A Semantical Method To Reduce Branching In Tableau Proofs

Alessandro Avellone, Guido Fiorino, and Ugo Moscato

Dipartimento di Metodi Quantitativi per l'Economia, Università Milano-Bicocca,Piazza dell'Ateneo Nuovo, 1, 20126 Milano, Italy,
{alessandro.avellone,guido.fiorino,ugo.moscato}@unimib.it

Abstract. We describe a technique, we call it Semantical Constraint Propagation (SCP), to reduce branching in tableau proofs. SCP is applicable to classical logic and the same ideas can adapted to other logics. At the end of this note we give the results obtained with our theorem prover PITP for propositional intuitionistic logic with SCP turned on and off.

1 Introduction

The main concern in proofs for classical, modal and description logics is to bound branching, which is the source of inefficiency. Both for clausal and non-clausal theorem proving, a great deal of research has been done in order to bound branching and many techniques employed in clausal theorem proving have been generalized or adapted to non-clausal theorem proving ([3, 2, 5, 4, 1]).

Here we are interested in tableau systems. The contribution of this note is a technique, we call it Semantical Branching Propagation (SCP for short), to bound branching in tableau proofs. Although in this note we consider the Smullyan tableau calculus for propositional classical logic, the same ideas can be applied to other logics. SCP is a strategy to select branching formulas in tableau proofs and is justified by semantical considerations. It can be inserted in a theorem prover together with the well known optimization techniques. We present some experimental results of a C++ implementation (PITP) for propositional intuitionistic logic. We have tested our implementation on the ILTP Library ([6]) and on formulas generated at random.

2 Preliminary and Notation

In order to describe SCP, we take as main reference the Smullyan calculus for propositional classical logic (**Cl**) provided in [7]. The rules of the calculus are given in Figure 1. The meaning of the signs **T** and **F** is explained in terms of *realizability* as follows. Let σ be a classical model, then $\sigma \rhd \mathbf{T}A$ (σ *realizes* $\mathbf{T}A$) iff $\sigma \models A$ (where $\models$ is the usual binary relation between models and formulas) and $\sigma \rhd \mathbf{F}A$ iff $\sigma \not\models A$. A signed formula (swff for short) is a formula prefixed with **T** or **F**. Given a set S, $\sigma \rhd S$ iff σ realizes every swff in S. A proof table

α	α_1	α_2
$\mathbf{T}(A \wedge B)$	$\mathbf{T}A$	$\mathbf{T}B$
$\mathbf{F}(A \vee B)$	$\mathbf{F}A$	$\mathbf{F}B$
$\mathbf{F}(A \rightarrow B)$	$\mathbf{T}A$	$\mathbf{F}B$

α	α_1, α_2
$\mathbf{T}(\neg A)$	$\mathbf{F}A$
$\mathbf{F}(\neg A)$	$\mathbf{T}A$

β	β_1	β_2
$\mathbf{F}(A \wedge B)$	$\mathbf{F}A$	$\mathbf{F}B$
$\mathbf{T}(A \vee B)$	$\mathbf{T}A$	$\mathbf{T}B$
$\mathbf{T}(A \rightarrow B)$	$\mathbf{F}A$	$\mathbf{T}B$

$$\frac{S, \alpha}{S, \alpha_1, \alpha_2}(\alpha) \qquad \frac{S, \beta}{S, \beta_1 | S, \beta_2}(\beta)$$

Fig. 1. Tableau calculus for propositional classical logic

(or proof tree) for S is a tree, rooted with S and obtained by the subsequent application of the rules of the calculus. When a rule (γ), $\gamma \in \{\alpha, \beta\}$, is applied to a set S, we say that S *is expanded by the rule* (γ) and that *the swff* γ *is expanded*. A set S is *contradictory* if $\{\mathbf{T}A, \mathbf{F}A\} \subseteq S$ and S is *final* if S is contradictory or S contains atomic formulas only. A *closed proof table* is a proof table whose leaves all contain contradictory sets. Closed proof tables are the proofs of the calculus. By the soundness and completeness of the calculus above, a formula A is classically valid iff there exists a closed proof table starting from $\{\mathbf{F}A\}$. Given a non-contradictory set of swffs S, with σ_S we denote the classical model defined as follows: for every atom p, $\sigma_S \models p$ iff $\mathbf{T}p \in S$ and we call σ_S the model underlying S. Moreover, we call *core of* σ_S the set $\{p | \mathbf{T}p \in S$ or $\mathbf{F}p \in S$ with p an atom$\}$. A classical model σ' *extends* σ_S if σ' behaves as σ_S on the core of σ_S. We remark that if $\sigma_S \not\models H$ and $\mathcal{PV}(H)$ is included in the core of σ_S, then, for every model σ' extending σ_S, $\sigma' \not\models H$ holds, where with $\mathcal{PV}(H)$ we denote the atoms occurring H.

3 Description of SCP

We attained SCP while we were engaged in the implementation of PITP, a theorem prover for propositional intuitionistic logic (**Int**) based on a non-analytical tableau calculus. SCP is always applicable in **Cl** deductions. On the other hand, by the meaning of implication and negation in **Int**, in intuitionistic theorem proving SCP is not always applicable. Despite this, the performances of PITP with SCP are far better than PITP without SCP.

Loosely speaking, the idea behind SCP can be explained from a semantical point of view: a tableau proof can be seen as an attempt to build a model satisfying a set of formulas. The construction of such a model proceeds by increasing the information necessary to define the model. Step by step the accuracy of the model increases. This process stops if we get contradictory information, this means that no model can exist, or when we have enough information to build up a model. Thus every branch of a tableau proof ends with a contradiction or with a set of atomic formulas S and from such a set S a model can be immediately defined. During the proof the *semantical content* of a set is not used to *drive the proof*. From the semantical content of a set S we can decide which among many branching formulas in S has to be used to expand S. Moreover, under certain

semantical conditions although a set is not contradictory we can deduce that it is not realizable (thus by a subsequent application of the rules of the calculus we will get a closed proof table). Finally, from certain semantical conditions we can deduce that a non-final set is realizable (thus by applying the rules of the calculus we will get a branch ending in a set which is final and non-contradictory). Let us explain what we mean for *semantical content* and *drive a proof* by an example. Let $S = \{\mathbf{F}(P0 \wedge P2), \mathbf{F}(P0 \wedge P4), \mathbf{F}(P2 \wedge P4), \mathbf{F}(P1 \wedge P3), \mathbf{F}(P1 \wedge P5), \mathbf{F}(P3 \wedge P5), \mathbf{T}(P0 \vee P1), \mathbf{T}(P2 \vee P3), \mathbf{T}(P4 \vee P5)\}$. Since σ_S realizes the $\mathbf{F}$-swffs in S but σ_S does not realize none of the $\mathbf{T}$-swffs of S, then we choose one of the $\mathbf{T}$-swffs, let us suppose $\beta = \mathbf{T}(P0 \vee P1)$. The rule (β) is applied to S and $S_1 = (S \setminus \{\beta\}) \cup \{\mathbf{T}P0\}$ and $S_2 = (S \setminus \{\beta\}) \cup \{\mathbf{T}P1\}$ are the subsequent sets of S. Now consider S_1. Since σ_{S_1} realizes $\mathbf{T}P0$ and all the $\mathbf{F}$-swffs in S_1, but does not realize neither $\mathbf{T}(P2 \vee P3)$ nor $\mathbf{T}(P4 \vee P5)$, we choose one of them, let us suppose $\beta = \mathbf{T}(P2 \vee P3)$. The rule (β) is applied to S_1 and $S_3 = (S_1 \setminus \{\beta\}) \cup \{\mathbf{T}P2\}$ and $S_4 = (S_1 \setminus \{\beta\}) \cup \{\mathbf{T}P3\}$ are the subsequent sets of S_1. Now, since σ_{S_3} does not realize neither $\mathbf{F}(P0 \wedge P2)$ nor $\mathbf{T}(P4 \vee P5)$ we expand S_3 by choosing one of them, let us suppose $\beta = \mathbf{T}(P4 \vee P5)$. The rule (β) is applied to S_3 and $S_5 = (S_3 \setminus \{\beta\}) \cup \{\mathbf{T}P4\}$ and $S_6 = (S_3 \setminus \{\beta\}) \cup \{\mathbf{T}P5\}$ are the subsequent sets. Now $\mathbf{F}(P0 \wedge P2)$, $\mathbf{F}(P2 \wedge P4)$, $\mathbf{F}(P0 \wedge P4)$ are not realized by σ_{S_5}. By applying the rule (β), with β any of them, we get two contradictory sets. The proof goes similarly for S_2, S_4 and S_6. We emphasize that in the proof above, to every set S is applied a rule (β) such that $\sigma_S \not\rhd \beta$. By choosing β-swffs at random we would have a huge closed proof table.

SCP exploits the fact that every non contradictory set S in the tableau proof defines a classical model σ_S as follows: if $\mathbf{T}p \in S$, then $\sigma_S(p) = true$, otherwise $\sigma_S(p) = false$. If σ_S realizes S, then we can stop the expansion of S and by the completeness of the calculus we deduce that σ_S realizes the starting set of the tableau proof. If σ_S does not realize S, then (in general) we do not know if there exists a model satisfying S and we must expand S in a new set S' with the aim to describe with more accuracy the possible model satisfying S and S'. The first remark to be done is that if we want add information to the model described by S it is more promising to choose a swff $X \in S$ such that $\sigma_S \not\rhd X$, thus the subsequent set S' describes a model $\sigma_{S'}$ realizing X if $\sigma_{S'}$ satisfies the immediate subformula(s) of X. This choice is important if X is a β-swff. The expansion of a set S by using a swff X such that $\sigma_S \rhd X$ is useless work. Thus SCP states:

> Let S be a non-final set S. If $\sigma_S \not\rhd S$ and β-swffs are the only formulas occurring in S, then choose $\beta \in S$ such that $\sigma_S \not\rhd \beta$ and apply the rule (β) to S.

We can generalize SCP to α and β formulas:

> Let S be a non-final set S. If $\sigma_S \not\rhd S$, then (i) if there exists a α-swff $H \in S$ such that $\sigma_S \not\rhd H$, then apply the rule (H) to S. Otherwise (ii) choose a β-swff $H \in S$ such that $\sigma_S \not\rhd H$ and apply the rule (H) to S.

SCP preserves the completeness of the calculus. As a matter of fact, if a branch of a proof table starting from S ends with a non-contradictory set S', then SCP and the definition of $\sigma_{S'}$ imply $\sigma_{S'} \rhd S'$. The rules of the calculus are invertible, and this implies $\sigma_{S'} \rhd S$.

From an analysis of the previous example we realize that we can further improve SCP. The underlying model of every subsequent set of S_3 extends the model σ_{S_3}. Since $\mathbf{T}P0, \mathbf{T}P2 \in S_3$, every subsequent set of S_3 contains $\mathbf{T}P0$ and $\mathbf{T}P2$. It follows that every model extending σ_{S_3} does not realize $\mathbf{F}(P0 \wedge P2)$. This implies that S_3 is not realizable. Thus expanding S_3 is pointless since $\mathbf{F}(P0 \wedge P2)$ is not realizable by the models extending σ_{S_3}. This is immediately clear if we apply the rule (β) to S_3 with $\beta = \mathbf{F}(P0 \wedge P2)$, indeed the subsequent sets are both contradictory. Summarizing, a set S is not realizable if σ_S does not realize a formula $H \in S$ and $\mathcal{PV}(H)$ is included in the core of σ_S (this condition has a syntactic and more general counterpart when the technique of [5] is used). Since the semantic check allows us to deduce that a set S is not realizable without to expand S to a closed proof table, some work is spared.

The computational cost of SCP coincides with a satisfiability check, which is linear in the length of the input formula. On the other hand avoiding a branching rule can greatly reduce the size of a proof tree. As intuitive comparison we can consider the tableau proof of our example when SCP is used or a "blind" proof is performed by expanding a set with the first β-swff occurring in it. Finally, some considerations can help to save search time for β-swffs. As an example if S' is the set obtained from S by an application of a β-rule and S and S' contain the same atomic swffs, then $\sigma_S = \sigma_{S'}$ and this implies that for every $H \in S$, $\sigma_S \rhd H$ iff $\sigma_{S'} \rhd H$. Moreover $\sigma_{S'} \not\rhd \beta_1$ and $\sigma_{S'} \not\rhd \beta_2$. If $\sigma_S \not\rhd \alpha$ and $S' = (S \setminus \alpha) \cup \{\alpha_1, \alpha_2\}$ and $\sigma_S = \sigma_{S'}$, then by the satisfiability check on α we know if $\sigma_{S'} \not\rhd \alpha_1$ holds or $\sigma_{S'} \not\rhd \alpha_2$ holds. Thus, if $\sigma_S = \sigma_{S'}$, then the satisfiability check can be avoided or performed on one swff.

In the following tables we give the results of our theorem prover PITP for propositional intuitionistic logic on the benchmark formulas of ILTP library [6] and on formulas generated at random. We see that when SCP is turned off the performances are far worse than when SCP is turned on. We emphasize that in the case of intuitionistic logic, SCP is applicable only to formulas that do not contain implications and negations (in intuitionistic logic negation and implication are not interpreted as in classical logic). At present PITP does not implement optimizations such as backjumping, BCP and semantic branching. Moreover, no optimization is used to perform the satisfiability check. Summarizing PITP behaves as follows: first PITP tries to expand α-swffs, then β-swffs, then swffs requiring backtracking (a case that does not arises in classical logic). When a β-rule has to be applied to a set S, PITP behaves as follows: (ia) if there exists a β-swff H such that H only contains $\wedge$ and $\vee$ connectives, $\sigma_S \not\rhd H$ and $\mathcal{PV}(H)$ is included in the core of σ_S, then PITP returns that S is not realizable. Otherwise (ib) if there exists a β-swff H such that H only contains $\wedge$ and $\vee$ connectives and $\sigma_S \not\rhd H$, then PITP expands H. Otherwise (ii) PITP expands one of the β-swffs. If PITP is run with SCP turned off, then PITP do not perform (ia)

Experimental results on ILTP library: index and CPU time
of the largest formula in the family decided within 600s CPU time

Wff family	SCP Off	SCP On
SYJ201+1	20 (1170e-3)	20 (30e-3)
SYJ202+1	3 (90e-3)	9 (572860e-3)
SYJ207+1	4 (40030e-3)	4 (40780e-3)
SYJ208+1	4 (2380e-3)	8 (83630e-3)
SYJ209+1	10 (539090e-3)	10 (534330e-3)
SYJ211+1	20 (505960e-3)	20 (504430e-3)
SYJ212+1	11 (501690e-3)	11 (503720e-3)

Experimental results on 2000 random generated formulas with 1000 connectives and 30 variables

	$<$ 1 sec	1 to 10 secs	10 to 60 secs	60 to 600	$\geq$ 600
SCP on	1995 (12)	4 (13)	1 (24)	0	0
SCP off	1264 (29)	88 (332)	36 (1136)	68 (15601)	544 ($\geq$ 326400)

Experimental results on 500 random generated formulas with 100000 connectives and 1000 variables

	$<$ 1 sec	1 to 10 secs	10 to 60 secs	60 to 600	$\geq$ 600
SCP on	11 (7)	432 (2291)	57(653)	0	0
SCP off	115 (61)	137 (336)	26 (807)	16 (3260)	206 ($\geq$ 123600)

Table 1. Experimental Results

and (ib). Since we do not have a propositional classical theorem prover, to have an idea of the behaviour of SCP in classical theorem proving we run PITP on random generated formulas of the of the kind $A \rightarrow B$, where A and B contain the same number of connectives and $\wedge$ and $\vee$ are the only connectives occurring in them. The results (on a Pentium II Klamath 300 MHz, cache size 512 KB, RAM 128 MB) are given in Table 1, where for each time interval, the number of formulas and, in parenthesis, the overall time to decide them are indicated.

References

1. Gabriel Aguilera, Inman P. de Guzmán, Manuel Ojeda-Aciego, and Agustín Valverde. Reductions for non-clausal theorem proving. *Theor. Comput. Sci.*, 266(1-2):81–112, 2001.
2. Jon William Freeman. *Improvements to propositional satisfiability search algorithms*. PhD thesis, University of Pennsylvania, 1995.
3. Fausto Giunchiglia and Roberto Sebastiani. Building decision procedures for modal logics from propositional decision procedure - The case study of modal K. *CADE*, volume 1104 of *LNCS*, pages 583–597. Springer, 1996.
4. Ian Horrocks and Peter F. Patel-Schneider. Optimizing description logic subsumption. *J. Log. Comput.*, 9(3):267–293, 1999.
5. Fabio Massacci. Simplification: A general constraint propagation technique for propositional and modal tableaux. *TABLEAUX*, volume 1397 of *LNCS*, pages 217–232. Springer-Verlag, 1998.
6. Thomas Raths, Jens Otten, and Christoph Kreitz. The ILTP library: Benchmarking automated theorem provers for intuitionistic logic. *TABLEAUX*, volume 3702 of *LNCS*, pages 333–337. Springer, 2005.
7. R.M. Smullyan. *First-Order Logic*. Springer, Berlin, 1968.