Simulation and Synthesis of Deduction Calculi

Renate A. Schmidt[1]

School of Computer Science
The University of Manchester
Manchester, UK

Abstract

This paper gives an overview of two methods for automatically or semi-automatically generating deduction calculi from the semantic specification of a logic. One approach is based on simulating deduction approaches with techniques of automated reasoning and first-order resolution. The second approach synthesises sound, complete and terminating tableau calculi directly from the semantic specification of a logic.

1 Introduction

Non-classical logics, including modal logics, description logic, hybrid logics and intuitionistic logic, are popular and have many applications. The applications range from multi-agent systems to medical domains, web services and the semantic web. Demand for deduction calculi and implemented provers for non-classical logics is thus high. Rather than developing deduction calculi one by one and proving important properties such as soundness, completeness and decidability for each individually, we are interested in the possibility of providing a generic framework for developing deduction calculi. The idea is that deduction calculi can be developed for all these different logics and their applications in a uniform way. There are various interactive prover engineering platform that can be used for developing calculi [1,7]. These provide flexible and general frameworks for defining and experimenting with implementations of different sets of tableau inference rules for different logics, different rule application strategies and different optimisations.

[1] Email: Renate.Schmidt@manchester.ac.uk

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.com/locate/entcs

As an alternative, we are interested in fully-automatically generating deduction calculi from the specification of a logic. The logic of interest is assumed to be defined by a high-level specification of its formal semantics. The aim is to turn this into a set of inference rules, which forms a sound and complete deduction calculus for the logic. Ideally we also want to be able to guarantee termination if the logic is decidable. Automated synthesis of calculi is a very challenging problem and in general it is of course not possible to turn every specification of a logic into a sound, complete and terminating deduction calculus. It is however possible to solve the problem for a large number of logics.

In this paper we discuss two methods for synthesising deduction calculi: the develop via first-order resolution method [22] and the direct tableau synthesis framework [26,27]. The focus of both methods is on developing ground semantic calculi for modal-type logics that operate on labelled modal formulae where the labels represent states in the underlying Kripke models.

2 The develop via first-order resolution method

Although resolution calculi apparently operate considerably differently from tableau calculi, it is possible to linearly simulate many forms of modal or description logic tableau calculi with standard techniques of first-order resolution theorem proving [6,13,14,16]. The simulations are in fact so close that they can be exploited to read off tableau rules from the clausal forms of the translations used. This is explored in [6,24] and has been turned into the *develop via first-order resolution approach* to devising deduction methods in [22].

The three main ingredients of the develop via first-order resolution approach are the following:

(i) An effective, satisfiability-equivalence preserving translation to first-order logic that retains enough information about the input formula of the given logic so that back-translation of resolution derivations to the original logic is possible.

(ii) A refinement of first-order resolution that performs inferences exactly like the kind of system we want to develop.

(iii) If needed, partial pre-saturation and purification of the clausal form.

If these ingredients are defined in the right way then the actual inference rules can be read off from the clausal form.

Transformations satisfying ingredient (i) are not hard to find. They can be based on encoding the semantics of the logic in first-order logic and structural transformation, a standard technique in automated reasoning, which introduces new predicate symbols and definitions for subformulae.

For ingredient (ii), refinements of first-order resolution suitable for the

simulation of ground semantic deduction calculi are variations of hyperresolution. Additionally it is important that clauses are in range-restricted form. A clause is range-restricted if all variables of the clause occur in the negative literals of that clause. For the mainstream modal and description logics the clauses are automatically range-restricted. For more expressive logics range-restriction can be ensured with generic range-restricting transformations as defined in [5].

Ingredient (iii), which is not always required, is not discussed here. An example of its use can be found in [22].

The main insight allowing inference rules to be extracted from the clausal form is the following. Hyperresolution on range-restricted clauses has the property that all positive parent clauses are ground clauses and all conclusions are ground clauses. This allows the negative, non-ground parent clause of a hyperresolution inference step to be interpreted as an inference rule $\mathcal{I}$ of the ground calculus. The positive, ground parent clauses of a hyperresolution inference step represent the premises of the rule $\mathcal{I}$, and the conclusions represent the conclusions of the rule. More details can be found in [6,22,24].

The form of calculi one obtains using the develop via first-order resolution approach depends very much on the precise definition of all three ingredients. It turns out that small modifications result in different variations of calculi and also different styles of calculi. To obtain calculi with branching rules splitting is used. For instance, hyperresolution with splitting and the relational translation mapping yields ground semantic tableau calculi in the spirit of Kripke. By this we mean tableau calculi based on structural rules for frame correspondence properties. In [6,22] we have shown how hyperresolution with splitting and the relational translation mapping give us tableau calculi from for the dynamic modal logics $K_{(m)}(\wedge, \vee, ^\smile)$ and $K_{(m)}(\wedge, \vee, ^\smile, \uparrow)$. Using the axiomatic translation mapping and hyperresolution with splitting it is possible to derive tableau calculi based on propagation rules rather than structural rules. This is illustrated in [24] for extensions of the basic modal logic K.

When using the relational translation and *dual* hyperresolution with splitting dual tableau calculi, or Rasiowa-Sikorski proof calculi [19], are obtained [22]. If splitting is omitted, that is, we use ordinary hyperresolution together with the relational translation then (labelled) modal resolution calculi similar to the ones introduced in [3] are obtained [22].

It is also possible to derive in a systematic way sound and complete calculi via other translation methods. If we use the functional translation (e.g. [17]) or the optimised functional translation (e.g. [18]) in combination with hyperresolution and splitting then prefix tableau calculi are produced. With the axiomatic translation we can generate prefix modal tableau calculi based on propagation rules. Without splitting, modal resolution calculi in which the labels are prefixes are generated. Dualising everything (including the transfor-

mations and the hyperresolution calculus) yields Rasiowa-Sikorski type calculi with prefixed formulae.

The approach allows techniques from resolution theorem proving to be carried over to modal deduction approaches. For example, using ordered hyperresolution produces deduction calculi with ordering restrictions [22]. In tableau calculi the ordering can be used to determine the literal to branch on when a splitting rule is applied. In modal resolution calculi the ordering restricts the application of the resolution rule and thus reduces the search space and improves the performance. Similarly for dual systems.

Another example is absorption. Absorption is a technique for avoiding unnecessary branching on formulae from the background theory (if there is one), which was introduced for description logics in [11]. It turns out that absorption can be naturally explained, proved correct, and generalised via simulation with hyperresolution [13].

In first-order resolution an important notion is redundancy elimination. This notion carries over to other calculi leading to three forms of redundancy: redundant formulae, redundant rule applications and notably redundant inference rules [22]. This means that reasoning modulo redundancy can be formalised also for modal deduction calculi leading to stronger results and better systems.

Soundness and completeness of the simulating resolution refinement carries over immediately to the extracted deduction calculus provided the extracted approach corresponds in a linear, step-wise fashion with the simulating approach. Decidability of the generated approach is automatic when the simulating resolution refinement is a decision procedure for the translation of the logic. Finding a suitable combination of a translation, resolution refinement and pre-saturation (if needed) is the easier part. More challenging is to develop ways to guarantee termination and prove decidability—should suitable decidability results not already be known. Ordered resolution can be used to decide numerous fragments of first-order logic into which (dynamic) modal logics can be embedded, for instance, the guarded fragments, the two-variable fragment, fluted logic and Maslov's class K, cf. [8]. For hyperresolution far fewer decidability results are currently available than for ordered resolution. Ways of using hyperresolution to decide modal logics, description logics and related first-order fragments are described in e.g. [6,9,10,14].

One of the attractions of the develop via first-order resolution method is the possibility to use existing first-order resolution theorem provers as implementations of the generated deduction calculi. Hyperresolution, or essentially equivalent refinements, are standardly implemented in well-known first-order resolution theorem provers. Ordering restrictions can be flexibly defined in these and splitting is currently available in at least (m)spass [15,29] and vampire [20]. With modest implementation effort it is therefore possible to

use these provers as modal tableau provers, modal Rasiowa-Sikorski provers, or modal resolution provers. All that is necessary, is to implement the appropriate translations and then choose the correct combination of flag settings so that the prover uses the simulating refinement. All these basic ingredients are already implemented in (m)spass. An adaptation of spass, which translates resolution proofs back into modal tableau proofs and first-order models into modal models has been been developed by AlBarakati [2]. This provides a new and slightly unusual implementation of a modal theorem prover. Currently it caters for the dynamic modal logic $K_{(m)}(\wedge, \vee, ^{\smile}, \restriction)$ and extensions with first-order frame correspondence properties.

The approach provides a common framework for the direct comparison of different deduction approaches. Tableau and resolution methods are typically regarded as quiet opposite. Interpreted within the framework the difference is actually small [22]: Modal resolution can be viewed as semantic tableau without splitting and semantic tableau can be viewed as modal resolution with splitting.

The approach also provides a uniform framework for the empirical comparison of different deduction approaches. Examples of empirical studies undertaken with (m)spass following essentially this approach can be found in [5,16]. Such experiments allow more meaningful comparisons of different calculi, or different styles of deduction, than experiments based on the comparisons of independently implemented provers [12,16].

We know that the approach can be generalised and applied to more expressive logics to yield sound and complete modal deduction calculi for many, if not all, first-order definable (dynamic) modal logics and the corresponding description logics. The approach applies also to other first-order definable logics and fragments of first-order logic. For example, linear correspondence results between tableau systems and resolution have been obtained in [10] for decidable first-order fragments closely related to the guarded fragment. This allows us to immediately pull out sound, complete and terminating tableau procedures for these fragments. Similarly, tableau decision procedure can be defined for the solvable class $\mathcal{BU}$ introduced and studied in [9].

3 The direct tableau synthesis framework

In this section we describe the *(direct) tableau synthesis framework* introduced in [27]. It provides an alternative approach to generating sound, complete and terminating tableau calculi.

The tableau synthesis framework consists of two stages:

(i) Automated synthesis of a tableau calculus.

(ii) Addition of a blocking mechanism to ensure termination.

The first stage takes a high-level specification of the formal semantics of a logic and transforms it into a set of tableau inference rules. In particular, the user defines the formal semantics of the given logic in a many-sorted first-order language. Then the method automatically reduces the semantic specification of the logic to Skolemised implicational forms, which are then rewritten as tableau inference rules. These are combined with default closure and equality rules. A set of rules obtained in this way provides a sound and constructively complete calculus when certain well-definedness conditions are true for the specification [27].

A standard way of turning ground semantic tableau methods into decision procedures is to add blocking. Various different blocking mechanisms have been developed for different modal and description logics. In our framework we use a form of blocking, called unrestricted blocking [25]. Unrestricted blocking provides a very general and powerful mechanism to guarantee termination. Unrestricted blocking is based on a cut rule and ground equality reasoning. The idea is that two terms (labels) on a branch are identified; if this leads to a contradiction backtracking is performed and a lemma is added, which says that the two terms are different. To control the application of δ-rules, δ-rules may only be applied to formulae with minimal terms in any equivalence class. δ-rules are any rules triggering the creation of new terms, for example, $\Diamond$-rules. A second important restriction is that at some point in a branch blocking has been applied exhaustively before the application of any δ-rule.

For logics with the effective finite model property the unrestricted blocking mechanism ensures termination provided the following conditions all hold [26].

(a) The effective finite model property can be shown by a filtration argument.

(b) The tableau calculus is sound and constructively complete.

(c) A weak form of subformula property holds for tableau derivations.

Constructive completeness is a slightly stronger notion than completeness. It means that for every open branch in a tableau there is a model, which reflects all the formulae occurring on the branch. The result provides a general methodology for turning sound and constructively complete tableau calculi for modal-type logics (not only generated ones) into terminating calculi [26].

The direct tableau synthesis framework generates tableau calculi satisfying the prerequisites (b) and (c). It turns out that provided the semantic specification of the logic is well-defined in a certain sense, the subformula property can be imposed on the generated calculi. Crucial is the separation of the syntax of the logic from the 'extras' in the meta-language needed for the semantic specification of the logic [27].

An important element of the method is rule refinement. The set of rules generated from a well-defined semantic specification is not immediately optimal, but can be transformed into improved and more elegant forms through

two refinements [27]. The first refinement reduces the number of branches of a rule by constraining a rule with additional premises rather than deriving new conclusions. The second refinement simplifies the language of the tableau calculus when there is enough expressivity in the language of the logic for representing the basics of the semantics of the logic.

For various logics, including intuitionistic logic [27] and basic modal logic $K_{(m)}$ with nominals (alternatively description logic $\mathcal{ALCO}$ or basic hybrid logic), the calculi produced are equivalent modulo inessential variations and a different kind of blocking to calculi commonly found in the literature. By design the method can generate tableau calculi for expressive description logics such as $\mathcal{ALBO}$ [25]. $\mathcal{ALBO}$ extends the description logic $\mathcal{ALC}$ with various role operators including role inverse, role union and role negation. It is equivalent to Boolean modal logic extended with the relational inverse operator and nominals, and has the same expressivity as the two-variable fragment of first-order logic. We have applied the method to develop a tableau-based algorithm for testing the admissibility of modal rules, for which we needed to devise a tableau calculus for a logic that has not been considered before [4]. We believe the direct tableau synthesis approach is also applicable to most known, first-order definable modal and description logics. Non first-order translatable logics such as propositional dynamic logic are currently beyond the scope of the method.

That the generated calculi are constructively complete has the advantage that models can be effectively generated from open, finished branches in tableau derivations. This means that the synthesised tableau calculi can be used for model building.

The direct tableau synthesis framework can be regarded as a mathematical formalisation and generalisation of tableau development methodologies. It essentially presents a general method for proving (constructive) completeness and termination of tableau calculi. The formalisation separates the creative part of tableau calculus development, which needs to be done by a human developer and the automatic part of the development process, which can be left to an automated tableau synthesiser and an automated theorem prover. The creative part is the specification of the logic and proving the effective finite model property based on filtration if this is not already known for the logic. The automatic part generates the tableau rules and verifies the conditions for well-definedness of the semantic specification and the conditions justifying the preservation of constructive completeness by rule refinement.

The MetTeL system [28] is an implementation of a generic tableau theorem prover incorporating the main features of the tableau synthesis framework. Though not a tableau calculus synthesiser it can handle different logics in a flexible way.

4 Concluding remarks

We have sketched two methods for synthesising deduction calculi for modal-type logics. Both approaches use the observation that tableau rules are essentially normal forms of the specification of the semantics of the logic. The develop via first-order resolution approach exploits the generality and versatility of first-order resolution to simulate different deduction approaches.

The direct tableau synthesis framework is currently limited to generating tableau calculi. It ensures that the generated tableau calculi are sound and constructively complete. This ensures that adding the unrestricted blocking mechanism produces a terminating tableau calculus, whenever the logic can be shown to admit finite filtration.

Though blocking has so far mainly been used in conjunction with tableau methods for non-classical logics, blocking can be used in a more general setting. We have defined and experimented with various forms of blocking as enhancements of bottom-up model generation methods for first-order logic [5]. Bottom-up model generation methods are closely related to hyperresolution and hypertableau methods. Using these methods in combination with the unrestricted blocking mechanism should allow us to devise new decision procedures for a wide range of logics and solvable first-order fragment. I expect that it is possible to do so in a systematic way.

All in all, a lot remains to be done, but I am confident that the ideas of both methods to synthesising deduction calculi can be taken a lot further.

Acknowledgements

I want to thank all my collaborators, including especially Ullrich Hustadt and Dmitry Tishkovsky. The work is supported by research grants EP/D056152/1, EP/F068530/1, EP/F014570/1 of the UK EPSRC.

References

[1] P. Abate and R. Goré. The Tableaux Work Bench. In M. C. Mayer and F. Pirri, editors, *Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX 2003)*, volume 2796 of *Lecture Notes in Computer Science*, pages 230–236. Springer, 2003.

[2] R. G. AlBarakati. Development of a tableaux resolution prover. Master's thesis, The University of Manchester, UK, 2009.

[3] C. Areces, M. de Rijke, and H. de Nivelle. Resolution in modal, description and hybrid logic. *Journal of Logic and Computation*, 11(5):717–736, 2001.

[4] S. Babenyshev, V. Rybakov, R. A. Schmidt, and D. Tishkovsky. A tableau method for checking rule admissibility in S4. *Electronic Notes in Theoretical Computer Science*. This volume.

[5] P. Baumgartner and R. A. Schmidt. Blocking and other enhancements for bottom-up model generation methods. In U. Furbach and N. Shankar, editors, *Automated Reasoning: Third International Joint Conference on Automated Reasoning (IJCAR 2006)*, volume 4130 of *Lecture Notes in Artificial Intelligence*, pages 125–139. Springer, 2006.

[6] H. De Nivelle, R. A. Schmidt, and U. Hustadt. Resolution-based methods for modal logics. *Logic Journal of the IGPL*, 8(3):265–292, May 2000.

[7] L. Fariñas del Cerro and O. Gasquet. A general framework for pattern-driven modal tableaux. *Logic Journal of the IGPL*, 10(1):51–83, 2002.

[8] C. Fermüller, A. Leitsch, U. Hustadt, and T. Tammet. Resolution decision procedures. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, pages 1791–1849. Elsevier, 2001.

[9] L. Georgieva, U. Hustadt, and R. A. Schmidt. A new clausal class decidable by hyperresolution. In A. Voronkov, editor, *Automated Deduction—CADE-18*, volume 2392 of *Lecture Notes in Artificial Intelligence*, pages 260–274. Springer, 2002.

[10] L. Georgieva, U. Hustadt, and R. A. Schmidt. Hyperresolution for guarded formulae. *Journal of Symbolic Computation*, 36(1–2):163–192, 2003.

[11] I. Horrocks. *Optimising Tableaux Decision Procedures for Description Logics*. PhD thesis, The University of Manchester, UK, 1997.

[12] U. Hustadt and R. A. Schmidt. An empirical analysis of modal theorem provers. *Journal of Applied Non-Classical Logics*, 9(4):479–522, 1999.

[13] U. Hustadt and R. A. Schmidt. On the relation of resolution and tableaux proof systems for description logics. In T. Dean, editor, *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 110–115. Morgan Kaufmann, 1999.

[14] U. Hustadt and R. A. Schmidt. Issues of decidability for description logics in the framework of resolution. In R. Caferra and G. Salzer, editors, *Automated Deduction in Classical and Non-Classical Logics*, volume 1761 of *Lecture Notes in Artificial Intelligence*, pages 191–205. Springer, 2000.

[15] U. Hustadt and R. A. Schmidt. MSPASS: Modal reasoning by translation and first-order resolution. In R. Dyckhoff, editor, *Automated Reasoning with Analytic Tableaux and Related Methods, International Conference (TABLEAUX 2000)*, volume 1847 of *Lecture Notes in Artificial Intelligence*, pages 67–71. Springer, 2000.

[16] U. Hustadt and R. A. Schmidt. Using resolution for testing modal satisfiability and building models. In I. P. Gent, H. van Maaren, and T. Walsh, editors, *SAT 2000: Highlights of Satisfiability Research in the Year 2000*, volume 63 of *Frontiers in Artificial Intelligence and Applications*, pages 459–483. IOS Press, Amsterdam, 2000.

[17] H. J. Ohlbach. Semantics based translation methods for modal logics. *Journal of Logic and Computation*, 1(5):691–746, 1991.

[18] H. J. Ohlbach and R. A. Schmidt. Functional translation and second-order frame properties of modal logics. *Journal of Logic and Computation*, 7(5):581–603, 1997.

[19] H. Rasiowa and R. Sikorski. *The Mathematics of Metamathematics*, volume 41 of *Monografie matematyczne*. Polish Scientific Publ., Warsaw, 1963.

[20] A. Riazanov and A. Voronkov. The design and implementation of VAMPIRE. *AI Communications*, 15(2-3):91–110, 2002.

[21] R. A. Schmidt. Developing modal tableaux and resolution methods via first-order resolution. In G. Governatori, I. Hodkinson, and Y. Venema, editors, *Advances in Modal Logic, Volume 6*, pages 1–26, London, 2006. College Publications.

[22] R. A. Schmidt. A new methodology for developing deduction methods. *Annals of Mathematics and Artificial Intelligence*, 55(1–2):155–187, 2009. Improved and extended version of [21].

[23] R. A. Schmidt and U. Hustadt. A principle for incorporating axioms into the first-order translation of modal formulae. In F. Baader, editor, *Automated Deduction—CADE-19*, volume 2741 of *Lecture Notes in Artificial Intelligence*, pages 412–426. Springer, 2003.

[24] R. A. Schmidt and U. Hustadt. The axiomatic translation principle for modal logic. *ACM Transactions on Computational Logic*, 8(4):1–55, 2007. The short version is [23].

[25] R. A. Schmidt and D. Tishkovsky. Using tableau to decide expressive description logics with role negation. In K. Aberer, K.-S. Choi, N. Fridman Noy, D. Allemang, K.-I. Lee, L. J. B. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber, and P. Cudré-Mauroux, editors, *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11–15, 2007*, volume 4825 of *Lecture Notes in Computer Science*, pages 438–451. Springer, 2007.

[26] R. A. Schmidt and D. Tishkovsky. A general tableau method for deciding description logics, modal logics and related first-order fragments. In A. Armando, P. Baumgartner, and G. Dowek, editors, *Automated Reasoning (IJCAR 2008)*, volume 5195 of *Lecture Notes in Computer Science*, pages 194–209. Springer, 2008.

[27] R. A. Schmidt and D. Tishkovsky. Automated synthesis of tableau calculi. In M. Giese and A. Waaler, editors, *Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX 2009)*, volume 5607 of *Lecture Notes in Computer Science*, pages 310–324. Springer, 2009.

[28] D. Tishkovsky. The METTEL system. http://www.cs.man.ac.uk/~dmitry/implementations/MetTeL/.

[29] C. Weidenbach, R. A. Schmidt, T. Hillenbrand, R. Rusev, and D. Topic. System description: SPASS version 3.0. In F. Pfenning, editor, *Automated Deduction—CADE-21*, volume 4603 of *Lecture Notes in Artificial Intelligence*, pages 514–520. Springer, 2007.