Quantitative Verification and Synthesis

Barbara Jobstmann
CNRS-Verimag, Grenoble, France

Joint work with Roderick Bloem, Krishnendu Chatterjee, Tom Henzinger, and Rohit Singh.

Analysis – Verification – Synthesis

System Implementation Algorithm
“How”

Verification

Specification Properties Model
“What”

Analysis

Synthesis

Reactive Systems (e.g. servers, hardware,...)
- Interact with environment
- Non-terminating
- Finite data (or data abstractions)
- Control-oriented

Specifications
- Set of good behaviors = language
- Temporal logic (LTL)

Resource Controller

Client 1
r1
r2

Controller

Client 2

1) always(not(g1 and g2))
2) always(r1 implies eventually(g1))
3) always(r2 implies eventually(g2))
Classical Specifications

$\Sigma$ ... alphabet
$\Sigma^\omega$ ... set of all words/behaviors

$\varphi \subseteq \Sigma^\omega$
$\varphi : \Sigma^\omega \rightarrow \{0, 1\}$

Resource Controller

1) always(not($g_1$ and $g_2$))
2) always($r_1$ implies eventually($g_1$))
3) always($r_2$ implies eventually($g_2$))

Preference?

- Write more properties

S satisfies $\varphi$ iff $\forall w \in L(S) : \varphi(w) = 1$
Preference?

- Write more properties

1) Right properties might be hard to find
2) Properties usually difficult to express
3) Properties can over-constrain system

Quantitative Specifications

\[ \Sigma \ldots \text{alphabet} \]
\[ \Sigma^\omega \ldots \text{set of all words/behaviors} \]

\[ \varphi : \Sigma^\omega \rightarrow \text{Values} \]

Quantitative Specifications

\[ \Sigma \ldots \text{alphabet} \]
\[ \Sigma^\omega \ldots \text{set of all words/behaviors} \]

S better than S’ iff value(S) ≥ value(S’)

Two Main Questions

1. How to assign a value to a word?
2. How to assign a value to a system?
1. Value of a Word/Behavior

- **Idea:**
  - use weighted automata
  - give rewards for good behavior
- **Example:**
  - prefer fast reaction to request of client $i$

```plaintext
\[
\begin{align*}
&\text{Given word } w, \text{ let } s_0, s_1, s_2, \ldots \text{ be the run of } A \text{ on } w \\
&\text{Mean-payoff value} \\
&MP(w) = \lim_{n \to \infty} \sum_{i=0}^{n-1} r(s_i, s_{i+1})/n \\
&\text{Value } A(w) := MP(w)
\end{align*}
\]
```

2. Value of a System

- **Min/Max:**
  - Given a system $S$ with set of behaviors $L(S)$

```
\[
\text{Value}(S) = \min_{w \in L(S)} \varphi(w)
\]
```
2. Value of a System

\[ value(S) = \min_{w \in L(S)} \varphi(w) \]

How to Compute Value of System?

- Given a mean-payoff automaton A and a reactive system S, compute \( \text{value}(S) \)

System × Specification

\[ value(S) = \min_{w \in L(S)} \varphi(w) \]
How to Construct Optimal System?

- Given a qualitative specification $\varphi$ and a quantitative specification $\psi'$, construct a reactive system $S$ that
  (i) satisfies $\varphi$ and
  (ii) optimizes $\psi'$

Synthesis of Reactive Systems

1. **Qualitative Specification**
2. Construct two player game
3. Solve game
4. Construct system
5. Correct system

Synthesis

1. **Qualitative Specification**
2. Construct two player game
3. Solve game
4. Construct system
5. Correct system

Synthesis

1. **Safety**
2. Construct two player game
3. Solve game
4. Construct system

Synthesis

1. **Mean-payoff**
2. Construct two player game
3. Solve game
4. Construct system

[ Ehrenfeucht, Mycielski 1979 ]
Synthesis

Parity

Construct two-player game

Solve game

Construct system

Correct system

Optimal

Lexicographic Extension

- Combining quantitative specifications
- Weighted automata with tuples:
  - Natural numbers \( \text{value}_A : \Sigma^\omega \to R \)
  - Vectors \( \text{lvalue}_A : \Sigma^\omega \to R^d \)

\[
LMP(w) = \lim_{n \to \infty} \sum_{i=0}^{n-1} \frac{r(s_i, s_{i+1})}{n}
\]

Value of System (revisited)

- Recall, resource controller
- Aim: fast response, \( \text{value}_{A1} + \text{value}_{A2} \)
- Worse-case behavior?
- Worse-case input? \( (r_1 r_2)^\omega \)
- Best response? \( (g_1 g_2, \overline{g_1} g_2)^\omega \)
Resource Controllers

System × Specification

System × Specification

Differently worst-case behavior

System × Specification
### System × Specification

- In all states “equally frequently visited”
- Reward in every state: \( \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = 1.5 \)
- Average behavior: \( \frac{4 \cdot 1.5}{4} = 1.5 \)

### Average Over Input Sequences

- s0 is visited more often: s0: 2/3, s1: 2/9, s2: 1/9
- Rewards:
  - s0: \( \frac{3}{4} \cdot 2 + \frac{1}{4} \cdot 1 = 1.75 \)
  - s1,s2: \( \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = 1.5 \)
- Average behavior: 1.67

### Markov Chain – Input Distribution

- How likely are different input values?
How to Construct Optimal System?

- Given a qualitative specification $\varphi$ and a quantitative specification $\psi$, and a probabilistic environment assumption $\mu$, construct a reactive system $S$ that
  (i) satisfies $\varphi$ with probability 1 under $\mu$
  (ii) optimizes $\psi$ under $\mu$
Markov Decision Process

• Safety × mean-payoff × Markov chain = MDP

MDPs with mean-payoff parity

• Polynomial-time algorithm [CAV 2010]
• End-components: set of states that is strongly-connected and closed for probabilistic player
• Key observation:
  MP-parity value in an end component with even minimal priority is equal to MP value
  – Computes maximal end components with even minimal priority
  – Fix value for these states to corresponding MP value
  – Compute way to best end component

Synthesis

Construct two player game
Solve game
Construct system

Correct system
Optimal
Summary

• Quantities are good for verification & synthesis
  – to rank implementations wrt preference
  – to state “soft” properties

• Value of word
  – Mean-payoff automata (weighted automata that average over weights)
  – Lexicographic extension

• Value of system
  – Min/Max value over words: lexicographic MP-parity
  – Average value over words: MDP with MP-parity