TouIST: a Friendly Language for Propositional Logic and More

Jorge Fernandez[1], **Olivier Gasquet**[1], **Andreas Herzig**[2], **Dominique Longin**[2], **Emiliano Lorini**[2], **Frédéric Maris**[1] and **Pierre Régnier**[1]

[1]IRIT - Univ. Paul Sabatier, Toulouse
[2]IRIT - CNRS, Toulouse
{Jorge.Fernandez, Olivier.Gasquet, Andreas.Herzig, Dominique.Longin, Emiliano.Lorini, Frederic.Maris, Pierre.Regnier}@irit.fr

Abstract

This work deals with logical formalization and problem solving using automated solvers. We present the automatic translator TouIST that provides a simple language to generate logical formulas from a problem description. Our tool allows us to model many static or dynamic combinatorial problems and to benefit from the regular improvements of SAT, QBF or SMT solvers in order to solve these problems efficiently. In particular, we show how to use TouIST to solve different classes of planning tasks in Artificial Intelligence.

1 Introduction to TouIST

TouIST is an automatic translator which offers user friendly language and graphical interface to easily use SAT, SMT (SAT Modulo Theories) or QBF solvers. Input formulas need not to be in clausal form and arbitrary connectives may be used. The translation into DIMACS, QDIMACS or SMT-LIB format is done automatically, depending on the selected solver. Beyond the Boolean connectives of propositional logic, the input language of TouIST has sets, conjunctions and disjunctions parametrized by sets, abbreviations...We can directly express complex propositional formulas such as:

$$\bigwedge_{i \in \{1..N\}} \bigvee_{X \in S(i)} \bigwedge_{n \in \mathrm{X}} \bigwedge_{m \in \mathrm{Y} | m \neq n} (p_{i,X,n} \rightarrow \neg p_{i,X,m})$$

where we can define the variable N as a particular integer, the $S(i)$ as sets of sets of symbols for each $i \in \{1, \ldots, N\}$, and Y as a set of symbols. For example, if $N = 2$, $S(1) = \{\{blue, red\}, \{red\}\}$, $S(2) = \{\{red\}, \{blue\}, \{white, blue, red\}\}$, and $Y = \{white, red\}$, we write in the TouIST input language:

```
$N = 2
$S(1) = [[blue, red], [red]]
$S(2) = [[red], [blue], [white, blue, red]]
$Y = [white, red]
bigand $i in [1..$N]:
  bigor $X in $S($i):
    bigand $n in $X:
      bigand $m in $Y when $m!=$n:
        p($i,$X,$n) => not p($i,$X,$m)
      end
    end
  end
end
```

We can also use multiple binding of indexes as in $\bigwedge_{i \in A, j \in B}$ and rich computations on indexes as well as on domain sets as in $\bigwedge_{i \in (A \cup (B \cap C))}$, expressed in the TouIST input language as:

```
bigand $i,$j in $A,$B:
  ...
end
bigand $i in $A union ($B inter $C):
  ...
end
```

Running the solver only consists in clicking a button and the tool displays the models successively computed by the solvers in the syntax of the input formula. Literals of interest can be filtered by regular expressions. Moreover, it is possible to use the software in command line and/or batch modus.

TouIST is publicly available for download from the following site: https://www.irit.fr/TouIST/

On the one hand, the tool can be used by researchers to compute logical encodings of problems, for example of symbolic AI problems such as planning tasks. On the other hand, TouIST is a pedagogical tool to show the power of propositional logic to students who have been trained a couple of hours to formalize sentences in logic and who have acquired basic notions of validity and satisfiability: it allows them to automatically solve some combinatorial puzzles.

In the sequel, we introduce several examples of static and dynamic reasoning that will serve to demonstrate TouIST.

2 Static Reasoning with TouIST

2.1 Solving Puzzles with SAT

TouIST allows us to encode and solve static generalized games such as the well known Sudoku for a $N \times N$ grid. For example, to express that each cell must have at least one value we write the formula:

$$\bigwedge_{i \in [1..N]} \bigwedge_{j \in [1..N]} \bigvee_{k \in [1..N]} p(i,j,k)$$

where $p(i,j,k)$ means that cell (i,j) has value k.
This formula is expressed in the TouIST input language as:

```
bigand $i,$j in [1..$N],[1..$N]:
  bigor $k in [1..$N]:
    p($i,$j,$k)
  end
end
```

It also allows us to solve well-known puzzles and games involving epistemic deductive reasoning, given existing polynomial embeddings of fragments of epistemic logic into propositional logic. This includes "Guess Who?" and the muddy children puzzle [Barwise, 1981].

2.2 Solving Puzzles with SMT

In a similar way to Sudoku, the Binario (binary game) consists in filling a grid by deduction with only 0s and 1s. It is possible to model it in propositional logic, but to obtain a more compact encoding one can use SMT (SAT Modulo Theories) with atoms of QF-LIA (linear arithmetic on integers).[1]

In particular we can encode the rule "each row and column must contain as many 0s as 1s" by

$$\bigwedge_{i=1}^{N_R}\left(\sum_{j=1}^{N_C} x_{i,j} = \frac{N_C}{2}\right) \quad \wedge \quad \bigwedge_{j=1}^{N_C}\left(\sum_{i=1}^{N_R} x_{i,j} = \frac{N_R}{2}\right)$$

where N_R is the number of rows of the grid and N_C is the number of columns.

Another rule is that "there is no more than two of either number adjacent to each other", expressed in the TouIST input language as:

```
bigand $i,$j in [1..$NR-2],[1..$NC]:
  x($i,$j)!=x($i+1,$j) or x($i+1,$j)!=x($i+2,$j)
end
bigand $i,$j in [1..$NR],[1..$NC-2]:
  x($j,$i)!=x($j,$i+1) or x($j,$i+1)!=x($j,$i+2)
end
```

3 Dynamic Reasoning with TouIST

3.1 Finding a Winning Strategy

The language of QBF allows us to express naturally and concisely the existence of winning strategies as described in [Kroening and Strichman, 2016]. The moves of player 0 (for whom we are searching for a winning strategy) will be existentially quantified while those of his opponent will be universally quantified: we look for the moves of player 0 which will lead him to victory regardless of the moves made by player 1. TouIST natively integrates the QBF solver *Quantor 3.2* [Biere, 2005] and can be interfaced with other solvers supporting the QDIMACS format. Selecting this prover in TouIST allows us to use quantifiers $\forall$ and $\exists$ on propositional variables.

Figure 1 shows the exhaustive set of solutions in a Nim's game with four matches. The root of the tree represents the initial number of matches and each arrow represents the action of removing 1 ($\dashrightarrow$) or 2 ($\longrightarrow$) matches. We see that there is a winning strategy for player 0 if she starts. We are leveraging QBF to write this strategy in TouIST. The variable $takes\_2(i)$ is true if the current player takes 2 matches at step i and is false if she takes only one. If we denote by Φ the conjunction of formulas representing the rules of Nim's game then the existence of a winning strategy for player 0 is simply written:

$$\begin{aligned}&\exists takes\_2(0)\forall takes\_2(1)\\ &\quad \exists takes\_2(2)\forall takes\_2(3)\\ &\quad \exists takes\_2(4)\,.\,(\neg 0\_lost \wedge \Phi)\end{aligned}$$

3.2 Solving Classical Planning Tasks

A planning task can be transformed into a propositional formula whose models correspond to solution plans (i.e., sequences or steps of actions starting from an initial state and leading to a goal). These models can be found using a SAT solver [Kautz and Selman, 1992]. Numerous improvements of this approach have been proposed via the development of more compact and efficient encodings, see [Kautz and Selman, 1996; Ernst *et al.*, 1997; Mali and Kambhampati, 1999; Rintanen *et al.*, 2006] among others. We here illustrate the expressive power of the TouIST language by encoding of explanatory frame-axioms. If a fact is false at step $i-1$ of a solution plan and becomes true at step i then the disjunction of actions that can establish the fact (i.e. it is a positive effect of such an action) at step i of the plan is true. Indeed, at least one of the actions that can establish the fact must have been applied.

$$\bigwedge_{i\in\{1..PlanLength\}} \bigwedge_{f\in Facts} \left((\neg f(i-1) \wedge f(i)) \Rightarrow \bigvee_{a\in Actions|f\in Effects^+(a)} a(i)\right)$$

```
bigand $i in [1..$PlanLength]:
  bigand $f in $Facts:
    not $f($i-1) and $f($i) =>
      bigor $a in $Actions when $f in $Effects_pos($a):
        $a($i)
      end
  end
end
```

Much more compact QBF encodings have also been developed [Cashmore *et al.*, 2012; Gasquet *et al.*, 2018].

3.3 Solving Conformant/Temporal Planning Tasks

Beyond classic planning, TouIST allows us to encode and solve *conformant* planning tasks with QBF [Rintanen, 2007].

[1] The theories QF-IDL, QF-RDL (difference logic on integers/rationals) and QF-RDL (linear arithmetic on rationals) are also available in TouIST.

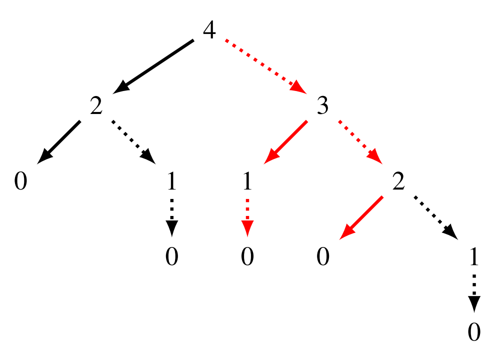

Figure 1: Solutions for Nim's game with 4 matches and 2 players. The winning strategy for player 0 is in red.

It can also be used to solve *temporal* planning tasks involving durative actions, exogenous events and temporally extended goals with SMT encodings [Shin and Davis, 2005; Rintanen, 2015]. We here focus on the SMT encoding rules proposed in [Maris and Régnier, 2008]. Below we give an encoding of temporal mutual exclusion of actions. If two actions a_1 and a_2, respectively producing a fact f (i.e. f is a positive effect of a_1) and its negation $\neg f$ (i.e. f is a negative effect of a_2), are active in the plan, then the time interval $[\tau^{+}_{\mathit{start}}(a_1, f), \tau^{+}_{\mathit{end}}(a_1, f)]$ corresponding to the activation of f by a_1 and the time interval $[\tau^{-}_{\mathit{start}}(a_2, f), \tau^{-}_{\mathit{end}}(a_2, f)]$ corresponding to the activation of $\neg f$ by a_2 are disjoint.

$$\bigwedge_{a_1 \in Actions} \; \bigwedge_{a_2 \in Actions} \; \bigwedge_{f \in Facts | f \in \mathit{Effects}^{+}(a_1) \cap \mathit{Effects}^{-}(a_2)} \left((a_1 \wedge a_2) \Rightarrow \begin{pmatrix} (\tau^{-}_{\mathit{end}}(a_2, f) < \tau^{+}_{\mathit{start}}(a_1, f)) \\ \vee \, (\tau^{+}_{\mathit{end}}(a_1, f) < \tau^{-}_{\mathit{start}}(a_2, f)) \end{pmatrix} \right)$$

```
bigand $a1,$a2,$f in $Actions,$Actions,$Facts
when $f in $Effects_pos($a1)and $f in $Effects_neg($a2):
    $a1 and $a2 =>
      (t_end_del($a2,$f) < t_start_add($a1,$f))
      or (t_end_add($a1,$f) < t_start_del($a2,$f))
end
```

3.4 TouISTPlan Module

In order to tune and compare different logical encodings of planning tasks we have implemented the TouISTPlan module which automatically solves planning tasks with TouIST. For example, thanks to this module we compared the performance of different QBF encodings for reference planning problems from different International Planning Competitions (IPC) [Gasquet *et al.*, 2018]. We were able to show that our new encodings are two times more efficient in terms of resolution time.

4 Conclusion

We have developed TouIST to offer a friendly language together with a modular tool that makes it easier to use SAT, SMT and QBF solvers. The aim of this demonstration is to show that TouIST can be used to solve many combinatorial problems, in particular for symbolic AI, and to spread its use in the community.

References

[Barwise, 1981] Jon Barwise. Scenes and other situations. *Journal of Philosophy*, 78(7):369–397, 1981.

[Biere, 2005] Armin Biere. Resolve and expand. In *Proceedings of the 7th International Conference on Theory and Applications of Satisfiability Testing*, SAT'04, pages 59–70, Berlin, Heidelberg, 2005. Springer-Verlag.

[Cashmore *et al.*, 2012] Michael Cashmore, Maria Fox, and Enrico Giunchiglia. Planning as quantified boolean formula. In *ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31 , 2012*, pages 217–222, 2012.

[Ernst *et al.*, 1997] Michael D. Ernst, Todd D. Millstein, and Daniel S. Weld. Automatic sat-compilation of planning problems. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, IJCAI 97, Nagoya, Japan, August 23-29, 1997, 2 Volumes*, pages 1169–1177, 1997.

[Gasquet *et al.*, 2018] Olivier Gasquet, Dominique Longin, Frederic Maris, Pierre Régnier, and Maël Valais. Compact tree encodings for planning as QBF. *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial*, 21(62):103–114, 2018.

[Kautz and Selman, 1992] Henry A. Kautz and Bart Selman. Planning as satisfiability. In *10th European Conference on Artificial Intelligence, ECAI 92, Vienna, Austria, August 3-7, 1992. Proceedings*, pages 359–363, 1992.

[Kautz and Selman, 1996] Henry A. Kautz and Bart Selman. Pushing the envelope: Planning, propositional logic and stochastic search. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence and Eighth Innovative Applications of Artificial Intelligence Conference, AAAI 96, IAAI 96, Portland, Oregon, USA, August 4-8, 1996, Volume 2*, pages 1194–1201, 1996.

[Kroening and Strichman, 2016] Daniel Kroening and Ofer Strichman. *Decision Procedures - An Algorithmic Point of View, Second Edition*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2016.

[Mali and Kambhampati, 1999] Amol Dattatraya Mali and Subbarao Kambhampati. On the utility of plan-space (causal) encodings. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence and Eleventh Conference on Innovative Applications of Artificial Intelligence, July 18-22, 1999, Orlando, Florida, USA*, pages 557–563, 1999.

[Maris and Régnier, 2008] Frederic Maris and Pierre Régnier. TLP-GP: new results on temporally-expressive planning benchmarks. In *20th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2008), November 3-5, 2008, Dayton, Ohio, USA, Volume 1*, pages 507–514, 2008.

[Rintanen *et al.*, 2006] Jussi Rintanen, Keijo Heljanko, and Ilkka Niemelä. Planning as satisfiability: parallel plans and algorithms for plan search. *Artif. Intell.*, 170(12-13):1031–1080, 2006.

[Rintanen, 2007] Jussi Rintanen. Asymptotically optimal encodings of conformant planning in QBF. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*, pages 1045–1050, 2007.

[Rintanen, 2015] Jussi Rintanen. Discretization of temporal models with application to planning with SMT. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 3349–3355, 2015.

[Shin and Davis, 2005] Ji-Ae Shin and Ernest Davis. Processes and continuous change in a sat-based planner. *Artif. Intell.*, 166(1-2):194–253, 2005.