From Boolean to Quantitative System Specifications

Tom Henzinger
EPFL
Outline

1 The Quantitative Agenda
2 Some Basic Open Problems
3 Some Promising Directions
The Boolean Agenda

Program/System  Property/Specification

Analysis

Yes/No
The Boolean Agenda

Program/ System  Property/ Specification

Analysis

Yes/No
- perhaps a proof
- perhaps some counterexamples
- perhaps even a proposed fix
The Boolean Agenda

Structure

Program/System

Property/Specification

Formula

Satisfaction Relation

Yes/No

- perhaps a proof
- perhaps some counterexamples
- perhaps even a proposed fix
The Boolean Agenda

Transition system.

Program/ System
Property/ Specification

Analysis

Every request is followed by a grant.

Yes/No
- perhaps a proof
- perhaps some counterexamples
- perhaps even a proposed fix
The Boolean Agenda

Timed automaton.

Quantitative Program/System

Quantitative Property/Specification

Every request is followed by a grant within 5 time units.

Analysis

Yes/No
- perhaps a proof
- perhaps some counterexamples
- perhaps even a proposed fix
The Boolean Agenda

Quantitative Program/System

Quantitative Property/Specification

Every request is followed by a grant within probability 1/2.

Markov process.

Analysis

Yes/No

- perhaps a proof
- perhaps some counterexamples
- perhaps even a proposed fix
The Boolean Agenda

Quantitative Program/System

Quantitative Property/Specification

Every request is followed by a grant within probability 1/2.

Analysis

B

- perhaps a proof
- perhaps some counterexamples
- perhaps even a proposed fix
The Quantitative Agenda

Quantitative Program/System

Quantitative Property/Specification

Analysis

R

-measure of “fit” between system and spec
-could be cost, quality, etc.
The Quantitative Agenda

Quantitative
Program/System

Quantitative
Property/Specification

Every request is followed by a grant.

Analysis

The less time between requests and grants, the better.

R
-measure of “fit” between system and spec
-could be cost, quality, etc.
Every request is followed by a grant.

The fewer unnecessary grants, the better.

- measure of “fit” between system and spec
- could be cost, quality, etc.
The Quantitative Agenda

Q1 Assigning values to behaviors
   Boolean case: correct vs. incorrect behaviors

Q2 Assigning values to systems/properties
   Boolean case: sets of behaviors (nondeterminism)

Q3 Assigning values to pairs of systems
   Boolean case: preorders on systems
Q1 Assigning Values To Behaviors

a. Probabilities

b. Resource use
   - worst case vs. average case (e.g. response time, QoS)
   - peak vs. accumulative (e.g. power consumption)

c. Quality measures
   - discounting vs. long-run averaging (e.g. reliability)
Q1 Assigning Values To Behaviors

a: ok
b: fail

Discounted value (0 < d < 1):

<table>
<thead>
<tr>
<th>Behavior</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>aaaaaaaaaaa...</td>
<td>1</td>
</tr>
<tr>
<td>aaaaaaab...</td>
<td>1 - d^8</td>
</tr>
<tr>
<td>aab...</td>
<td>1 - d^3</td>
</tr>
<tr>
<td>b...</td>
<td>0</td>
</tr>
</tbody>
</table>

Long-run average value:

<table>
<thead>
<tr>
<th>Behavior</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>aaaaaaaaaaa...</td>
<td>1</td>
</tr>
<tr>
<td>aaababaabaaab...</td>
<td>1 - 1/4</td>
</tr>
<tr>
<td>abaabaab...</td>
<td>1</td>
</tr>
<tr>
<td>babbabbbba...</td>
<td>0</td>
</tr>
</tbody>
</table>
Q2 Assigning Values To Systems

x: behaviors
w: observations
A,B: systems

\[ A(w) = \sup_{x} \{ \text{val}(x) : \text{obs}(x) = w \} \]
\[ B(w) = \exp_{x} \{ \text{val}(x) : \text{obs}(x) = w \} \]
Q2, Q3 Assigning Values To Systems

\[ x: \text{ behaviors} \]
\[ w: \text{ observations} \]
\[ A, B: \text{ systems} \]

\[ A(w) = \mathop{\sup}_x \{ \text{val}(x) : \text{obs}(x) = w \} \]
\[ B(w) = \mathop{\exp}_x \{ \text{val}(x) : \text{obs}(x) = w \} \]

\[ \text{diff}(A,B) = \mathop{\sup}_w \{ |A(w) - B(w)| \} \]

Compositionality: \[ \text{diff}(A || B, A' || B) \cdot f(\text{diff}(A, A')) \] [AFHMS].
Is there a Quantitative Framework with

- an appealing mathematical formulation,
- useful expressive power, and
- good algorithmic properties?

(Like the boolean theory of $\omega$-regularity.)
Outline

1 The Quantitative Agenda
2 Some Basic Open Problems
3 Some Promising Directions
Property = Language

Alphabet: \( \Sigma \)
\( \Sigma = \{a,b,c\} \)

Language: \( L \in \Sigma^\omega \)
\( L = (a^+b)^+(a^\omega[c^\omega][a^\omega b)^\omega \)

abaabaaaabcccccc... \( \notin L \)
abcabc... \( \notin L \)
Boolean Language

Alphabet: $\Sigma$
$\Sigma = \{a, b, c\}$

Language: $L \mu \Sigma^\omega$
$L = (a^+b)^+ (a^\omega[c^\omega][a^\omega b^\omega]$

abaabaaabcccccc... $\in L$
abcabc... $\notin L$

$L: \Sigma^\omega ! B$
Specification = Automaton

Q
\lambda: Q ! \Sigma
q_0 \in Q
\Gamma
\delta: Q \times \Gamma \rightarrow Q

A:

\Gamma = \{0, 1\}
$Q$ states

$\lambda: Q \rightarrow \Sigma$ labeling

$q_0 \in Q$ initial state

$\Gamma$ choices

$\delta: Q \times \Gamma \rightarrow Q$ transition function

$A$: 

$\Gamma = \{0,1\}$

$L(A) = (a+b)^+(a^\omega[c^\omega]) \cdot (a+b)^\omega$
Specification = Automaton

\[ Q \]
\[ \lambda: Q \to \Sigma \]
\[ q_0 \in Q \]
\[ \Gamma \]
\[ \delta: Q \times \Gamma \to Q \]

states
labeling
initial state
choices
transition function

A:

```
0 1 1
```

“scheduler” 0101111... ! aababccc... “outcome”
Specification = Automaton

\[ Q \]
\[ \lambda: Q \rightarrow \Sigma \] labeling
\[ q_0 \in Q \] initial state
\[ \Gamma \] choices
\[ \delta: Q \times \Gamma \rightarrow Q \] transition function

Scheduler: \[ x: Q^+ \rightarrow \Gamma \]
\[ S \ldots \text{set of schedulers} \]

Outcome: \[ f(x) = q_0q_1q_2 \ldots \]
\[ \text{where } 8 \ i : q_{i+1} = \delta(q_i, x(q_0 \ldots q_i)) \]

Language: \[ L = \{ \lambda(f(x)) : x \in \Gamma \} \]
Satisfaction = Language Inclusion

Given two automata A and B, is \( L(A) \subseteq L(B) \)?

i.e. \( \forall w \in \Sigma^\omega : L(A)(w) \cdot L(B)(w) \)
Satisfaction = Language Inclusion

Given two automata A and B, is $L(A) \subseteq L(B)$?

i.e. $\forall w \in \Sigma^\omega : L(A)(w) \cdot L(B)(w)$

For finite automata, PSPACE-complete.
Probabilistic Language

Word: element of $\Sigma^\omega$
Probabilistic Word: probability space on $\Sigma^\omega$
Probabilistic Language: set of probabilistic words

$w$: $ab\Sigma^\omega!1/2$
aab$\Sigma^\omega!1/4$
aaab$\Sigma^\omega!1/8$
...

Markov Decision Process

$Q$

$\lambda: Q \rightarrow \Sigma$

$q_0 \in Q$

$\Gamma$

$\delta: Q \times \Gamma \rightarrow D(Q)$

states

labeling

initial state

choices

transition function

A:
Markov Decision Process

Q states
λ: Q ! Σ labeling
q₀ 2 Q initial state
Γ choices
δ: Q × Γ → D(Q) transition function

A:

0101111... ! abccc... ! 1/2
aabccc... ! 1/4
...

0: 0.5
1: 1
0: 0.5
1: 1
Markov Decision Process

<table>
<thead>
<tr>
<th>Q</th>
<th>states</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda$: Q ! $\Sigma$</td>
<td>labeling</td>
</tr>
<tr>
<td>$q_0 \in Q$</td>
<td>initial state</td>
</tr>
<tr>
<td>$\Gamma$</td>
<td>choices</td>
</tr>
<tr>
<td>$\delta$: Q $\times$ $\Gamma$ $\rightarrow$ D(Q)</td>
<td>transition function</td>
</tr>
</tbody>
</table>

Pure scheduler: $x$: Q$^+$ $\rightarrow$ $\Gamma$

Probabilistic scheduler: $x$: Q$^+$ $\rightarrow$ D($\Gamma$)
Markov Decision Process

\begin{itemize}
  \item \( Q \) \hspace{2cm} \text{states}
  \item \( \lambda : Q \rightarrow \Sigma \) \hspace{2cm} \text{labeling}
  \item \( q_0 \in Q \) \hspace{2cm} \text{initial state}
  \item \( \Gamma \) \hspace{2cm} \text{choices}
  \item \( \delta : Q \times \Gamma \rightarrow D(Q) \) \hspace{2cm} \text{transition function}
\end{itemize}

\[
\begin{array}{c}
A:
\begin{array}{c}
\text{a}
\end{array}
\begin{array}{c}
\text{b}
\end{array}
\begin{array}{c}
\text{c}
\end{array}
\end{array}
\]

\[
\begin{array}{c}
\text{0: 0.5}
\end{array}
\begin{array}{c}
\text{0: 0.5}
\end{array}
\begin{array}{c}
\text{0,1}
\end{array}
\]

\[
\begin{array}{c}
\text{1: 1}
\end{array}
\begin{array}{c}
\text{1: 1}
\end{array}
\begin{array}{c}
\text{0: 0.5}
\end{array}
\]

\[
\{0: 0.5, 1: 0.5\}^\omega \quad ! \quad abccc... \quad ! \quad 9/16
\]

\[
aabccc... \quad ! \quad 9/64
\]

\[
...\]
Probabilistic Language Inclusion

Given two MDPs A and B, is $L(A) \subseteq L(B)$?
Probabilistic Language Inclusion

Given two MDPs $A$ and $B$, is $L(A) \subseteq L(B)$?
Probabilistic Language Inclusion

Given two MDPs A and B, is $L(A) \subseteq L(B)$?

? 

Open even if specification B is deterministic (i.e. $|\Gamma| = 1$) and implementation scheduler required to be pure.

If both sides are deterministic, then it can be solved in polynomial time (equivalence of Rabin’s probabilistic automata) [Tzeng, DHR].
Quantitative Language

Language: \( L: \sum^\omega ! \ B \)

Quantitative Language: \( L: \sum^\omega ! \ R \)

\[ L(ab^\omega) = \frac{1}{2} \]
\[ L(aab^\omega) = \frac{1}{4} \]
\[ L(aaab^\omega) = \frac{1}{8} \]
...
Weighted Automaton

$Q$  
$\lambda: Q \rightarrow \Sigma$  
$q_0 \in Q$  
$\Gamma$  
$\delta: Q \times \Gamma \rightarrow R$  

states  
labeling  
initial state  
choices  
transition function

A:

\[\begin{align*}
0; 4 & \rightarrow a \\
1; 2 & \rightarrow b \\
0; 0 & \rightarrow b \\
& \rightarrow c
\end{align*}\]
A: Weighted Automaton

\[ Q, \lambda: Q \rightarrow \Sigma, q_0 \in Q, \Gamma, \delta: Q \times \Gamma \rightarrow \mathbb{R} \leq Q \]

states
labeling
initial state
choices
transition function

Max value:

<table>
<thead>
<tr>
<th>String</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>0101111...</td>
<td>4</td>
</tr>
<tr>
<td>aababccc...</td>
<td>4</td>
</tr>
<tr>
<td>1111111...</td>
<td>2</td>
</tr>
<tr>
<td>abccc...</td>
<td>2</td>
</tr>
</tbody>
</table>
Weighted Automaton

\[ Q \]
\[ \lambda: Q \rightarrow \Sigma \] labeling
\[ q_0 \in Q \] initial state
\[ \Gamma \] choices
\[ \delta: Q \times \Gamma \rightarrow R \leq Q \] transition function

Outcome:
\[ f(x) = q_0 v_1 q_1 v_2 q_2 \ldots \]
where \( 8 \ i : (v_{i+1}, q_{i+1}) = \delta(q_i, x(q_0 \ldots q_i)) \)

Max value:
\[ \text{val}(q_0 v_1 q_1 v_2 q_2 \ldots) = \sup\{ v_i : i \geq 1 \} \]
**Weighted Automaton**

- **Q**: states
- **λ**: Q \[\rightarrow\] \(\Sigma\): labeling
- **q_0**: initial state
- **Γ**: choices
- **δ**: \(Q \times Γ \rightarrow \mathbb{R} \subseteq Q\): transition function

**Outcome:**
\[f(x) = q_0 v_1 q_1 v_2 q_2 ...\]
where \(\delta(q_i, x(q_0...q_i)) = (v_{i+1}, q_{i+1})\)

**Max value:**
\[\text{val}(q_0 v_1 q_1 v_2 q_2 ...) = \sup\{ v_i : i \geq 1 \}\]

**q-Language:**
\[L(w) = \sup\{ \text{val}(f(x)) : x \in S \text{ s.t. } \lambda(f(x)) = w \}\]
Different Value Functions

Max value: \( \text{val}(q_0v_1q_1v_2q_2...) = \sup\{v_i : i \geq 1 \} \)  
(only 0 and 1 costs: finite automaton)

Limsup value: \( \text{val} = \lim_{n\to\infty} \sup\{v_i : i \geq n \} \)  
(only 0 and 1 costs: Buechi automaton)
Different Value Functions

Max value: \[ \text{val}(q_0v_1q_1v_2q_2...) = \sup \{ v_i : i \geq 1 \} \]
(only 0 and 1 costs: finite automaton)

Limsup value: \[ \text{val} = \lim_{n \to 1} \sup \{ v_i : i \geq n \} \]
(only 0 and 1 costs: Buechi automaton)

Limavg value: \[ \text{val} = \lim_{n \to 1} \frac{1}{n} \sum_{1 \leq i \leq n} v_i \]
Different Value Functions

Max value: \( \text{val}(q_0v_1q_1v_2q_2...) = \sup \{ v_i : i \geq 1 \} \)
(only 0 and 1 costs: finite automaton)

Limsup value: \( \text{val} = \lim_{n \to \infty} \sup \{ v_i : i \geq n \} \)
(only 0 and 1 costs: Buechi automaton)

Limavg value: \( \text{val} = \lim_{n \to \infty} \frac{1}{n} \sum_{1 \leq i \leq n} v_i \)

Discounted: \( \text{val} = \sum_{i \geq 1} d^i \mathcal{C} v_i \) for some \( 0 < d < 1 \)
Weighted Automaton

Limsup value: 01010101... ! aabababab...; 2
              11111111... ! abccc...; 0

Limavg value: 01010101... ! aabababab...; 1
              11111111... ! abccc...; 0

Discounted: 01010101... ! aabababab...; 2.66...
             11111111... ! abccc...; 1.25

(d = 0.5)
Quantitative Language Inclusion

Given two weighted automata A and B, is
\[ \forall w \in \Sigma^\omega : L(A)(w) \cdot L(B)(w) ? \]
Quantitative Language Inclusion

Given two weighted automata A and B, is $8 w \in \Sigma^\omega : L(A)(w) \cdot L(B)(w) \ ?$

For max and limsup values: PSPACE. For limavg and discounted values: Open.
Quantitative Language Inclusion

Given two weighted automata A and B, is
8 w 2 Σω : L(A)(w) · L(B)(w) ?

For max and limsup values: PSPACE. For
limavg and discounted values: Open.

If specification B is deterministic,
then it can be solved in polynomial time [CDH].
A not simulated by B.

Simulation game solvable in P for max, and in NP \textasciitilde coNP for limsup, limavg, discounted [CDH].
Quantitative Emptiness and Universality

Emptiness: Given a weighted automaton $A$, is $L(A)(w) \geq 1$ for some word $w \in \Sigma^\omega$?

In P for max, limsup, limavg, and discounted automata. Solvable by finding a path with maximal value [CDH].
Quantitative Emptiness and Universality

**Emptiness:** Given a weighted automaton $A$, is $L(A)(w) \geq 1$ for some word $w \in \Sigma^\omega$?

In $P$ for max, limsup, limavg, and discounted automata. Solvable by finding a path with maximal value [CDH].

**Universality:** Given a weighted automaton $A$, is $L(A)(w) \geq 1$ for all words $w \in \Sigma^\omega$?

As hard as language inclusion.
Quantitative Expressiveness

[CDH CSL08, LICS09]
E.g. limavg automata not determinizable:

\[ \Sigma^* b^\omega \] expressible by a nondeterministic limavg automaton.

\[ \Sigma^* b^\omega \] not expressible by a deterministic limavg automaton.

Every b-cycle would need weight 1.
Consider \( w_n = (ab^n)^\omega \).
Then \( \text{val}(w_n) = 1 \) for sufficiently large \( n \), but \( w_n \notin \Sigma^* b^\omega \).
### Quantitative Closure Properties

<table>
<thead>
<tr>
<th></th>
<th>$\cup$</th>
<th>$\cap$</th>
<th>$\Sigma^\omega \setminus L$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathbb{P} \text{ Sup}$</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>$\mathbb{P} \text{ LimInf}$</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>$\text{DLimSup}$</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>$\text{NLimSup}$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$\text{DLimAvg}$</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>$\text{NLimAvg}$</td>
<td>✓</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>$\text{DDisc}$</td>
<td>×</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>$\text{NDisc}$</td>
<td>✓</td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>
Quantitative Closure Properties

E.g. limavg automata not closed under min:

\[ \text{min}(L_1, L_2) \] not expressible by a limavg automaton.

Consider \( w_n = (a^n b^n)^\omega \) for large \( n \).
Some a-cycle or b-cycle would need average positive weight.
Then some word \( u a^\omega \) or \( u b^\omega \) would have a positive value.
Outline

1 The Quantitative Agenda
2 Some Basic Open Problems
3 Some Promising Directions
The Boolean Agenda

System Specification

Analysis

Yes/No
The Boolean Agenda

Specification

Synthesis

Correct System
The Boolean Agenda

$\omega$-Regular Automaton

Graph Game with $\omega$-Regular Objective

Correct System = Winning Strategy
3.1 Quantitative Synthesis

Quantitative Specification → Synthesis → Optimal System
3.1 Quantitative Synthesis

Weighted Automaton

Graph Game with Quantitative Objective

Optimal System = Optimal Strategy
3.1 Quantitative Synthesis

Weighted Automaton

Graph Game with Quantitative Objective

Optimal System = Optimal Strategy

- positional vs. finite-memory vs. unrestricted strategies
- optimal vs. \( \varepsilon \)-optimal strategies
1 Constrained Resources

-every weight is a resource cost (e.g. power consumption)
-optimise peak resource use: max objective
-optimise accumulative resource use: sum objective
[Chakrabarti et al.]
1 Constrained Resources

2 Preference between Different Implementations

- boolean spec, but certain implementations preferred
- formalized using lexicographic objectives
[Jobstmann et al.]

\[ h f, g_1, \ldots, g_n i \]

boolean objective \hspace{1cm} quantitative objectives
Following a request, all steps until the next grant are penalized.
Following a request, all repeated grants are penalized.
3.2 Robust Systems

1 Robustness as Mathematical Continuity:
- small input changes should cause small output changes
- only possible in a quantitative framework

\[ \varepsilon > 0, \delta > 0, \text{input-change} \cdot \delta \text{ ) output-change} \cdot \varepsilon \]
In general programs are not continuous. But they can less continuous:

read sensor value $x$;
if $x \cdot c$ then $y = f_1(x)$
else $y = f_2(x)$;
In general programs are not continuous. But they can be less continuous:

```plaintext
read sensor value x;
if x \cdot c then y = f_1(x)
else y = f_2(x);
```

Or more continuous:

```plaintext
if x \cdot c - \epsilon then y = f_1(x);
if x \cdot c + \epsilon then y = f_2(x)
else y = (f_2(c + \epsilon) - f_1(c - \epsilon))(x - c + \epsilon)/2\epsilon + f_1(c - \epsilon);
```

[Majumdar et al., Gulwani et al.]
3.2 Robust Systems

1 Robustness as Mathematical Continuity:
   - small input changes should cause small output changes
   - only possible in a quantitative framework

\[ \varepsilon > 0, \delta > 0, \text{input-change} \cdot \delta \Rightarrow \text{output-change} \cdot \varepsilon \]

Example of a Robustness Theorem [AHM]:
If \( \text{discountedBisimilarity}(A, B) > 1 - \varepsilon \),
then \( \exists w : |A(w) - B(w)| < f(\varepsilon) \).
3.2 Robust Systems

1 Robustness as Mathematical Continuity:
   - small input changes should cause small output changes
   - only possible in a quantitative framework

2 Robustness w.r.t. Faulty Assumptions:
   - environment may violate assumptions
   - few environment mistakes should cause few system mistakes
   - ratio of system to environment mistakes as quantitative quality measure

[Greimel et al.]
- Component interfaces expose resource requirements (e.g. time, memory, power).

- Interfaces are compatible if their combined requirements do not exceed the available resources.

- If the requirements are dynamic, then compatibility can be solved as a graph game with quantitative objectives.

[Chakrabarti et al.]
Max Constraint

minimizer
maximizer

node limit = 20
Max Constraint

minimizer

maximizer

node limit = 20

A

B

C

D

E

F

G

H

2

99

5

59

15

19

5
Sum Constraint

path limit = 10

minimizer
maximizer
3.4 System Reliability

-assuming x% of periodic input values are valid, y% of periodic output values should be valid

-hardware faulty, but can be replicated

-compiler ensures specified reliability through replication

[Ghosal et al.]
3.4 System Reliability

**a**: ok  
**b**: fail

Limit-average value:

- `aaaaaaaaaa...` 1
- `aaabaaabaab...` 3/4
- `ababbabbb...` 0

Want reliability of $1 - 10^{-x}$. 
Conclusions

- “Quantitative” is more than “timed” and “probabilistic.”
- We need to move from boolean correctness criteria to quantitative system preference metrics.
- We have interesting point solutions, but no convincing overall framework.