An ExpTime Tableau-based Decision Procedure for $\mathcal{ALCQI}$

Yu Ding and Volker Haarslev
Concordia University, Montreal, Canada

1 Motivation and Notions

The *algebraic method*, when used to deal with *qualified number restrictions*, is realized by an *atomic decomposition* step that generates an exponential number of sub-problems[HTM01]. It is not very clear this approach could lead to worst-case optimal tableau-based decision procedures for the fundamental *concept satisfiability* problem. Interestingly, it is known from practice that the *algebraic method* has better *run-time performance*. So far the practical success however is confined in DLs without *inverse roles*, and it is unclear how to use the *algebraic method* for DLs having both $\mathcal{Q}$ and $\mathcal{I}$. All these are baffling and beg for answers.

Firstly, we confirm that the *algebraic method* in general, according to the well-known result on *integer linear programming*[Pap81], leads to a worst-case ExpTime tableau-based decision procedure for the *concept satisfiability* problem. Secondly, we extends the *algebraic method* to DLs with *inverse roles*. To realize two goals simultaneously, several ingredients must be prepared to form a recipe. Naturally, the well-known *global (sub)tableaux caching technique* and the *atomic decomposition* principle must be selected. But neither of the two is directly applicable to DLs with inverse roles, extra ingredients are given below.

Tableau Structure: The *tableau structure* (TS) is a *labeled graph*. Each node is labeled with a set of *concepts*, each edge is labeled with a *role*. Additionally, (1) each node is labeled with (0 to many) algebraic objects[1]; (2) each edge is associated with two variables, one for *indicator* and one for *cardinality number*.

Propositional Branch: This notion abstracts the execution of the $\sqcap$-rule and the $\sqcup$-rule (common of *tableau expansion rules* for DLs in the $\mathcal{ALC}$-family).

Cut Formulae: Given a concept E subject to *concept satisfiability test* w.r.t. a GCI $\top \sqsubseteq G$ in $\mathcal{ALCQI}$, for each modal subformula (of E and G) of the form $\exists^{\bowtie n}R.C$ (where R is any role), $\top \sqsubseteq C \sqcup \tilde{C} \sqcup \exists^{\leq 0}R^{-}.\top$ is a *cut formula*[2].

Constraints Fine-Tuning:[3] In the *tableau structure*, let x and y be neighbors, x is completed and y is not completed, x has a R^{-}-edge to y, we have:

- if $\exists^{\leq n}R.C \in \mathcal{B}(y)$ and $C \in \mathcal{C}(y)$, then $\exists^{\leq n-1}R.C \in \mathcal{B}'(y)$; else $\exists^{\leq n}R.C \in \mathcal{B}'(y)$;
- if $\exists^{\geq n}R.C \in \mathcal{B}(y)$ and $C \in \mathcal{C}(y)$, then $\exists^{\geq n-1}R.C \in \mathcal{B}'(y)$; else $\exists^{\geq n}R.C \in \mathcal{B}'(y)$;

where $\mathcal{B}(y)$ denotes the *working propositional branch*, and $\mathcal{B}'(y)$ its *fine-tuning*. The adjustment of restrictions of a *propositional branch* at a node based on the *cut-set*s ($\mathcal{C}(y)$, concepts from cut formulae) at its neighbors is called *fine-tuning*.

Atomic Decomposition and Integer Linear Programs: What is typical of the *algebraic approach* is the building of one *integer linear program* from the decomposition of *role fillers* for a group of *(qualified) number restrictions*.

---

[1] Each object represents an integer program $[A|B]x = b$, where A is $m \times (2^m - 1)$ coefficient matrix, B is $m \times m$ matrix, b is a m-vector, s.t. $x \geq 0$, integer.

[2] It can be read in $\mathcal{ALCQI}$ as $\exists^{\geq 1}R^{-}.\top \sqsubseteq C \sqcup \tilde{C}$ or in $\mathcal{ALCI}$ as $\exists R^{-}.\top \sqsubseteq C \sqcup \tilde{C}$.

[3] Due to space limit, the fine-tuning shown here is correct only for tree structures.

The *atomic decomposition* forms all combinations[4] about *role fillers* and their negations. Each such combination is considered as a conjunction, and will be formed as a successor node. The unsat. of one combination (i.e. one successor) is reflected by setting the indicator variable to 0, which in turn might affect the feasibility of the *integer program* at the *fine-tuned working propositional branch*.

2 A Short Description of the Decision Procedure

Decision Procedure: There are two major data structures: `Nogood` (for unsat *caching* across different tableau structures) and `Witness` (for static *blocking* of tableau nodes within one tableau structure). The procedure starts building a tableau structure (TS) from a root node labeled with the concept subject to satisfiability test. It switches to explore a different TS if the current one can not be saturated clash-free. When switching to explore another TS, at least one new `Nogood` will be found. Thus, at most $2^{O(n)}$ number of TSs will be explored[5].

Complexity Analysis: Three factors are considered : (1) the maximum cost per *tableau node*; (2) the maximum size of a *tableau structure*; (3) the maximum number of tableau structures necessarily to form. We allow $2^{O(n^c)}$ cost per node for some constant c, but require the size of each tableau structure be of $2^{O(n)}$ and only $2^{O(n)}$ such tableau structures be formed. So, the total cost is $2^{O(n^{max(1,c)})}$.

Conclusion: For DLs with both *inverse roles* and *(qualified) number restrictions*, we have introduced restricted *cut-formulae* to guarantee the soundness of the *global (sub)-tableaux caching*[DM99], and introduced the *fine-tuning* of qualified number restrictions to adapt the *algebraic method* to DLs with inverse roles. The former can be treated like GCIs; the latter can be integrated in the *tableau expansion rules* as shown in [HTM01] for $\mathcal{SHQ}$. Solving *integer linear program* is known to be NP-complete, and so is the *propositional satisfiability* problem. Both problems possibly appear simultaneously in each tableau node. The decision procedure is designed to respect these facts. Further, the *inconsistency propagation rules*[DM99] are extended to work on *tableau structures* labeled with *algebraic objects*. The decision procedure is a worst-case ExpTime framework for $\mathcal{SHOQ}$ and $\mathcal{SHIQ}$. Details and proofs are omitted. Comments and references to important literature are regrettably impossible to give here.

References

[DM99] Francesco M. Donini and Fabio Massacci. Exptime tableaux for alc. *Artificial Intelligence*, 124:87–138, 1999.

[HTM01] Volker Haarslev, Martina Timmann, and Ralf Möller. Combining tableaux and algebraic decision procedures for dealing with qualified number restrictions in description logics. *IJCAR-2001*, pages 39–48, 2001.

[Pap81] Christos H. Papadimitriou. On the complexity of integer programming. *J. ACM*, 28(4):765–768, 1981.

[4] Regardless of the differences, the *atomic decomposition* and the *integer linear programs* have intricate connections to the *choose-rule* and $\leq$-rule.

[5] To explore one TS, either DFS or BFS suffices. There is no assumption about the order in which TSs are explored. The switching could be implemented either as restarting or backjumping, or any sound method that shares previous computations.