$	*exporter*	(right arrow-introduction)
	$\mid$	$P\, \widehat{\alpha}\, [x]\, \widehat{y}\, Q$	*importer*	(left arrow-introduction)
	$\mid$	$P\widehat{\alpha} \dagger \widehat{x}Q$	*cut*	(cut)
	$\mid$	$x \odot P$	*left-eraser*	(left weakening)
	$\mid$	$P \odot \alpha$	*right-eraser*	(right weakening)
	$\mid$	$z{<}^{\widehat{x}}_{\widehat{y}}\langle P]$	*left-duplicator*	(left contraction)
	$\mid$	$[P\rangle^{\widehat{\alpha}}_{\widehat{\beta}}{>}\gamma$	*right-duplicator*	(right contraction)

Figure 1: The syntax of ${}^*\mathcal{X}$

Assigning types to terms

The type system presents the way constructors are linked with logic, i.e., with proofs. Expressions of the form $P \vdots \ \Gamma \vdash \Delta$ represent the *type assignment* [3].

Definition 2 (Typable terms) *We say that a term P is* typable *if there exist contexts Γ and Δ such that $P \vdots \ \Gamma \vdash \Delta$ holds in the system of inference rules given by Figure 2.*

$$\frac{}{\langle x.\alpha \rangle \vdots \ x : A \vdash \alpha : A} \textit{(ax)}$$

$$\frac{P \vdots \ \Gamma \vdash \alpha : A, \Delta \qquad Q \vdots \ \Gamma', y : B \vdash \Delta'}{P\,\widehat{\alpha}\,[x]\,\widehat{y}\,Q \vdots \ \Gamma, \Gamma', x : A \to B \vdash \Delta, \Delta'} (\to L) \qquad \frac{P \vdots \ \Gamma, x : A \vdash \alpha : B, \Delta}{\widehat{x}\,P\,\widehat{\alpha} \cdot \beta \vdots \ \Gamma \vdash \beta : A \to B, \Delta} (\to R)$$

$$\frac{P \vdots \ \Gamma \vdash \alpha : A, \Delta \qquad Q \vdots \ \Gamma', x : A \vdash \Delta'}{P\widehat{\alpha} \dagger \widehat{x}Q \vdots \ \Gamma, \Gamma' \vdash \Delta, \Delta'} \textit{(cut)}$$

$$\frac{P \vdots \ \Gamma \vdash \Delta}{x \odot P \vdots \ \Gamma, x : A \vdash \Delta} \textit{(weak-L)} \qquad \frac{P \vdots \ \Gamma \vdash \Delta}{P \odot \alpha \vdots \ \Gamma \vdash \alpha : A, \Delta} \textit{(weak-R)}$$

$$\frac{P \vdots \ \Gamma, x : A, y : A \vdash \Delta}{z < \frac{\widehat{x}}{\widehat{y}} \langle P] \vdots \ \Gamma, z : A \vdash \Delta} \textit{(cont-L)} \qquad \frac{P \vdots \ \Gamma \vdash \alpha : A, \beta : A, \Delta}{[P\rangle \frac{\widehat{\alpha}}{\widehat{\beta}} > \gamma \vdots \ \Gamma \vdash \gamma : A, \Delta} \textit{(cont-R)}$$

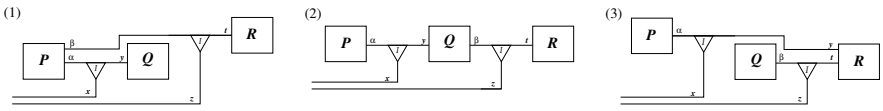
Figure 2: The type assignent for implicative fragment

Reduction rules are numerous and capture the richness and complexity of classical cut elimination, but we will not be dealing with the dynamic of the system here (see [4, 5] for details).

The congruence relation

We introduce the congruence relation on terms, denoted $\equiv$, represented by a list of equations (the list is very partial due to limited space). For the complete list see [5]. Congruence relation induced is reflexive, symmetric and transitive relation closed under any context. The motivation for introducing it into the system is to come closer to the essence of classical proofs, and abstract away from unessential in classical proofs. Every rule is associated with one corresponding diagram. A name is assigned to every congruence rule, and thus they are presented in the form: *name* : $P \equiv Q$.

importer-importer



[3]Technically we assign contexts, which are sets of pairs (name, formula), to terms. If we forget about labels and consider only types, we are going back to Gentzen's classical system $G1$ (see Figure 2), where contexts are multisets of formulas.

$ii1$: $(P\,\widehat{\alpha}\,[x]\,\widehat{y}\,Q)\,\widehat{\beta}\,[z]\,\widehat{t}\,R \equiv (P\,\widehat{\beta}\,[z]\,\widehat{t}\,R)\,\widehat{\alpha}\,[x]\,\widehat{y}\,Q$ with $\alpha, \beta \in N(P)$

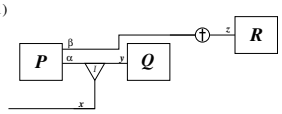
$ii2$: $(P\,\widehat{\alpha}\,[x]\,\widehat{y}\,Q)\,\widehat{\beta}\,[z]\,\widehat{t}\,R \equiv P\,\widehat{\alpha}\,[x]\,\widehat{y}\,(Q\,\widehat{\beta}\,[z]\,\widehat{t}\,R)$ with $y, \beta \in N(Q)$

$ii3$: $(Q\,\widehat{\beta}\,[z]\,\widehat{t}\,R) \equiv Q\,\widehat{\beta}\,[z]\,\widehat{t}\,(P\,\widehat{\alpha}\,[x]\,\widehat{y}\,R)$ with $y, t \in N(R)$

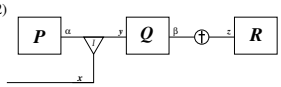
---

cut-importer

(1)



(2)

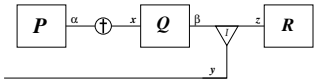


$ci1$: $(P\,\widehat{\alpha}\,[x]\,\widehat{y}\,Q)\widehat{\beta}\,\dagger\,\widehat{z}R \equiv (P\widehat{\beta}\,\dagger\,\widehat{z}R)\,\widehat{\alpha}\,[x]\,\widehat{y}\,Q$ with $\alpha, \beta \in N(P)$

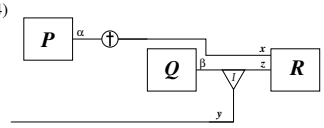
$ci2$: $(P\,\widehat{\alpha}\,[x]\,\widehat{y}\,Q)\widehat{\beta}\,\dagger\,\widehat{z}R \equiv P\,\widehat{\alpha}\,[x]\,\widehat{y}\,(Q\widehat{\beta}\,\dagger\,\widehat{z}R)$ with $y, \beta \in N(Q)$

---

(3)



(4)



$ci3$: $P\widehat{\alpha}\,\dagger\,\widehat{x}(Q\,\widehat{\beta}\,[y]\,\widehat{z}\,R) \equiv (P\widehat{\alpha}\,\dagger\,\widehat{x}Q)\,\widehat{\beta}\,[y]\,\widehat{z}\,R$ with $x, \beta \in N(Q)$

$ci4$: $P\widehat{\alpha}\,\dagger\,\widehat{x}(Q\,\widehat{\beta}\,[y]\,\widehat{z}\,R) \equiv Q\,\widehat{\beta}\,[y]\,\widehat{z}\,(P\widehat{\alpha}\,\dagger\,\widehat{x}R)$ with $x, z \in N(R)$

The relation $\equiv$ induces congruence classes on terms. It has been argued in [6] that two sequent proofs induce the same proof net if and only if one can be obtained from the other by a sequence of transpositions of independent rules. At this static level we have proceeded further as we have *explicitly* shown by congruence rules, how exactly this transposition is done.

Basic properties of $\equiv$, and terms as diagrams. The congruence relation enjoys important properties. Since it describes the way to perform restructuring of terms, it is important to have preservation of the set of free names. Then, the property of type preservation ensures that term-restructuring defined by $\equiv$ can be seen as proof-transformation.

The reader could already see the diagrams as intuitive illustration of congruence rules. It is possible to define a translation (call it $\mathcal{D}$) from terms to diagrams, inductively on the structure of terms, but due to a lack of space we don't present it here. Based on that definition, ideally we hope to prove that each congruence class (with many terms) has a single diagrammatic representation. Moreover, in the framework of future work and the dynamics of the system, to show that a single reduction step corresponds to a diagrammatic reduction step.

Conclusion

By explicitly stating which syntactically different terms should be considered the same, we unveil the unessential part of sequent classical proofs, sometimes referred to as the *syntactic bureaucracy*. This is done in the framework of ${}^*\mathcal{X}$. It is illustrated that such computational model comes close to diagrammatic computation, as the concept of reducing modulo corresponds to diagrammatic reductions which focuses on the essence of classical proofs and drastically reduces the number of reduction steps.

References

[1] A. S. Troelstra and H. Schwichtenberg, *Basic Proof Theory*. New York, NY, USA: Cambridge University Press, 1996.

[2] P.-L. Curien and H. Herbelin, "The duality of computation," in *Proc. 5 th ACM SIGPLAN Int. Conf. on Functional Programming (ICFP'00)*, pp. 233–243, ACM, 2000.

[3] C. Urban and G. M. Bierman, "Strong normalisation of cut-elimination in classical logic," *Fundam. Inf.*, vol. 45, no. 1,2, pp. 123–155, 2001.

[4] S. van Bakel, S. Lengrand, and P. Lescanne, "The language $\mathcal{X}$: circuits, computations and classical logic," in *Proc.9th Italian Conf. on Theoretical Computer Science (ICTCS'05)*, vol. 3701 of *Lecture Notes in Computer Science*, pp. 81–96, 2005.

[5] D. Žunić, *Computing With Sequent and Diagrams in Classical Logic - Calculi ${}^{*}\mathcal{X}$, ${}^{©}\mathcal{X}$ and ${}^{d}\mathcal{X}$*. PhD thesis, Ecole Normale Supériéure de Lyon, France, 2007.

[6] E. Robinson, "Proof nets for classical logic," *Journal of Logic and Computation*, vol. 13, no. 5, pp. 777–797, 2003.