GIST: A Solver for Probabilistic Games

Krishnendu Chatterjee[1], Thomas A. Henzinger[1], Barbara Jobstmann[2], and Arjun Radhakrishna[1]

[1] IST Austria (Institute of Science and Technology Austria)
[2] CNRS/Verimag, France

Abstract. GIST is a tool that (a) solves the qualitative analysis problem of turn-based probabilistic games with ω-regular objectives; and (b) synthesizes reasonable environment assumptions for synthesis of unrealizable specifications. Our tool provides the first and efficient implementations of several reduction-based techniques to solve turn-based probabilistic games, and uses the analysis of turn-based probabilistic games for synthesizing environment assumptions for unrealizable specifications.

1 Introduction

GIST (Game solver from IST) is a tool for (a) qualitative analysis of *turn-based probabilistic games ($2\frac{1}{2}$-player games)* with ω-regular objectives, and (b) computing environment assumptions for synthesis of unrealizable specifications. The class of $2\frac{1}{2}$-player games arise in several important applications related to verification and synthesis of reactive systems. Some key applications are: (a) synthesis of stochastic reactive systems; (b) verification of probabilistic systems; and (c) synthesis of unrealizable specifications. We believe that our tool will be useful for the above applications. GIST is available for download at `http://pub.ist.ac.at/gist`.

$2\frac{1}{2}$-player games. $2\frac{1}{2}$-player games are played on a graph by two players along with probabilistic transitions. We consider ω-regular objectives over infinite paths specified by parity, Rabin and Streett (strong fairness) conditions that can express all ω-regular properties such as safety, reachability, liveness, fairness, and most properties commonly used in verification. Given a game and an objective, our tool determines whether the first player has a strategy to ensure that the objective is satisfied with probability 1, and if so, it constructs such a witness strategy. Our tool provides the first implementation of qualitative analysis (probability 1 winning) of $2\frac{1}{2}$-player games with ω-regular objectives.

Synthesis of environment assumptions. The synthesis problem asks to construct a finite-state reactive system from an ω-regular specification. In practice, initial specifications are often unrealizable, which means that there is no system that implements the specification. A common reason for unrealizability is that assumptions on the environment of the system are incomplete. The problem of correcting an unrealizable specification Ψ by computing an environment assumption Φ such that the new specification $\Phi \rightarrow \Psi$ is realizable was studied in [3].

This research was supported by the European Union project COMBEST and the European Network of Excellence ArtistDesign..

The work [3] constructs an assumption Φ that constrains only the environment and is as weak as possible. Our tool implements the algorithms of [3]. We believe our implementation will be useful in analysis of realizability of specifications and computation of assumptions for unrealizable specifications.

2 Definitions

Game graphs. A *turn-based probabilistic game graph* ($2\frac{1}{2}$*-player game graph*) $G = ((S, E), (S_0, S_1, S_P), \delta)$ consists of a directed graph (S, E), a partition (S_0, S_1,S_P) of the finite set S of states, and a probabilistic transition function δ: $S_P \to \mathcal{D}(S)$, where $\mathcal{D}(S)$ denotes the set of probability distributions over the state space S. The states in S_0 are the *player*-0 states, where player 0 decides the successor state; the states in S_1 are the *player*-1 states, where player 1 decides the successor state; and the states in S_P are the *probabilistic* states, where the successor state is chosen according to the probabilistic transition function δ. *2-player game graphs* are a special case where $S_P = \emptyset$.

Objectives. We consider the three canonical forms of ω-regular objectives: Streett and its dual Rabin objectives; and parity objectives. The Streett objective consists of d request-response pairs $\{ (Q_1, R_1), (Q_2, R_2), \ldots, (Q_d, R_d) \}$ where Q_i denotes a request and R_i denotes the corresponding response (each Q_i and R_i are subsets of the state space). The objective requires that if a request Q_i happens infinitely often, then the corresponding response must happen infinitely often. The Rabin objective is its dual. The parity objective is a special case of Streett objectives where $Q_1 \subset R_1 \subset Q_2 \subset R_2 \subset Q_3 \subset \cdots \subset Q_d \subset R_d$.

Qualitative analysis. The qualitative analysis for $2\frac{1}{2}$-player games is as follows: the input is a $2\frac{1}{2}$-player game graph, and an objective Φ (Streett, Rabin or parity objective), and the output is the set of states such that player 0 can ensure Φ with probability 1. For detailed description of game graphs, plays, strategies, objectives and notion of winning see [1]. We focus on qualitative analysis because: a) In applications like synthesis the relevant analysis is qualitative analysis: the goal is to synthesize a system that behaves correctly with probability 1; (b) Qualitative analysis for probabilistic games is independent of the precise probabilities, and thus robust with imprecision in the exact probabilities (hence resilient to probabilistic modeling errors). The qualitative analysis can be done with discrete graph theoretic algorithms. Thus qualitative analysis is more robust and efficient, and our tools implements these efficient algorithms.

3 Tool Implementation

Qualitative analysis of $2\frac{1}{2}$-player games. Our tool presents the first implementation for the qualitative analysis of $2\frac{1}{2}$-player games with Streett, Rabin and parity objectives. We have implemented the linear-time reduction for qualitative analysis of $2\frac{1}{2}$-player Rabin and Streett games to 2-player Rabin and Streett games of [1], and the linear-time reduction for $2\frac{1}{2}$-player parity games to 2-player parity games of [4]. The 2-player Rabin and Streett games are solved by reducing them to the 2-player parity games using the LAR construction [5]. The 2-player parity games are solved using the tool PGSolver [6].

Environment assumptions for synthesis. Our tool implements a two-step algorithm for computing the environment assumptions as presented in [3]. The algorithm operates on the game graph that is used to answer the realizability question. First, a safety assumption that removes a minimal set of environment edges from the graph is computed. Second, a fairness assumption that puts fairness conditions on some of the remaining environment edges is computed. The problem of finding a minimal set of fair edges is computationally hard [3], and a reduction to $2\frac{1}{2}$-player games was presented in [3] to compute a locally minimal fairness assumption. The details of the implementation are as follows: given an LTL formula ϕ, the conversion to an equivalent deterministic parity automaton is achieved through GOAL [7]. Our tool then converts the parity automaton into a 2-player parity game by splitting the states and transitions based on input and output symbols. Our tool then computes the safety assumption by solving a safety model-checking problem. The computation of the fairness assumption is achieved in the following steps:

- Convert the parity game with fairness assumption into a $2\frac{1}{2}$-player game.
- Solve the $2\frac{1}{2}$-player game (using our tool) to check whether the assumption is sufficient (if so, go to the previous step with a weaker fairness assumption).

The synthesized system is obtained from a witness strategy of the parity game. The flow is illustrated in Figure 1.

We illustrate, how our tool works, on a simple example. Consider the LTL formula $\Phi = GF(g) \wedge G(c \rightarrow \neg g)$, where G and F denote globally and eventually, respectively. The specification says that we want to see infinitely many grants (g) but when we receive a cancel (c) we are not allowed to give a grant. From Φ our tool constructs a deterministic parity automaton that accepts exactly the words that satisfies Φ. The parity automaton is then converted into the parity game shown in Figure 2(a). We use $\Box$ to represent player-0 states and $\Diamond$ to represent player-1 states. In this example, the safety assumption is true, because the environment cannot force the play outside the cooperative winning region. Now, we are searching for a locally minimal fairness assumption by reducing a game with fairness assumptions on edges to a $2\frac{1}{2}$-player parity game (see [3]). If the initial state in the $2\frac{1}{2}$-player game is winning with probability 1 for player 0, then the assumption is sufficient. Figure 2(b) illustrates the $2\frac{1}{2}$-player game obtained with a fairness assumption on the edge $(0, 4)$. The $\bigcirc$ state is a probabilistic state with uniform distribution over its successors. The assumption on edge $(0, 4)$ is the minimal fairness assumption for the example. From this, our tool extract an automaton representing the environment assumption. For the example, we obtain an automaton equivalent to $G(c \rightarrow \neg g) \rightarrow GF(\neg c)$.

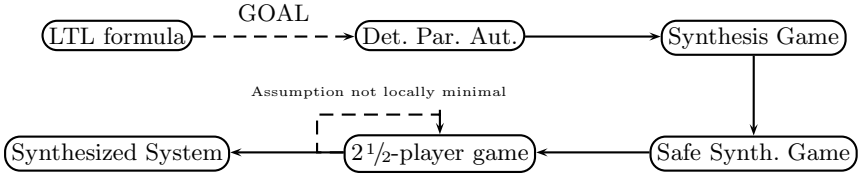

Fig. 1. The flow of the tool for computation of environment assumptions

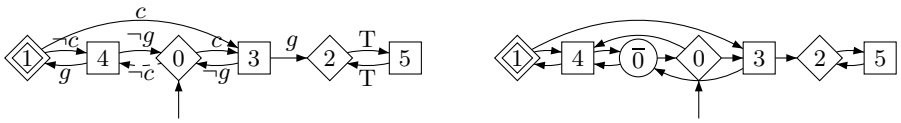

Fig. 2. (a) Parity game with fairness assumption (b) Equivalent $2\frac{1}{2}$-player game

The tools also constructs a system that implements the specification under this assumption. The constructed system sets g high whenever c is low and vice-versa.

Performance of Gist**.** Our implementation of reduction of $2\frac{1}{2}$-player games to 2-player games is linear time and efficient, and the computationally expensive step is solving 2-player games. For qualitative analysis of $2\frac{1}{2}$-player games, Gist can handle game graphs of size that can be typically handled by tools solving 2-player games. Typical run-times for qualitative analysis of $2\frac{1}{2}$-player parity games of various sizes are summarized in Table 1. The games used were generated using the benchmark tools of PGSolver and converting one-tenth fraction of the states into probabilistic states (further experimental results in [2]). For synthesis of environment assumptions, the expensive step is the reduction of LTL formula to deterministic parity automata. Our tool can handle formulas that are handled by classical tools for translation of LTL formula to deterministic parity automata.

Other features of Gist**.** Our tool is compatible with several other game solving and synthesis tools: Gist is compatible with PGSolver and GOAL. Our tool provides a graphical interface to describe games and automata, and thus can also be used as a front-end to PGSolver to graphically describe games.

References

1. K. Chatterjee. *Stochastic ω-Regular Games*. PhD thesis, UC Berkeley, 2007.
2. K. Chatterjee, T. Henzinger, B. Jobstmann, and A. Radhakrishna. Gist: A solver for probabilistic games. 2010.
3. K. Chatterjee, T. A. Henzinger, and B. Jobstmann. Environment assumptions for synthesis. In *CONCUR*, 2008.
4. K. Chatterjee, M. Jurdziński, and T.A. Henzinger. Quantitative stochastic parity games. In *SODA*, 2004.
5. Y. Gurevich and L. Harrington. Trees, automata, and games. In *STOC*, 1982.
6. M. Lange and O. Friedmann. The pgsolver collection of parity game solvers. Technical report, Institut für Informatik, Ludwig-Maximilians-Universität, 2009.
7. Y. Tsay, Y. Chen, M. Tsai, W. Chan, and C. Luo. Goal extended: Towards a research tool for omega automata and temporal logic. In *TACAS*, 2008.

Table 1. Runtimes for solving $2\frac{1}{2}$-player parity games

States	Edges	Runtime (sec.)		
		Avg.	Best	Worst
1000	5000	1.17	0.63	1.59
10000	50000	51.43	39.38	62.61
50000	250000	2513.18	2063.40	2711.23