Modeling, Analysis, and Synthesis
of
Quantitative System Requirements

Tom Henzinger
IST Austria

Joint work with Krishnendu Chatterjee and Laurent Doyen.
Outline

1 A Quantitative Systems Theory
2 Some Basic Open Problems
3 Some Promising Directions
Boolean Systems Theories

Program/ System  Property/ Specification

Analysis

Yes/No
Boolean Systems Theories

Program/ System  Property/ Specification

Analysis

Yes/No
- perhaps a proof
- perhaps some counterexamples
- perhaps even a proposed fix
Boolean Systems Theories

Structure

Program/ System

Property/ Specification

Formula

Satisfaction Relation

Yes/No
- perhaps a proof
- perhaps some counterexamples
- perhaps even a proposed fix
Boolean Systems Theories

Transition system.

Program/System

Property/Specification

Every request is followed by a grant.

Analysis

Yes/No

-perhaps a proof
-perhaps some counterexamples
-perhaps even a proposed fix
Boolean Systems Theories

Timed automaton.

Quantitative Program/System

Quantitative Property/Specification

Analysis

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

Yes/No
- perhaps a proof
- perhaps some counterexamples
- perhaps even a proposed fix
Boolean Systems Theories

Markov process.

Quantitative Program/System

Quantitative Property/Specification

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

Analysis

Yes/No

- perhaps a proof
- perhaps some counterexamples
- perhaps even a proposed fix
Boolean Systems Theories

- Markov process.
- Every request is followed by a grant within probability 1/2.
- Quantitative Program/System
- Quantitative Property/Specification
- Analysis
  - perhaps a proof
  - perhaps some counterexamples
  - perhaps even a proposed fix
A Quantitative Systems Theory

Quantitative Program/System

Quantitative Property/Specification

Analysis

\( R \)

- measure of “fit” between system and spec
- could be cost, quality, etc.
A Quantitative Systems Theory

Quantitative Program/System  Quantitative Property/Specification

Analysis

Every request is followed by a grant.

The less time between requests and grants, the better.

- measure of “fit” between system and spec
- could be cost, quality, etc.
A Quantitative Systems Theory

Quantitative Property/Specification

Analysis

Every request is followed by a grant.

The fewer unnecessary grants, the better.

R

- measure of “fit” between system and spec
  - could be cost, quality, etc.
A Quantitative Systems Theory

Q1 Assigning values to behaviors

Q2 Assigning values to systems/properties

Q3 Assigning values to pairs of systems/properties
A Quantitative Systems Theory

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/properties
   Boolean case: preorders (refinement)
Boolean Systems Theories

P₁  P₂  P₃

S₁  S₁'  S₂  S₂'  S₂''
Boolean Systems Theories
Boolean Systems Theories

S_1 \quad S'_1 \quad S_2 \quad S'_2 \quad S''_2
A Quantitative Systems Theory
A Quantitative Systems Theory
A Quantitative Systems Theory

![Diagram showing relationships between P₁, P₂, P₃, S₁, S₁', S₂, S₂', S₂'']

- P₁ connects to S₁ and S₁' with weights 0.9 and 0.5, respectively.
- P₁ connects to P₂ with a weight of 0.8.
- P₂ connects to S₂ with a weight of 0.7.
- P₂ connects to S₂' with a weight of 0.2.
- P₃ is connected to S₂' without a weight indicated.

Weights: 0.9, 0.5, 0.8, 0.7, 0.2
Q1 Assigning Values To Behaviors

a. Probabilities
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)
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: Safety

a: ok
b: fail

Discounted value \((0 < d < 1)\):

- aaaaaa... \(1\)
- aaaaaaab... \(1 - d^8\)
- aab... \(1 - d^3\)
- b... \(0\)
Q1 Assigning Values To Behaviors: Safety

a: ok
b: fail

Discounted value (0 < d < 1):

- aaaaaaaaaaa... \( \text{1} \)
- aaaaaaaab... \( 1 - d^8 \)
- aab... \( 1 - d^3 \)
- b... \( 0 \)

Long-run average value:

- aaaaaaaaaaa... \( 1 \)
- abaabaaab... \( 1 \)
- aaabaaabaaab... \( 3/4 \)
- babbabbbba... \( 0 \)
Q2  Assigning Values To Systems

x: behaviors
w: observations (infinite words)
A,B: systems

\[ A(w) = \sup_x \{ \text{val}(x) : \text{obs}(x) = w \} \]
Q2 Assigning Values To Systems

x: behaviors
w: observations (infinite words)
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 Assigning Values To Systems

x: behaviors
w: observations (infinite words)
A,B: systems

\[ A(w) = \sup_x \{ \text{val}(x) : \text{obs}(x) = w \} \]
\[ B(w) = \exp_x \{ \text{val}(x) : \text{obs}(x) = w \} \]

relative to input distribution
Q3 Assigning Distances To Systems

x: behaviors
w: observations (infinite words)
A,B: systems

\[ A(w) = \sup_x \{ \text{val}(x): \text{obs}(x) = w \} \]
\[ B(w) = \exp_x \{ \text{val}(x): \text{obs}(x) = w \} \]
\[ \text{diff}(A,B) = \sup_w \{ |A(w) - B(w)| \} \]
Q3 Assigning Distances To Systems

x: behaviors
w: observations (infinite words)
A,B: systems

\[ A(w) = \sup_x \{ \text{val}(x) : \text{obs}(x) = w \} \]
\[ B(w) = \exp_x \{ \text{val}(x) : \text{obs}(x) = w \} \]

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

Boolean compositionality: if \( A \leq A' \) then \( A || B \leq A'||B \)
Quantitative compositionality: \( \text{diff}(A||B,A'||B) \leq f(\text{diff}(A,A')) \) [AFHMS]
Is there a Quantitative Systems Theory with
  - an appealing mathematical formulation,
  - useful expressive power, and
  - good algorithmic properties?

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

1 A Quantitative Systems Theory

2 Some Basic Open Problems:
   - Language inclusion for MDPs
   - Language inclusion for weighted automata

3 Some Promising Directions
Property = Language

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

Language: \( L \subseteq \Sigma^\omega \)
\( L = (a^+b)^+(a^\omega \cup c^\omega) \cup (a^+b)^\omega \)
abaabaaabcccccc... \( \in L \)
abcabc... \( \notin L \)
Boolean Language

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

Language: \( L \subseteq \Sigma^\omega \)
\( L = (a^*b)^+(a^\omega \cup c^\omega) \cup (a^*b)^\omega \)

abaabaaabcccccc... \( \in L \)
abcabc... \( \notin L \)

\( L : \Sigma^\omega \rightarrow \mathbb{B} \)
Specification = Automaton

\( Q \)
\( \lambda: Q \rightarrow \Sigma \)
\( q_0 \in Q \)
\( \Gamma \)
\( \delta: Q \times \Gamma \rightarrow Q \)

states
labeling
initial state
choices
transition function

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

- $Q$: states
- $\lambda: Q \to \Sigma$: labeling
- $q_0 \in Q$: initial state
- $\Gamma$: choices
- $\delta: Q \times \Gamma \to Q$: transition function

Graph:

- States: $a$, $b$, $c$
- Transitions:
  - From $a$ to $b$ on $1$
  - From $a$ to $a$ on $0$
  - From $b$ to $c$ on $1$
  - From $b$ to $b$ on $1$

“scheduler” $010111... \to$ aababccc... “outcome”
Specification = Automaton

Q \quad \text{states}
\lambda: Q \rightarrow \Sigma \quad \text{labeling}
q_0 \in Q \quad \text{initial state}
\Gamma \quad \text{choices}
\delta: Q \times \Gamma \rightarrow Q \quad \text{transition function}

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

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

Language: \quad L = \{ \lambda(f(x)) : x \in S \}
Satisfaction = Language Inclusion

Given two automata A and B, is $L(A) \subseteq L(B)$?
Satisfaction = Language Inclusion

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

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

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

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

For finite/Büchi automata, PSPACE-complete.
Probabilistic Language

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

\[
w: \begin{align*}
ab\Sigma^\omega & \rightarrow 1/2 \\
aab\Sigma^\omega & \rightarrow 1/4 \\
aaab\Sigma^\omega & \rightarrow 1/8 \\
... 
\end{align*}
\]
Markov Decision Process

Q states
\( \lambda: Q \rightarrow \Sigma \) labeling
q_0 \in Q initial state
\Gamma choices
\( \delta: Q \times \Gamma \rightarrow D(Q) \) transition function

A:

```
A: 0: 0.5
   ↓
 a 0: 0.5 1: 1
   ↓
 b 0: 0.5
   ↓
 c 0,1
```

Markov Decision Process

- $Q$: states
- $\lambda: Q \rightarrow \Sigma$: labeling
- $q_0 \in Q$: initial state
- $\Gamma$: choices
- $\delta: Q \times \Gamma \rightarrow D(Q)$: transition function

Example:

```
0101111... \rightarrow abccc... \rightarrow 1/2
aabccc... \rightarrow 1/4
...```

Diagram:

- States: a, b, c
- Actions: 0, 1
- Probabilities:
  - $a$: 1
  - $b$: 0.5, 1
  - $c$: 0.5, 0.5
Markov Decision Process

<table>
<thead>
<tr>
<th>Q</th>
<th>states</th>
</tr>
</thead>
<tbody>
<tr>
<td>λ: Q → Σ</td>
<td>labeling</td>
</tr>
<tr>
<td>q₀ ∈ Q</td>
<td>initial state</td>
</tr>
<tr>
<td>Γ</td>
<td>choices</td>
</tr>
<tr>
<td>δ: Q × Γ → D(Q)</td>
<td>transition function</td>
</tr>
</tbody>
</table>

Pure scheduler:  

x: Q⁺ → Γ

Probabilistic scheduler:  

x: Q⁺ → D(Γ)
Markov Decision Process

Q  states
\lambda: Q \rightarrow \Sigma  labeling
q_0 \in Q  initial state
\Gamma  choices
\delta: Q \times \Gamma \rightarrow D(Q)  transition function

A:

{0: 0.5, 1: 0.5}^\omega \rightarrow \text{abccc...} \rightarrow 9/16
\text{aabccc...} \rightarrow 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 \( L(A) \subseteq L(B) \)?
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: $L: \Sigma^\omega \rightarrow \mathbb{B}$

Quantitative Language: $L: \Sigma^\omega \rightarrow \mathbb{R}$

$L(ab^\omega) = 1/2$
$L(aab^\omega) = 1/4$
$L(aaab^\omega) = 1/8$
...

A weighted automaton is defined by the following components:

- **States** \( Q \)
- **Labeling** \( \lambda : Q \to \Sigma \)
- **Initial State** \( q_0 \in Q \)
- **Choices** \( \Gamma \)
- **Transition Function** \( \delta : Q \times \Gamma \to \mathbb{R} \times Q \)

The diagram visualizes the transitions with labels and weights:

- From state **a**:
  - Transition to **b** with label 1; 2, weight 4
  - Transition to **c** with label 0, weight 0

- From state **b**:
  - Transition to **c** with label 1; 1

- From state **c**:
  - Transition to **b** with label 0, weight 0
  - Transition to **a** with label 0, weight 0

- Initial state **b** with label 0; 0, weight 1; 1

- Final states are represented by circles with incoming arrows and labels.

The diagram illustrates the directed graph of the automaton with weighted edges and transitions.
Weighted Automaton

Q: states
\lambda: Q \rightarrow \Sigma: labeling
q_0 \in Q: initial state
\Gamma: choices
\delta: Q \times \Gamma \rightarrow \mathbb{R} \times Q: transition function

A: 0; 4
0; 0
1; 2
1; 1
0,1; 0

Value:
0101111... \rightarrow aababccc...; 4
1111111... \rightarrow abccc...; 2
Different Value Functions

Max value: \[ \text{val}(q_0v_1q_1v_2q_2... ) = \sup \{ v_i : i \geq 1 \} \]

Limsup value: \[ \text{val} = \lim_{n \to \infty} \sup \{ v_i : i \geq n \} \]
Different Value Functions

Max value: \( \text{val}(q_0 v_1 q_1 v_2 q_2 \ldots) = \sup \{ v_i : i \geq 1 \} \)
(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: \[ \text{val} = \lim_{n \to \infty} \sup \{ v_i : i \geq n \} \]
(only 0 and 1 costs: Büchi automaton)

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

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

Limsup value:
- $01010101... \rightarrow \text{aabababab}...; 2$
- $11111111... \rightarrow \text{abccc}...; 0$

Limavg value:
- $01010101... \rightarrow \text{aabababab}...; 1$
- $11111111... \rightarrow \text{abccc}...; 0$

Discounted: $(d = 0.5)$
- $01010101... \rightarrow \text{aabababab}...; 2.66...$
- $11111111... \rightarrow \text{abccc}...; 1.25$
Quantitative Language Inclusion

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

Given two weighted automata $A$ and $B$, is
\[ \forall w \in \Sigma^\omega : L(A)(w) \leq L(B)(w) \, ? \]

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

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

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].
Quantitative Simulation

A:

B:
Quantitative Simulation

A: 

\[ \begin{array}{c}
1 \\
\downarrow \\
a \\
1 \\
\downarrow \\
b \\
1 \\
\downarrow \\
1 \\
\end{array} \]

\[ \leq \]

B: 

\[ \begin{array}{c}
2 \\
\downarrow \\
b \\
2 \\
\downarrow \\
a \\
0 \\
\downarrow \\
0 \\
\downarrow \\
b \\
2 \\
\downarrow \\
a \\
\end{array} \]

A not simulated by B.
Quantitative Simulation

A: 

B: 

A not simulated by B.

Simulation game solvable in P for max values; in NP $\cap$ coNP for limsup, limavg, discounted values [CDH].
E.g. limavg automata not determinizable [CDH]:

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

E.g. limavg automata not determinizable [CDH]:

$\Sigma^*b^\omega$ expressible by a nondeterministic 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

E.g. limavg automata not closed under min [CDH]:
E.g. limavg automata not closed under \( \min \) [CDH]:

\begin{align*}
L_1: & & 0 & & 1 \\
& \xrightarrow{a} & 1 & \xrightarrow{b} & 0 \\
L_2: & & 1 & & 0 \\
& \xrightarrow{a} & 0 & \xrightarrow{b} & 1 \\
\end{align*}

\( \min(L_1,L_2) \) 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 Verification Agenda
2 Some Basic Open Problems:
   - Language inclusion for MDPs
   - Language inclusion for weighted automata
3 Some Promising Directions
Outline

1 The Quantitative Verification Agenda
2 Some Basic Open Problems
3 Some Promising Directions:
   - Quantitative Synthesis
   - Robust Systems
Boolean Systems Theories

System Specification

Analysis

Yes/No
Boolean Systems Theories

Specification

Synthesis

Correct System
Boolean Systems Theories

ω-Regular Automaton

Graph Game with ω-Regular Objective

Correct System = Winning Strategy
Quantitative Synthesis

Quantitative Specification

Synthesis

Optimal System
Quantitative Synthesis

Weighted Automaton

Graph Game with Quantitative Objective

Optimal System = Optimal Strategy
Automaton states are partitioned into min and max states.

Game: minimizer against maximizer

- in min states, minimizer chooses successor
- in max states, maximizer chooses successor

- minimizer tries to minimize value of a word
- maximizer tries to maximize value of a word

Scheduler is replaced by two strategies, one for the minimizer and one for the maximizer:

\[ L(w) = \sup \inf \ldots \]
1 Constrained Resources

- every weight is a resource cost (e.g. power consumption)
- optimize peak resource use: max objective
- optimize accumulative resource use: sum objective

[Chakrabarti et al.]
Max Game

minimizer
maximizer
Max Game

value = 15

minimizer
maximizer

A

B

C

D

E

F

G

H

99

59

5

15

19

5

9

2
minimizer
maximizer

Sum Game
Sum Game

value = 9

minimizer
maximizer

B
99

C
5

D
59

E
15

F
19

G
9

H
-9
Sum Game

minimizer
maximizer
Sum Game

C: 5 + max(0, min(59, 99)) = 64
E: $15 + \max(0, \max(9, -9)) = 24$
Sum Game

minimizer
maximizer

Iteration 1.
Sum Game

minimizer
maximizer

Iteration 2.
Sum Game

minimizer
maximizer

Iteration 3.
Sum Game

Iteration 4 = fixpoint.
1 Constrained Resources

2 Preference between Different Implementations
   - boolean spec, but certain implementations preferred
   - formalized using lexicographic objectives
   [Jobstmann et al.]

\[ \langle f, g_1, \ldots, g_n \rangle \]

\[ \uparrow \quad \uparrow \]

boolean objective  quantitative objectives
Following a request, all steps until the next grant are penalized.
Following a request, all repeated grants are penalized.
Outline

1 The Quantitative Verification Agenda
2 Some Basic Open Problems
3 Some Promising Directions:
   - Quantitative Synthesis
   - Robust Systems
1 Robustness as Mathematical Continuity:

- small input changes should cause small output changes
- only possible in a quantitative framework

∀ ε>0. ∃ δ>0. input-change ≤ δ ⇒ output-change ≤ ε
In general programs are not continuous. But they can less continuous:

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

read sensor value x;
if $x \leq c$ then $y = f_1(x)$
else $y = f_2(x)$;

Or more continuous:

if $x \leq c - \varepsilon$ then $y = f_1(x)$;
if $x \geq c + \varepsilon$ then $y = f_2(x)$
else $y = (f_2(c+\varepsilon)-f_1(c-\varepsilon))(x-c+\varepsilon)/2\varepsilon + f_1(c-\varepsilon)$;

[Majumdar et at., Gulwani et al.]
1 Robustness as Mathematical Continuity:

- small input changes should cause small output changes
- only possible in a quantitative framework

\[ \forall \varepsilon > 0. \ \exists \delta > 0. \text{ input-change } \leq \delta \Rightarrow \text{ output-change } \leq \varepsilon \]

Example of a Robustness Theorem [AHM]:

If discounted\textbf{B}isimilarity\textbf{(}A,B\textbf{)} \(> 1 - \varepsilon,\)
then \(\forall w : |A(w) - B(w)| < f(\varepsilon).\)
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.]
Conclusions

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