A New Refutation Calculus With Logical Optimizations for PLTL

Mauro Ferrari
DiSTA
Università degli Studi dell'Insubria
Email: mauro.ferrari@uninsubria.it

Camillo Fiorentini
DI
Università degli Studi di Milano
Email: fiorentini@di.unimi.it

Guido Fiorino
DISCo
Università degli Studi di Milano-Bicocca
Email: guido.fiorino@unimib.it

***Abstract*—Propositional Linear Temporal Logic (PLTL) is a tool for reasoning about systems whose states change in time. We present an ongoing work on a new proof-search procedure for Propositional Linear Temporal Logic and its implementation. The proof-search procedure is based on a one-pass tableau calculus with a multiple-conclusion rule treating temporal-operators and on some logical optimization rules. These rules have been devised by applying techniques developed by the authors for logics with Kripke semantics and here applied to the Kripke-based semantics for Propositional Linear Temporal Logic.**

Keywords–Propositional Linear Temporal Logic; Tableaux; Satisfiability checking.

I. INTRODUCTION

In recent years, while studying proof-search procedures for non classical logics, we have introduced new tableau calculi and logical optimization rules for propositional Intuitionistic Logic (INT) [1] and propositional Gödel-Dummett Logic (DUM) [2]. As an application of these results, we have implemented theorem provers for these logics [2][3] that outperform their competitors. The above quoted calculi and optimizations are the result of a deep analysis of the Kripke semantics of the logic at hand. In this paper, we show how such semantical analysis can also be fruitfully applied to other non-classical logics with a Kripke-based semantics, by analyzing the case of PLTL. In particular, we present a new refutation tableau calculus and logical optimizations for PLTL and we briefly discuss a prototype Prolog implementation of the resulting proof-search procedure.

As for related works, our tableau calculus for PLTL lies in the line of the one-pass calculi based on sequents and tableaux of [4][5][6], whose features are suitable for automated deduction. We also cite as related the approaches based on sequent calculi discussed in [7][8] and the natural deduction based proof-search technique discussed in [9]. The results in [10][11] are based on resolution, thus they are related less to our approach.

The paper is organized as follows. Section II provides the core of our logical characterization for PLTL, Section III describes some optimization rules based on replacement of formulas, Section IV gives an account of the performances of the Prolog prototype under development, finally Section V summarizes the work with a short discussion.

II. A NEW LOGICAL CHARACTERIZATION OF PLTL

We restrict ourselves to the temporal connectives Until $\mathcal{U}$ and Next $\circ$. Entering more in details, our first contribution is a new logical characterization of PLTL by means of a tableau calculus whose distinguished feature is the multiple-conclusion rule Lin, that is a rule whose number of conclusions depends on the number of $\mathcal{U}$-formulas in the premise. The rule Lin is inspired by the multiple-conclusion rules we have developed in [2][12] to logically characterize the logic DUM. As a matter of fact, PLTL and DUM are semantically characterized by Kripke models based on linearly ordered states (we recall that PLTL and DUM have different languages and a different interpretation of the connective $\rightarrow$).

We give an account of rule Lin by means of an example which also introduces the argument to prove its correctness. Figure 1 contains the version for the language restricted to the temporal connectives $\mathcal{U}$ and $\circ$. Let us suppose that at time t a PLTL model $\underline{K}$ satisfies the set of formulas $S = \{\circ A, \circ(B\mathcal{U}C), \circ(D\mathcal{U}E)\}$ (in formulas $t \Vdash S$), where the main connective of A is not $\mathcal{U}$. This implies that: (i) $t+1 \Vdash A$ holds; (ii) there exists $t_1 > t$ such that $t_1 \Vdash C$ and, for every $t < t' < t_1$, $t' \Vdash B$ hold; (iii) there exists $t_2 > t$ such that $t_2 \Vdash E$ and, for every $t < t' < t_2$, $t' \Vdash D$ hold. The possible relationships among $t+1, t_1$ and t_2 are the following: (1) $t+1 < t_1, t_2$. In this case, $t+1 \Vdash \{A, B, D, \circ(B\mathcal{U}C), \circ(D\mathcal{U}E)\}$ holds; (2) $t+1 = t_1$ and $t_1 \leq t_2$. Hence $t+1 \Vdash \{A, C, (D\mathcal{U}E)\}$ holds; (3) $t+1 = t_2$ and $t_1 > t_2$. So we get $t+1 \Vdash \{A, B, \circ(B\mathcal{U}C), E\}$ holds. Summarizing rule Lin handles $\circ$-formulas introducing in the conclusion one branch for every formula of the kind $\circ(A\mathcal{U}B)$ and one branch for all the other kind of $\circ$-formulas. In Figure 2 we show the tableau tree for the example.

At first sight, rule Lin does not seem helpful to perform automated deduction, since it can generate an huge number of branches. However, as discussed in [2] for DUM, theorem provers using the multiple-conclusion rules as Lin can be effective. Moreover, a theorem prover can benefit from some further formulas (we call them side formulas) that we can insert in the conclusions of Lin. The side formulas represent correct information which is not necessary to handle to get the completeness. Let us suppose that neither (1) nor (2) hold. Then we can also prove that $t+1 \Vdash \neg(D \wedge \circ E)$ holds. When $D = \top$ (that is $D\mathcal{U}E$ coincides with $\diamond E$), then we can prove that for every $t' \geq t+2$, $t' \Vdash \neg E$ holds. This information is not necessary to the deduction but it can be exploited to perform automated deduction. In particular, when the eventuality $D\mathcal{U}E$

 ISBN: 978-1-61208-394-0

$$\frac{\circ A_1, \ldots, \circ A_n, \circ(B_1 \mathcal{U} C_1), \ldots, \circ(B_m \mathcal{U} C_m)}{\mathcal{S}, B_1, \ldots, B_m, \circ(B_1 \mathcal{U} C_1), \ldots, \circ(B_m \mathcal{U} C_m) | \mathcal{S}, C_1, B_2 \mathcal{U} C_2, \ldots, B_m \mathcal{U} C_m | H_2 | \ldots | H_m} \text{Lin}$$

where, $\mathcal{S} = \{A_1, \ldots, A_n\}$ and for i=2,...,m

$H_i = ((\mathcal{S} \cup \{\circ(B_1 \mathcal{U} C_1), \ldots, \circ(B_m \mathcal{U} C_m)\}) \setminus \{\circ(B_i \mathcal{U} C_i), \ldots, \circ(B_m \mathcal{U} C_m)\}) \cup \{B_1, \ldots, B_{i-1}, C_i, \neg(B_i \wedge \circ C_i), B_{i+1} \mathcal{U} C_{i+1}, \ldots, B_m \mathcal{U} C_m\}$

Figure 1. The multiple-conclusion rule Lin

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\circ A, \circ(B \mathcal{U} C), \circ(D \mathcal{U} E)}{A, B, D, \circ(B \mathcal{U} C), \circ(D \mathcal{U} E)} \text{Lin}}{\dfrac{C, D \mathcal{U} E}{C, E \mid \dfrac{C, \circ(D \mathcal{U} E)}{E} \text{Lin}} \mathcal{U} \mid \dfrac{\circ(B \mathcal{U} C), E}{C} \text{Lin}} \mid \dfrac{A, C, D \mathcal{U} E}{A, C, E \mid \dfrac{A, C, \circ(D \mathcal{U} E)}{E} \text{Lin}} \mathcal{U} \mid \dfrac{A, \circ(B \mathcal{U} C), E}{C} \text{Lin}}$$

where $\mathcal{U}$ denotes the application of rule $\dfrac{A \mathcal{U} B}{B | \circ(A \mathcal{U} B)} \mathcal{U}$

Figure 2. Example of application of rule Lin

coincides with $\diamond E$ we get that from the time $t+2$ the formula E coincides with $\bot$. As a consequence, we have that if there is a loop, then t and $t+1$ are not part of the loop and a theorem prover can reset the history information. Moreover, since E is equivalent to $\bot$, it is correct to replace all the occurrences of the formula E with $\bot$. Rules based on replacement of formulas with logical constants have been proved effective both for INT [1] and for DUM [2].

III. REPLACEMENTS RULES

In this section, we consider $\Box$ in the language. We write $S[B/A]$ to denote the set of formulas obtained by replacing in S every occurrence of A with B. The rules Replace-$\Box$ and Replace-$\Box\neg$ given below are the analogous of rules Replace-**T** and Replace-**T**$\neg$ given in [1][13]:

$$\frac{S, \Box A}{S[\top/A], \Box A} \text{Replace-}\Box, \quad \frac{S, \Box \neg A}{S[\bot/A], \Box \neg A} \text{Replace-}\Box\neg.$$

A special case of Replace-$\Box$ and Replace-$\Box\neg$ are the rules

$$\frac{S, A}{S\{\top/A\}, A} \text{Replace-}cl, \quad \frac{S, \neg A}{S\{\bot/A\}, \neg A} \text{Replace-}cl\text{-}\neg.$$

where $S\{B/A\}$ (note the curly braces) denotes the set of formulas obtained by replacing with B the occurrences of A in S that are not under the scope of any temporal connective.

In [1], we have introduced some variants of Replace-$\Box$ and Replace-$\Box\neg$ based on the Kleene sign property. We exploit some conditions under which we can replace a propositional variable p applying the rules Replace-$\Box$ and Replace-$\Box\neg$ also when neither $\Box p$ nor $\Box\neg p$ explicitly occur in the premise of the rule. The condition for the applicability of these rules is based on the notion of *polarity* of p: p can be eliminated from a set of formulas S (replaced with $\top$ or $\bot$) if all the occurrences of p in S have the same polarity. The notion of polarity can be easily explained in the framework of PLTL where formulas of the kind $A \rightarrow B$ are written as $\neg A \vee B$ and Negation Normal Form is applied: a propositional variable p occurs with positive (resp. negative) polarity in a set S, denoted with $p \preceq^+ S$ (resp. $p \preceq^- S$) iff no occurrence (resp. every occurrence) of p in S is of the kind $\neg p$. The following are replacement rules of propositional variables fulfilling the notion of positive or negative occurrence in a set that can be computed in linear time on S:

$$\frac{S}{S[\top/p]} \preceq^+, \text{ provided } p \preceq^+ S; \quad \frac{S}{S[\bot/p]} \preceq^-, \text{ provided } p \preceq^- S.$$

In Figure 3 we show a piece of deduction for the formula `acacia-demo-v3_1`, where also we apply some obvious boolean simplifications.

IV. PRELIMINARY RESULTS

We have developed a Prolog prototype to perform some experiments on the benchmark formulas for PLTL. The development of the prover is at the very early stage. We have only focused on the implementation of the logical calculus with rule Lin in its first simplified version (without side formulas) and the optimization rules provided in Section III. The part of the prover related to the history construction, loop-checking and loop-satisfaction is very naive. Thus, in general, the known provers outperform our implementation. However, there are some remarks related to the proposed optimizations that deserve a comment. First, all the optimizations rules we have described are effective in speeding-up the deduction. Second, the rules $\preceq^+$ and $\preceq^-$ apply to the benchmark formulas `acacia-demo-v3`, `alaska-szymanski`, `rozier` (some subfamilies), `O1-schuppan` and `trp` (some subfamilies) without requiring any deduction step. On our Mac OS X (2.7 GHz, Core i7, 8GB), in less than 10 (often in less that 1) seconds, the prototype decides the families `acacia`, `alaska-lift` (except for the non-negated `l` variant), `alaska-szymanski`, `anzu-amba`, `anzu-amba_c` and `anzu-amba_cl` in negated version, `forobotsr1f0` (many formulas in the negated version and in a few cases also the non-negated versions), `rozier-formulas`, `rozier-patterns`, `schuppan-O1` and `trp` (some cases). Without the described optimizations timings would be greater by some order of magnitudes.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{H \equiv \top\,\mathcal{U}\,(canc \wedge \circ\neg go) \vee (\Box(\neg req \vee \circ grt \vee \circ\circ grt \vee \circ\circ\circ grt) \wedge \Box(\neg grt \vee \circ\neg grt) \wedge \Box(\neg canc \vee \circ(\neg grt\,\mathcal{U}\,go)))}{\top\,\mathcal{U}\,(canc \wedge \circ\neg go) \vee (\Box(\neg\bot \vee \circ grt \vee \circ\circ grt \vee \circ\circ\circ grt) \wedge \Box(\neg grt \vee \circ\neg grt) \wedge \Box(\neg canc \vee \circ(\neg grt\,\mathcal{U}\,go)))}\,\texttt{bool}}{\top\,\mathcal{U}\,(canc \wedge \circ\neg go) \vee (\Box(\top \vee \circ grt \vee \circ\circ grt \vee \circ\circ\circ grt) \wedge \Box(\neg grt \vee \circ\neg grt) \wedge \Box(\neg canc \vee \circ(\neg grt\,\mathcal{U}\,go)))}\,\texttt{bool}}{H' \equiv \top\,\mathcal{U}\,(canc \wedge \circ\neg go) \vee (\Box(\top) \wedge \Box(\neg grt \vee \circ\neg grt) \wedge \Box(\neg canc \vee \circ(\neg grt\,\mathcal{U}\,go)))}\preceq^{-}, grt \preceq^{-} H'}{\top\,\mathcal{U}\,(canc \wedge \circ\neg go) \vee (\Box(\top) \wedge \Box(\neg\bot \vee \circ\neg\bot) \wedge \Box(\neg canc \vee \circ(\neg\bot\,\mathcal{U}\,go)))}\,\texttt{bool}}{\top\,\mathcal{U}\,(canc \wedge \circ\neg go) \vee (\Box(\top) \wedge \Box(\top \vee \circ\top) \wedge \Box(\neg canc \vee \circ(\top\,\mathcal{U}\,go)))}\,\texttt{bool}}{\top\,\mathcal{U}\,(canc \wedge \circ\neg go) \vee (\Box(\top) \wedge \Box(\top) \wedge \Box(\neg canc \vee \circ(\top\,\mathcal{U}\,go)))}\,\texttt{bool}}{\top\,\mathcal{U}\,(canc \wedge \circ\neg go) \vee \Box(\neg canc \vee \circ(\top\,\mathcal{U}\,go))}$$

Figure 3. Example of application of rule $\preceq^{-}$ to the formula `acacia-demo-v3_1`

V. Conclusion and Future Work

We have presented our ongoing research on automated deduction for PLTL. We face the problem along different lines. In this note we have discussed two of them: Section II provides some ideas for a new proof-theoretical characterization of PLTL based on a multiple-conclusion rule; Section III describes logical rules to cut the size of the proofs. In addition to the given results, an important part of the future work is to exploit the notions of *local formula* [3] and *evaluation* [14] to develop an advanced strategy to avoid some rule application.

Acknowledgment

The third author acknowledges the support of the MIUR PRIN 2010-2011 grant "Automi e Linguaggi Formali: Aspetti Matematici e Applicativi", code 2010LYA9RH.

References

[1] M. Ferrari, C. Fiorentini, and G. Fiorino, "Simplification rules for intuitionistic propositional tableaux," ACM Trans. Comput. Logic, vol. 13, no. 2, Apr. 2012, pp. 14:1–14:23. [Online]. Available: http://doi.acm.org/10.1145/2159531.2159536

[2] G. Fiorino, "Refutation in Dummett logic using a sign to express the truth at the next possible world," in IJCAI, T. Walsh, Ed. IJCAI/AAAI, 2011, pp. 869–874.

[3] M. Ferrari, C. Fiorentini, and G. Fiorino, "fCube: An efficient prover for intuitionistic propositional logic," in LPAR (Yogyakarta), ser. Lecture Notes in Computer Science, C. G. Fermüller and A. Voronkov, Eds., vol. 6397. Springer, 2010, pp. 294–301.

[4] K. Brünnler and M. Lange, "Cut-free sequent systems for temporal logic," J. Log. Algebr. Program., vol. 76, no. 2, 2008, pp. 216–225.

[5] J. Gaintzarain, M. Hermo, P. Lucio, M. Navarro, and F. Orejas, "Dual systems of tableaux and sequents for PLTL," J. Log. Algebr. Program., vol. 78, no. 8, 2009, pp. 701–722.

[6] S. Schwendimann, "A new one-pass tableau calculus for PLTL," in Tableaux'98, 1998, pp. 277–291.

[7] R. Pliuskevicius, "Investigation of finitary calculus for a discrete linear time logic by means of infinitary calculus," in Baltic Computer Science, ser. Lecture Notes in Computer Science, J. Barzdins and D. Bjørner, Eds., vol. 502. Springer, 1991, pp. 504–528.

[8] B. Paech, "Gentzen-systems for propositional temporal logics," in CSL, ser. Lecture Notes in Computer Science, E. Börger, H. K. Büning, and M. M. Richter, Eds., vol. 385. Springer, 1988, pp. 240–253.

[9] A. Bolotov, O. Grigoriev, and V. Shangin, "Automated natural deduction for propositional linear-time temporal logic," in TIME. IEEE Computer Society, 2007, pp. 47–58.

[10] M. Fisher, C. Dixon, and M. Peim, "Clausal temporal resolution," ACM Trans. Comput. Log., vol. 2, no. 1, 2001, pp. 12–56.

[11] M. Suda and C. Weidenbach, "Labelled superposition for PLTL," in LPAR, ser. Lecture Notes in Computer Science, N. Bjørner and A. Voronkov, Eds., vol. 7180. Springer, 2012, pp. 391–405.

[12] G. Fiorino, "Tableau calculus based on a multiple premise rule," Information Sciences, vol. 180, no. 19, 2010, pp. 371–399.

[13] F. Massacci, "Simplification: A general constraint propagation technique for propositional and modal tableaux," in Proc. International Conference on Automated Reasoning with Analytic Tableaux and Related Methods, Oosterwijk, The Netherlands, ser. LNCS, H. de Swart, Ed., vol. 1397. Springer-Verlag, 1998, pp. 217–232.

[14] M. Ferrari, C. Fiorentini, and G. Fiorino, "An evaluation-driven decision procedure for G3i," TOCL, in press.

 ISBN: 978-1-61208-394-0