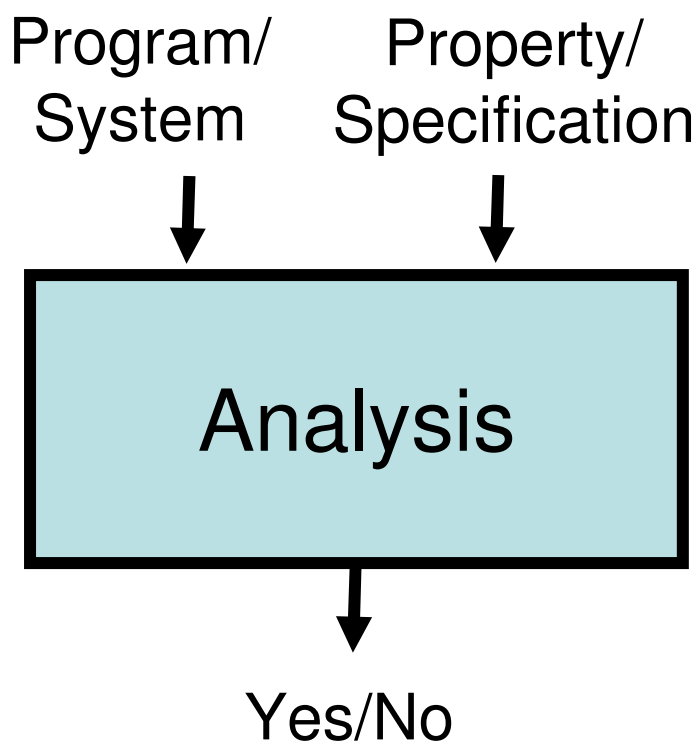
From Boolean to Quantitative System Specifications

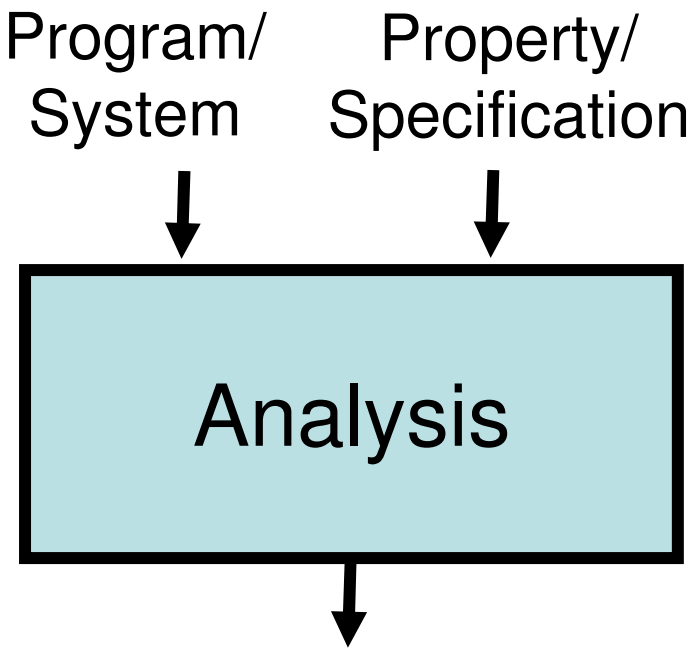
Tom Henzinger
EPFL

Outline

The Boolean Agenda



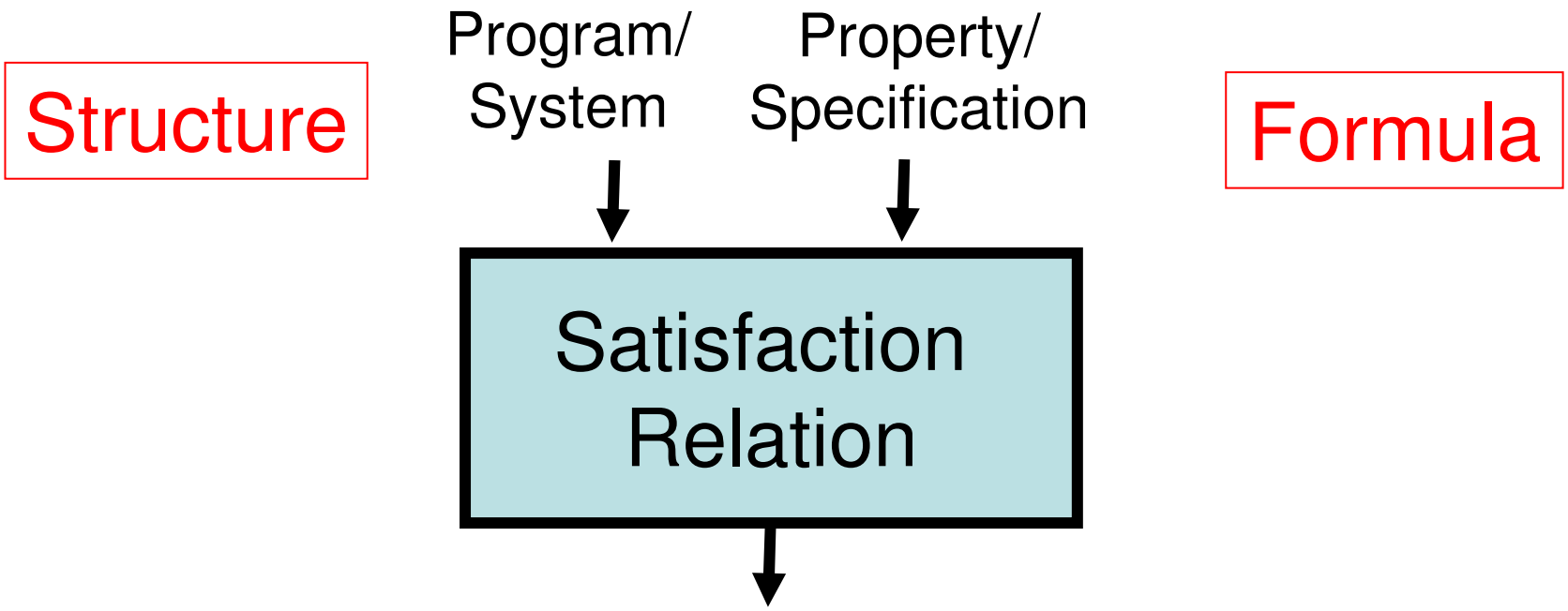
The Boolean Agenda



Yes/No

-perhaps a proof
-perhaps some counterexamples
-perhaps even a proposed fix

The Boolean Agenda



Yes/No

-perhaps a proof
-perhaps some counterexamples
-perhaps even a proposed fix

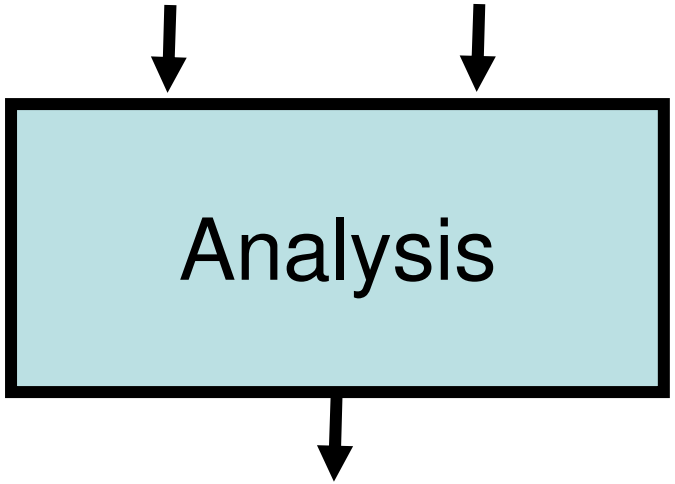
The Boolean Agenda

Transition system.

Program/ System

Property/ Specification

Every request is followed by a grant.



Yes/No

-perhaps a proof
-perhaps some counterexamples
-perhaps even a proposed fix

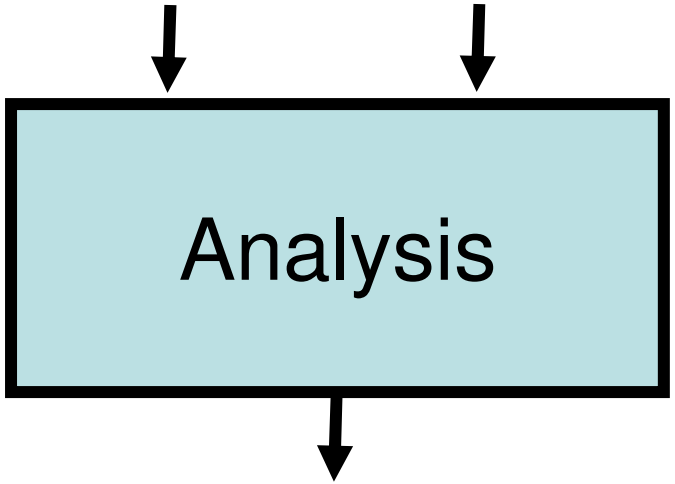
The Boolean Agenda

Timed automaton.

Quantitative Program/ System

Quantitative Property/ Specification

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



Yes/No

-perhaps a proof
-perhaps some counterexamples
-perhaps even a proposed fix

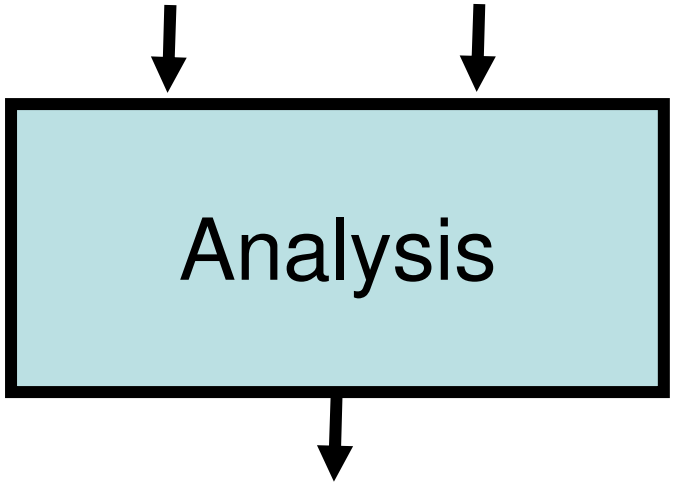
The Boolean Agenda

Markov process.

Quantitative Program/ System

Quantitative Property/ Specification

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



Yes/No

-perhaps a proof
-perhaps some counterexamples
-perhaps even a proposed fix

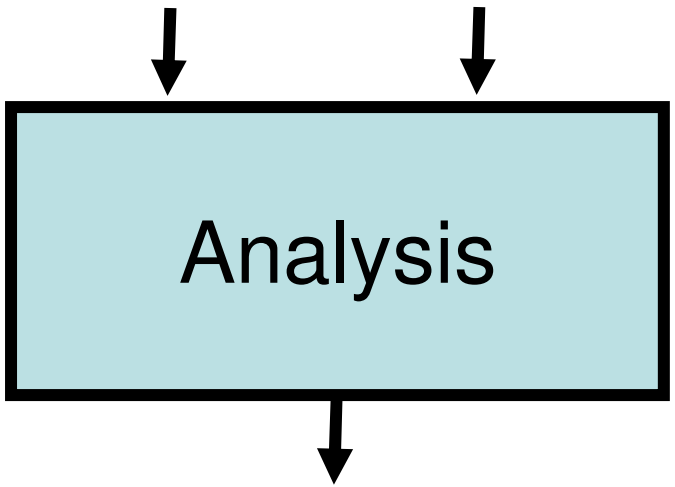
The Boolean Agenda

Markov process.

Quantitative Program/ System

Quantitative Property/ Specification

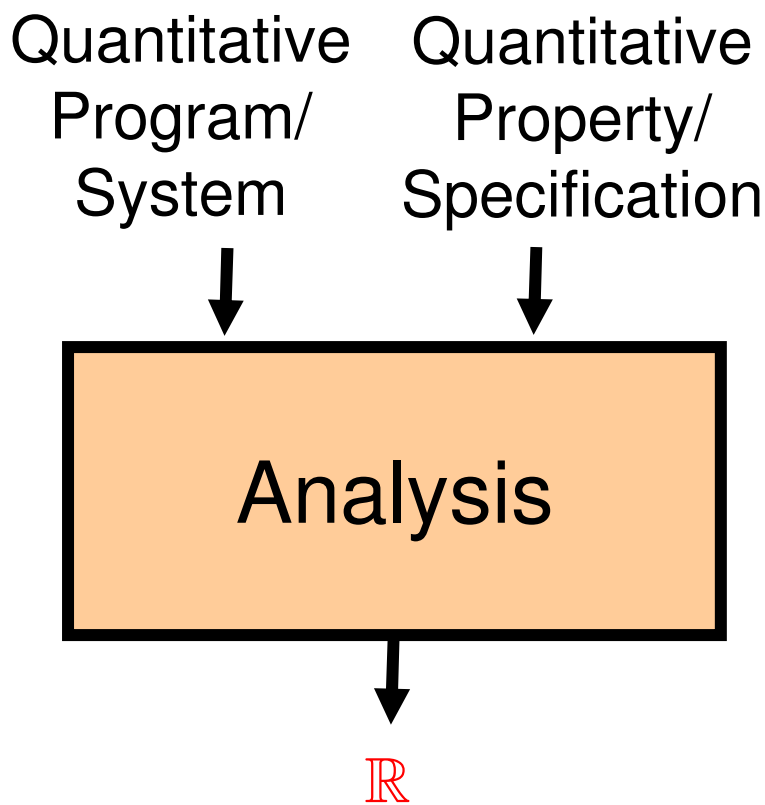
Every request is followed by a grant within probability 1/2.



$\mathbb{B}$

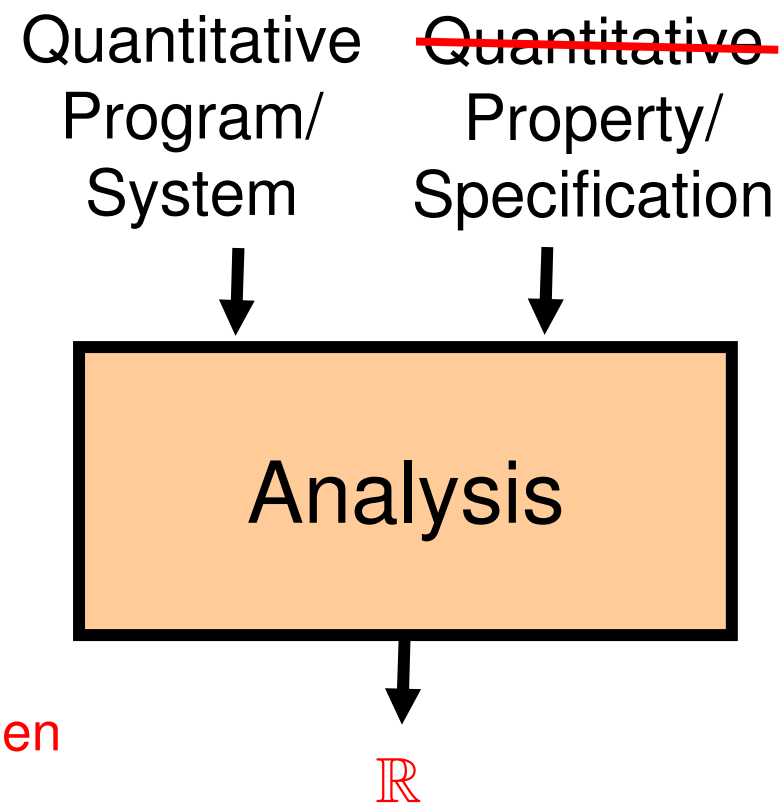
-perhaps a proof
-perhaps some counterexamples
-perhaps even a proposed fix

The Quantitative Agenda



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

The Quantitative Agenda

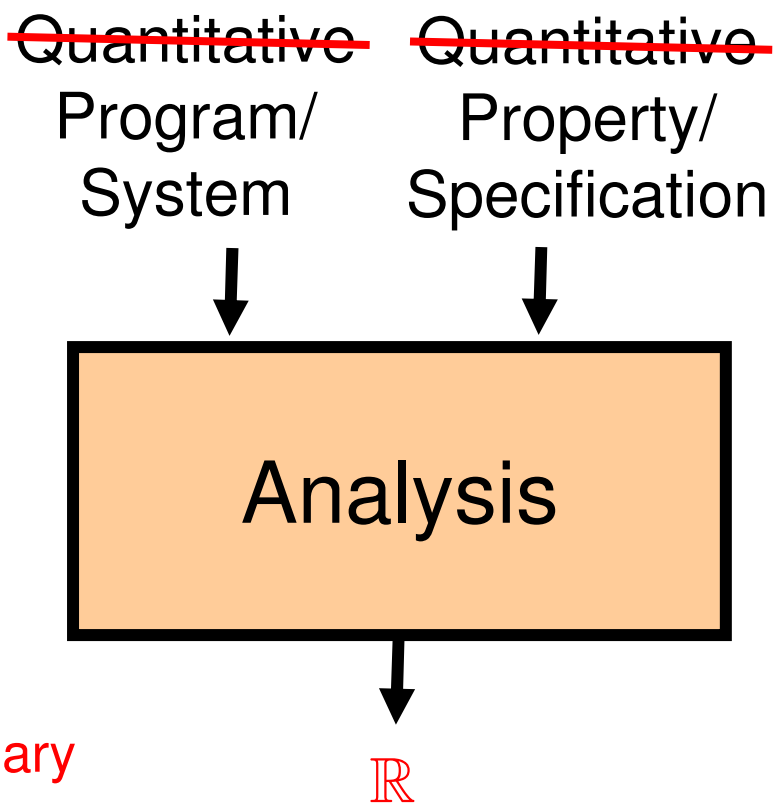


Every request is followed by a grant.

The less time between requests and grants, the better.

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

The Quantitative Agenda



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 ($0 < d < 1$):

aaaaaaaaaa...	1
aaaaaaab...	$1 - d^8$
aab...	$1 - d^3$
b...	0

Long-run average value:

aaaaaaaaaa...	1
aaabaaabaaab...	1 – ¼
abaabaaab...	0

Q2 Assigning Values To Systems

x: behaviors
w: observations
A,B: systems

$$A(w) = \sup_x \{ val(x) : obs(x) = w \} \quad ?$$
$$B(w) = \exp_x \{ val(x) : obs(x) = w \} \quad ??$$

Q2, Q3 Assigning Values To Systems

x: behaviors
w: observations
A,B: systems

$$A(w) = \sup_x \{ val(x) : obs(x) = w \} \quad ?$$
$$B(w) = \exp_x \{ val(x) : obs(x) = w \} \quad ??$$

$$diff(A,B) = \sup_w \{ |A(w) - B(w)| \} \quad ???$$

Compositionality: $diff(A||B,A'||B) \leq f(diff(A,A'))$ [AFHMS].

Is there a Quantitative Framework with

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

(Like the boolean theory of ω-regularity.)

Outline

Property = Language

Alphabet: Σ

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

Language: $L \subseteq \Sigma^\omega$

$L = (a^+b)^+(a^\omega \cup c^\omega) \cup (a^+b)^\omega$

abaabaaabccccc... $\in L$

abcabc... $\notin L$

Boolean Language

Alphabet: Σ

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

Language: $L \subseteq \Sigma^\omega$

$L = (a^+b)^+(a^\omega \cup c^\omega) \cup (a^+b)^\omega$

$abaabaaabccccc... \in L$

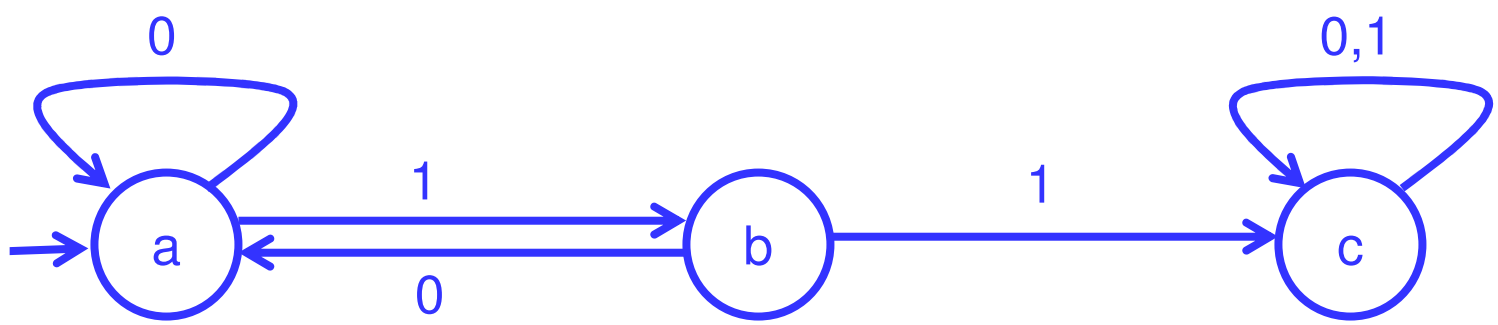
$abcabc... \notin L$

$L: \Sigma^\omega \rightarrow \mathbb{B}$

Specification = Automaton

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

A:

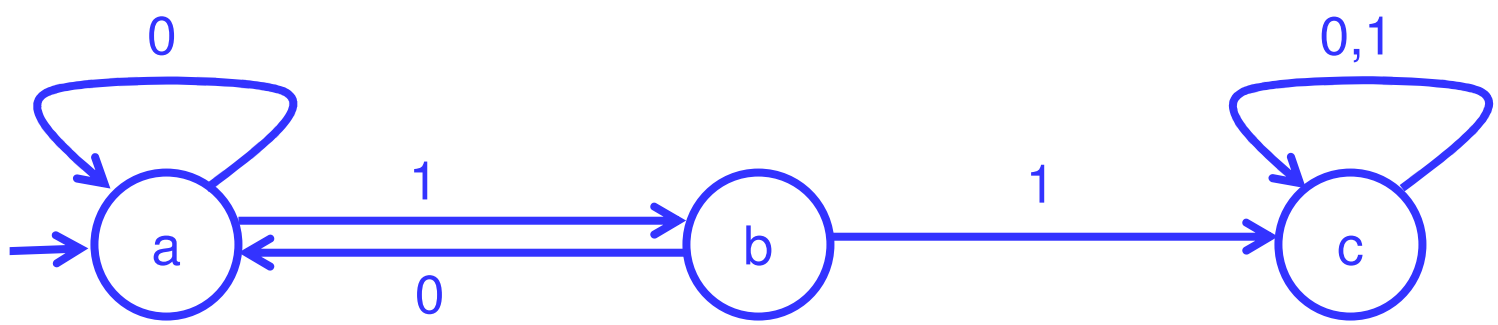


$\Gamma = \{0,1\}$

Specification = Automaton

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

A:



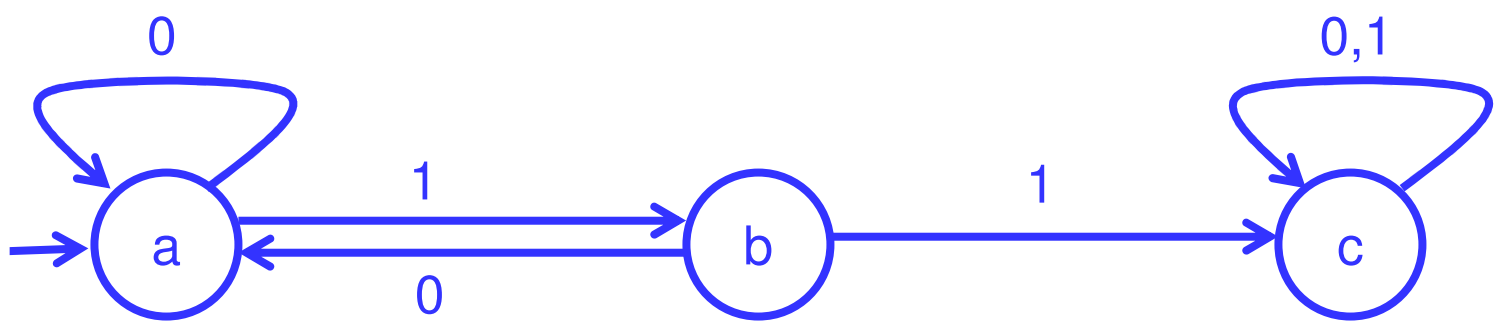
$\Gamma = \{0,1\}$

$L(A) = (a^+b)^+(a^\omega \cup c^\omega) \cup (a^+b)^\omega$

Automaton

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

A:



"scheduler" 0101111... $\to$ aababccc... "outcome"

Automaton

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

Scheduler: $x: Q^+ \to \Gamma$
S ... set of schedulers

Outcome: $f(x) = q_0 q_1 q_2 \ldots$
where $\forall\, i : q_{i+1} = \delta(q_i, x(q_0 \ldots q_i))$

Language: $L = \{ \lambda(f(x)) : x \in S \}$

Automaton

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

Scheduler: $x: Q^+ \to \Gamma$
S ... set of schedulers

Outcome: $f(x) = q_0 q_1 q_2 \ldots$
where $\forall\, i : q_{i+1} = \delta(q_i, x(q_0 \ldots q_i))$

Language: $L = \{ \lambda(f(x)) : x \in S \}$
$L(w) = \sup\{ f(x) : x \in S \text{ s.t. } \lambda(f(x)) = w \}$

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 automata, PSPACE-complete.

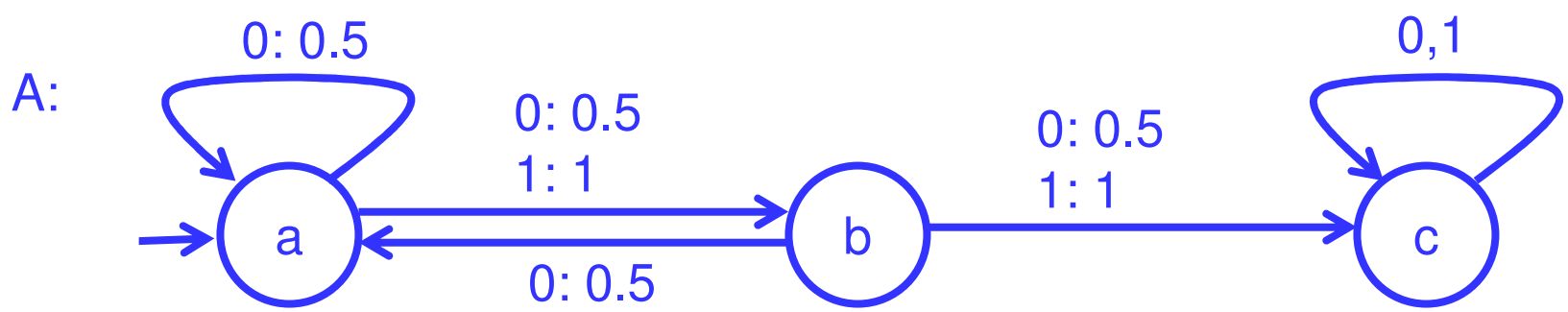
Probabilistic Language

Word:	element of Σ^ω
Probabilistic Word:	probability space on Σ^ω
Probabilistic Language:	set of probabilistic words

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

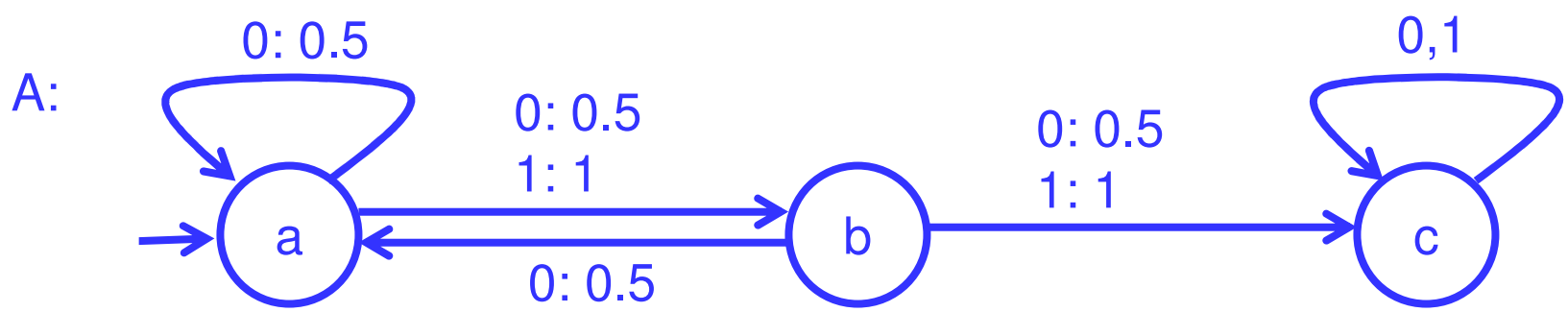
Markov Decision Process

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



Markov Decision Process

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



0101111... $\to$ abccc... $\to$ 1/2
aabccc... $\to$ 1/4
...

Markov Decision Process

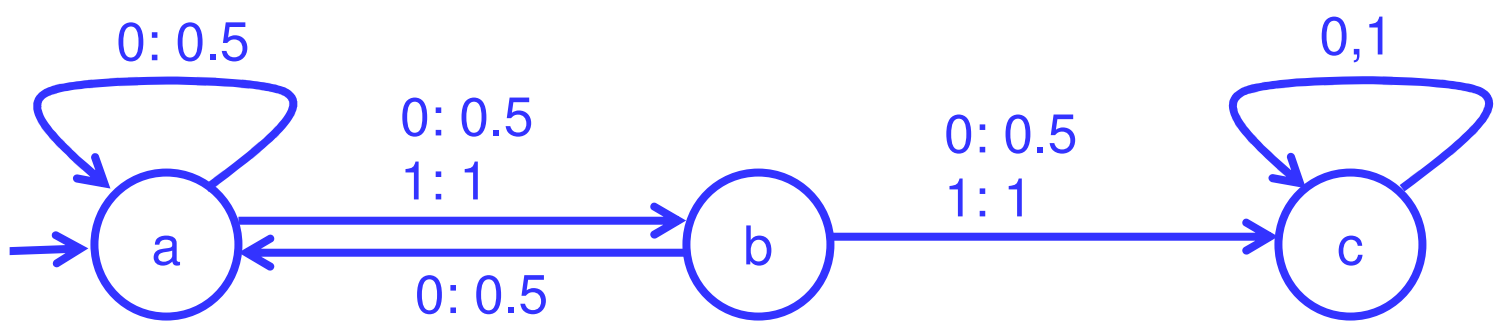
Q	states
$\lambda: Q \to \Sigma$	labeling
$q_0 \in Q$	initial state
Γ	choices
$\delta: \ Q \times \Gamma \to D(Q)$	transition function

Pure scheduler:	$x: Q^+ \to \Gamma$
Probabilistic scheduler:	$x: Q^+ \to D(\Gamma)$

Markov Decision Process

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

A:



$\{0: 0.5, 1: 0.5\}^\omega \ \to$ abccc... $\to$ 9/16

aabccc... $\to$ 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

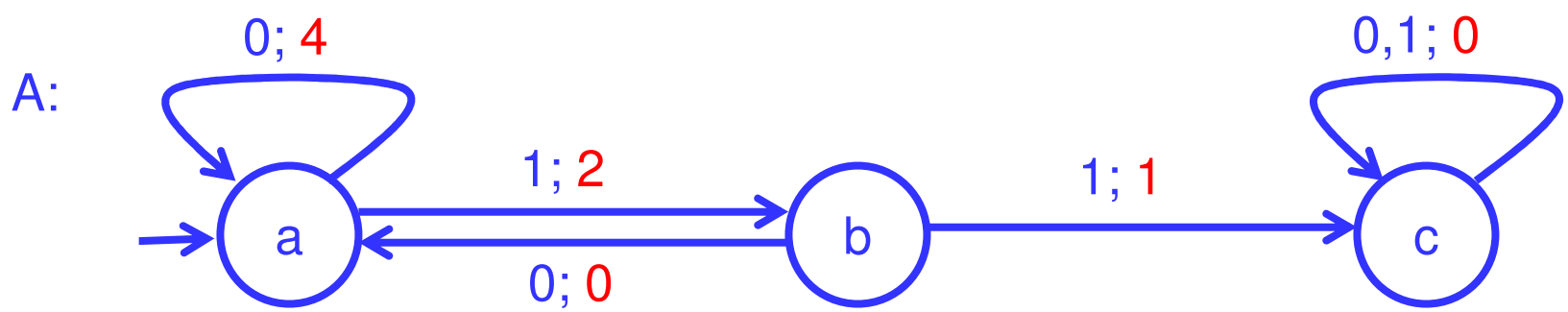
Language: $L: \Sigma^\omega \to \mathbb{B}$

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

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

Weighted Automaton

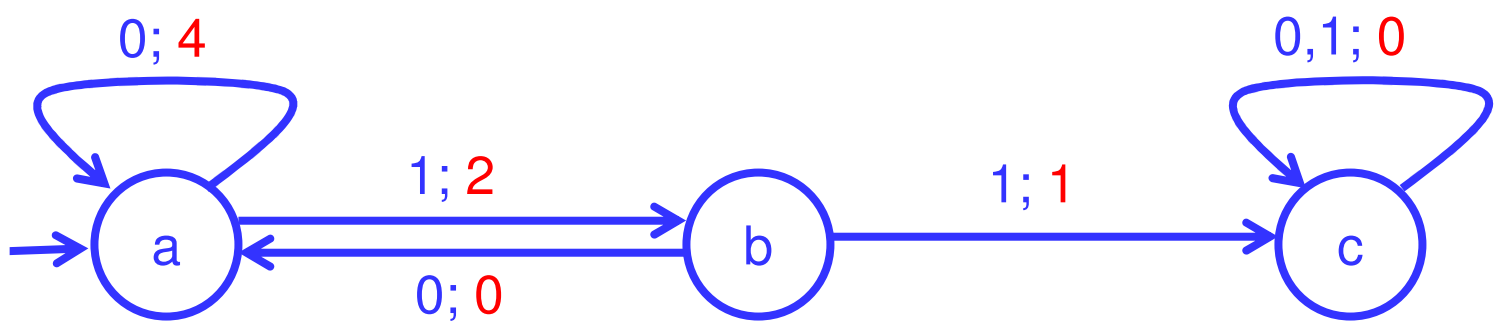
Q	states
$\lambda: Q \to \Sigma$	labeling
$q_0 \in Q$	initial state
Γ	choices
$\delta: \ Q \times \Gamma \to \mathbb{R} \times Q$	transition function



Weighted Automaton

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

A:



Max value:

0101111... $\to$ aababccc...; 4
1111111... $\to$ abccc...; 2

Weighted Automaton

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

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

Max value: $\mathrm{val}(q_0 v_1 q_1 v_2 q_2 \ldots) = \sup\{\, v_i : i \geq 1 \,\}$

Weighted Automaton

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

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

Max value: $\mathrm{val}(q_0 v_1 q_1 v_2 q_2 \ldots) = \sup\{\, v_i : i \geq 1 \,\}$

q-Language: $L(w) = \sup\{\, \mathrm{val}(f(x)) : x \in S \text{ s.t. } \lambda(f(x)) = w \,\}$

Different Value Functions

Max value: $\mathrm{val}(q_0 v_1 q_1 v_2 q_2 \ldots) = \sup\{ v_i : i \geq 1 \}$
(only 0 and 1 costs: finite automaton)

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

Different Value Functions

Max value: $val(q_0 v_1 q_1 v_2 q_2 ...) = \sup\{ v_i : i \geq 1 \}$
(only 0 and 1 costs: finite automaton)

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

Limavg value: $val = \lim_{n\to\infty} 1/n \cdot \sum_{1\leq i\leq n} v_i$

Different Value Functions

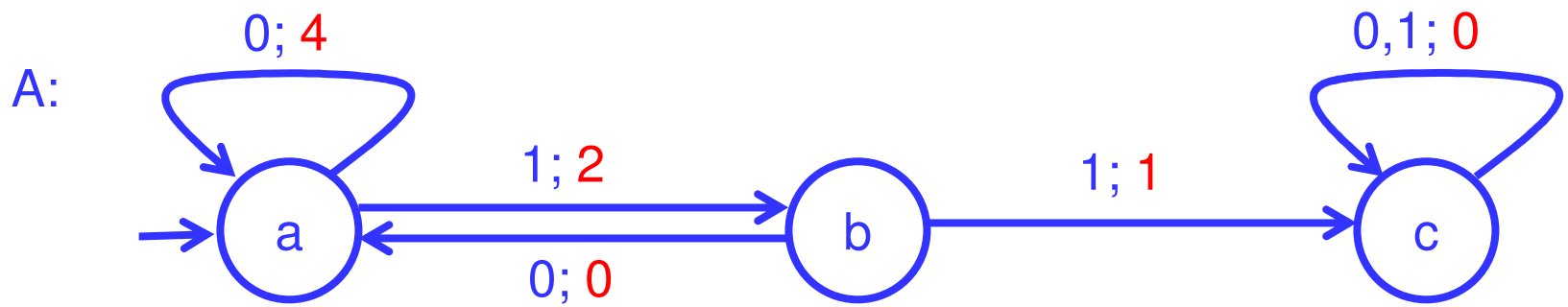
Max value: $val(q_0 v_1 q_1 v_2 q_2 \ldots) = \sup\{ v_i : i \geq 1 \}$
(only 0 and 1 costs: finite automaton)

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

Limavg value: $val = \lim_{n\to\infty} 1/n \cdot \sum_{1\leq i\leq n} v_i$

Discounted: $val = \sum_{i\geq 1} d^i \cdot v_i$ for some $0<d<1$

Weighted Automaton



Limsup value:

01010101... → aababababab...; 2
11111111... → abccc...; 0

Limavg value:

01010101... → aababababab...; 1
11111111... → abccc...; 0

Discounted:
(d = 0.5)

01010101... → aababababab...; 2.66...
11111111... → abccc...; 1.25

Quantitative Language Inclusion

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

Quantitative Language Inclusion

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

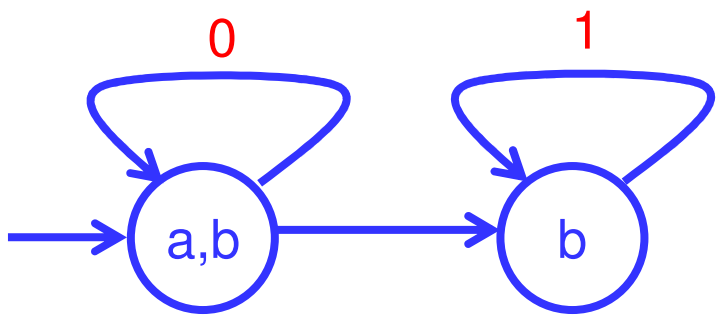
For max and limsup values: PSPACE.
For limavg and discounted values: Open.

If specification B is deterministic,
then it can be solved in polynomial time [CDH].

Expressiveness [CDH]

E.g. LimAvg automata not determinizable:

$\Sigma^* b^\omega$ expressible by nondeterministic LimAvg automaton.

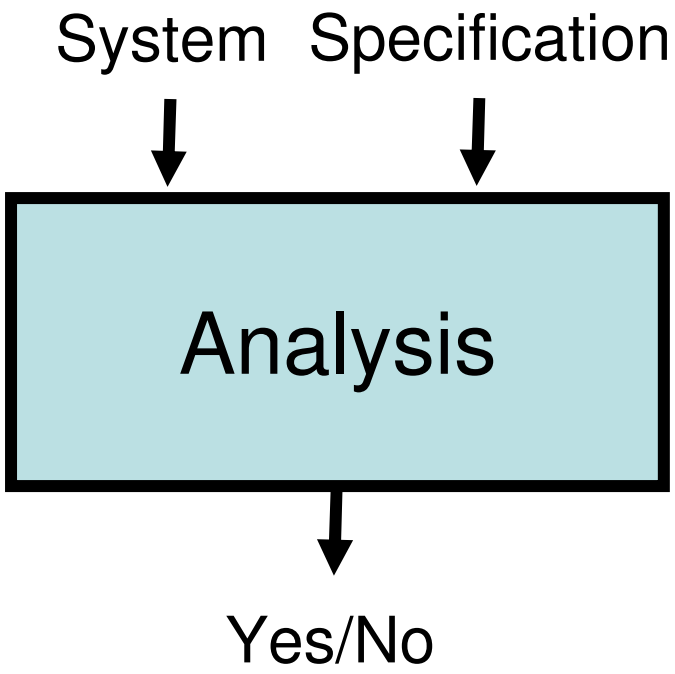


$\Sigma^* b^\omega$ not expressible by deterministic limAvg automaton.

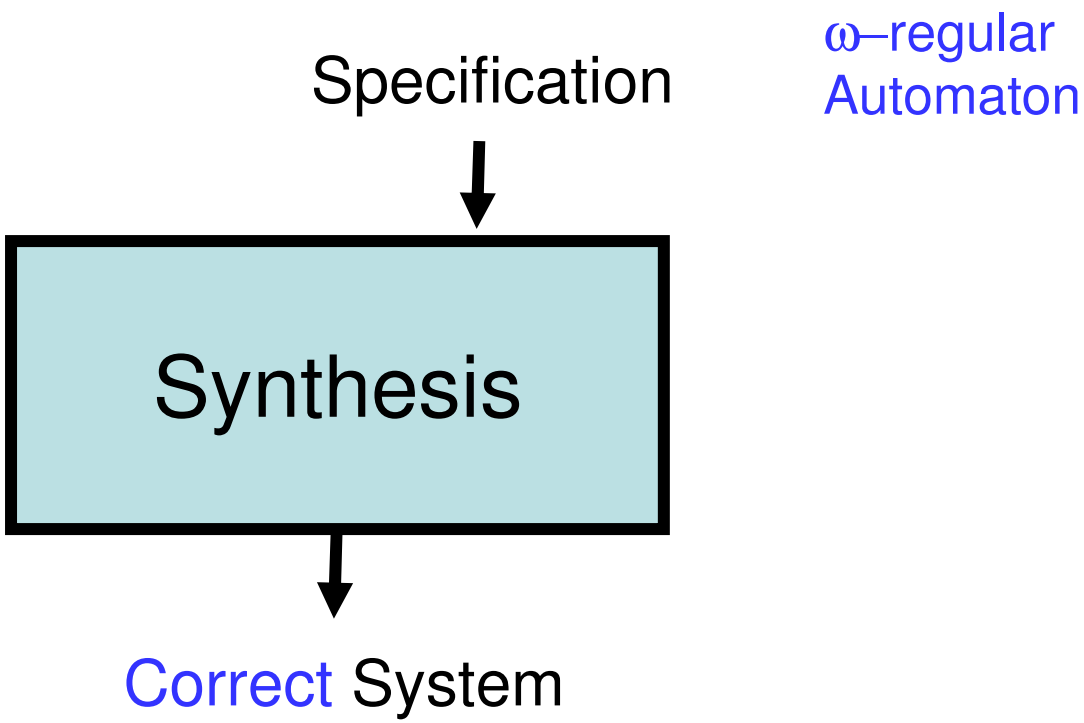
Every b-cycle would need weight 1.
Consider $w_n = (ab^n)^\omega$.
Then $val(w_n)=1$ for sufficiently large n, but $w_n \notin \Sigma^* b^\omega$.

Outline

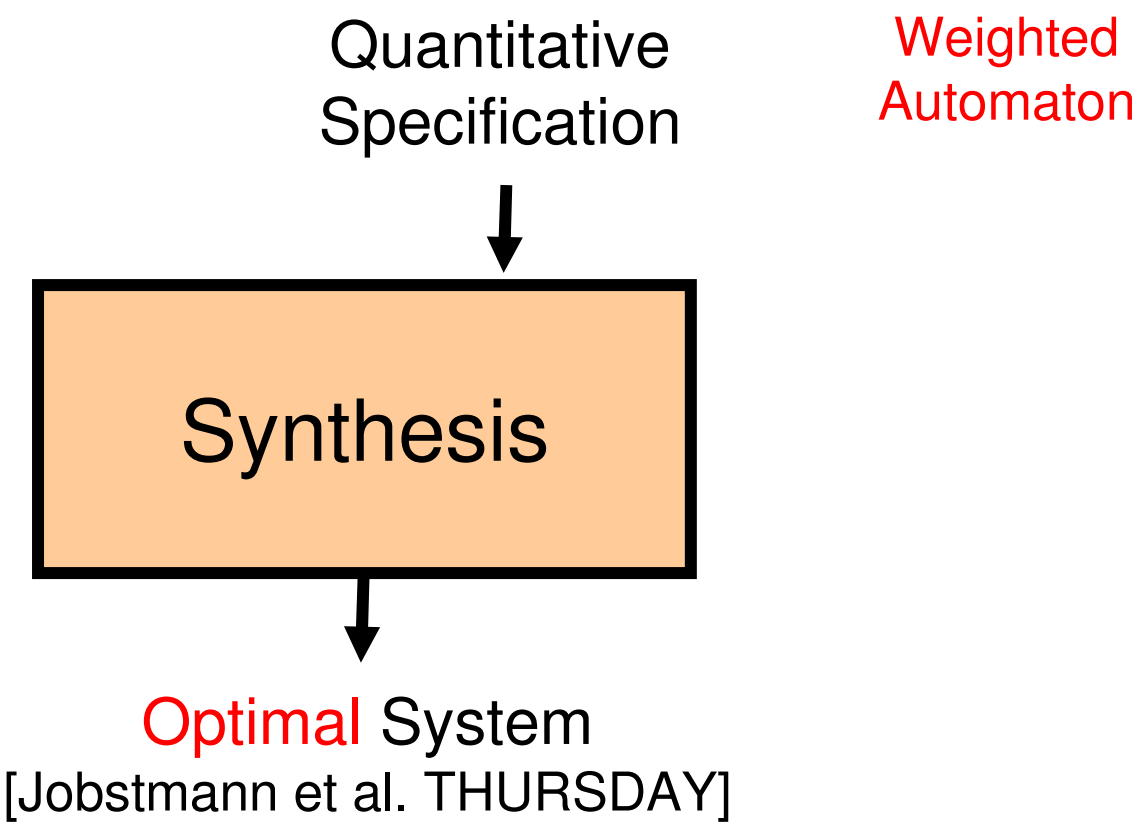
The Boolean Agenda



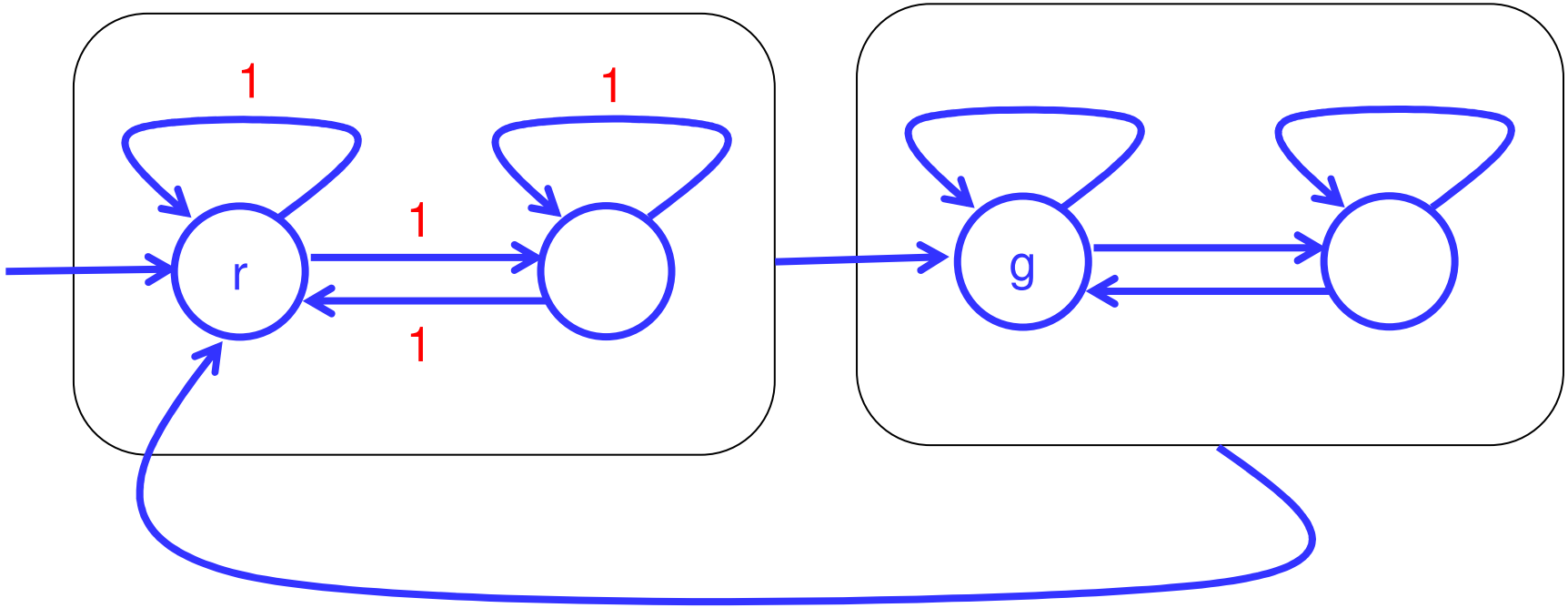
The Boolean Agenda



3.1 Quantitative Synthesis

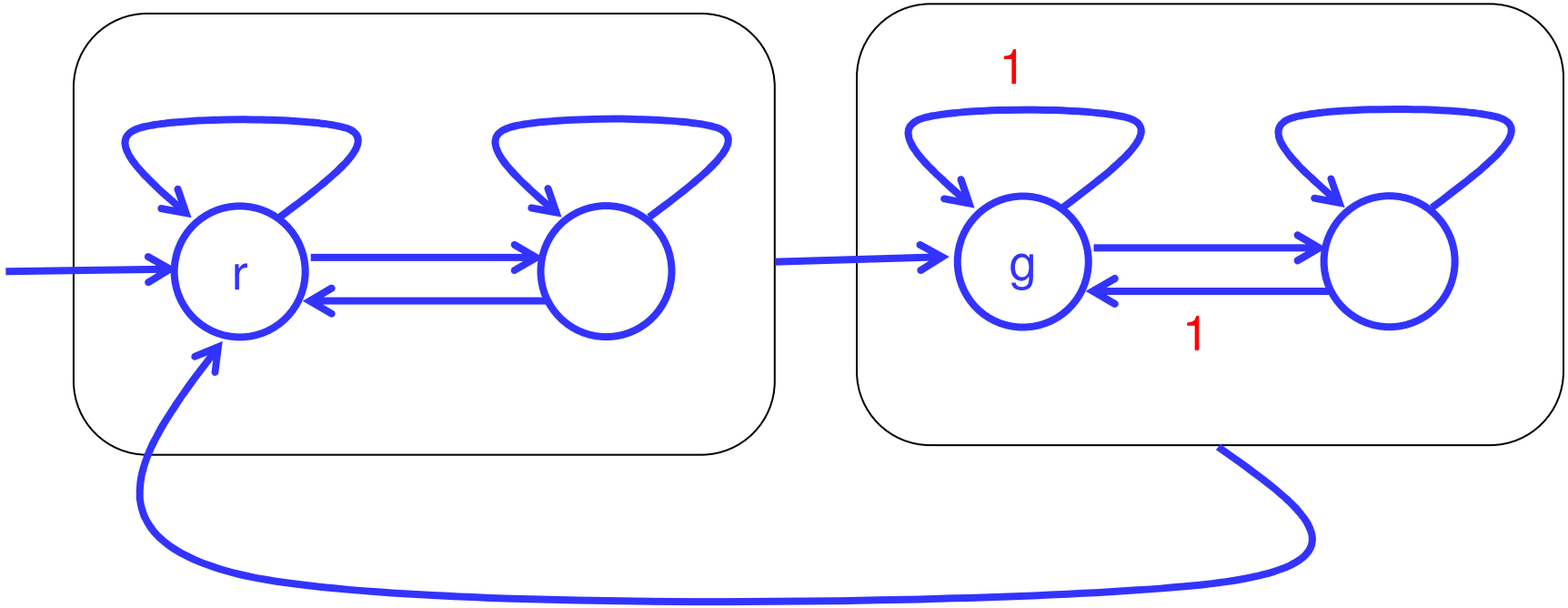


Request-Grant Automaton 1



Following a request, all steps until the next grant are penalized.

Request-Grant Automaton 2



Following a request, all repeated grants are penalized.

3.2 Robust Systems

A Robustness as Mathematical Continuity:

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

$\forall\, \varepsilon>0.\ \exists\, \delta>0.$ input-change $\leq \delta \Rightarrow$ output-change $\leq \varepsilon$

In general programs are not continuous.
But they can be more continuous:

```
read sensor value x at time t;
compute "continuous" function y = f(x);
write output value y at time t+d;
```

Or less continuous:

```
read sensor value x;
if x ≤ c  then y = f1(x)
          else y = f2(x);
```

In general programs are not continuous.
But they can be more continuous:

```
read sensor value x at time t;
compute continuous function y = f(x);
write output value y at time t+d;
```

Or less continuous:

```
read sensor value x;
if x ≤ c  then y = f1(x)
          else y = f2(x);
```

Better:

```
if x ≤ c - ε   then y = f1(x);
if x ≥ c + ε   then y = f2(x)
               else y =
```

$$\boxed{(f2(c+\varepsilon)-f1(c-\varepsilon))(x-c+\varepsilon)/2\varepsilon + f1(c-\varepsilon)}$$

[Majumdar et at., Gulwani et al.]

3.2 Robust Systems

A Robustness as Mathematical Continuity:

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

$\forall\, \varepsilon>0.\ \exists\, \delta>0.$ input-change $\leq \delta \Rightarrow$ output-change $\leq \varepsilon$

Example of a Robustness Theorem [AHM]:

If discountedBisimilarity(A,B) > 1 - ε,
then $\forall w : |A(w) - B(w)| < f(\varepsilon)$.

3.2 Robust Systems

A Robustness as Mathematical Continuity:

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

B Robustness w.r.t. Faulty Assumptions:

-few environment mistakes cause few system mistakes
-ratio of system to environment mistakes as quantitative quality measure

[Greimel et al. TODAY]

3.3 Resource Interfaces

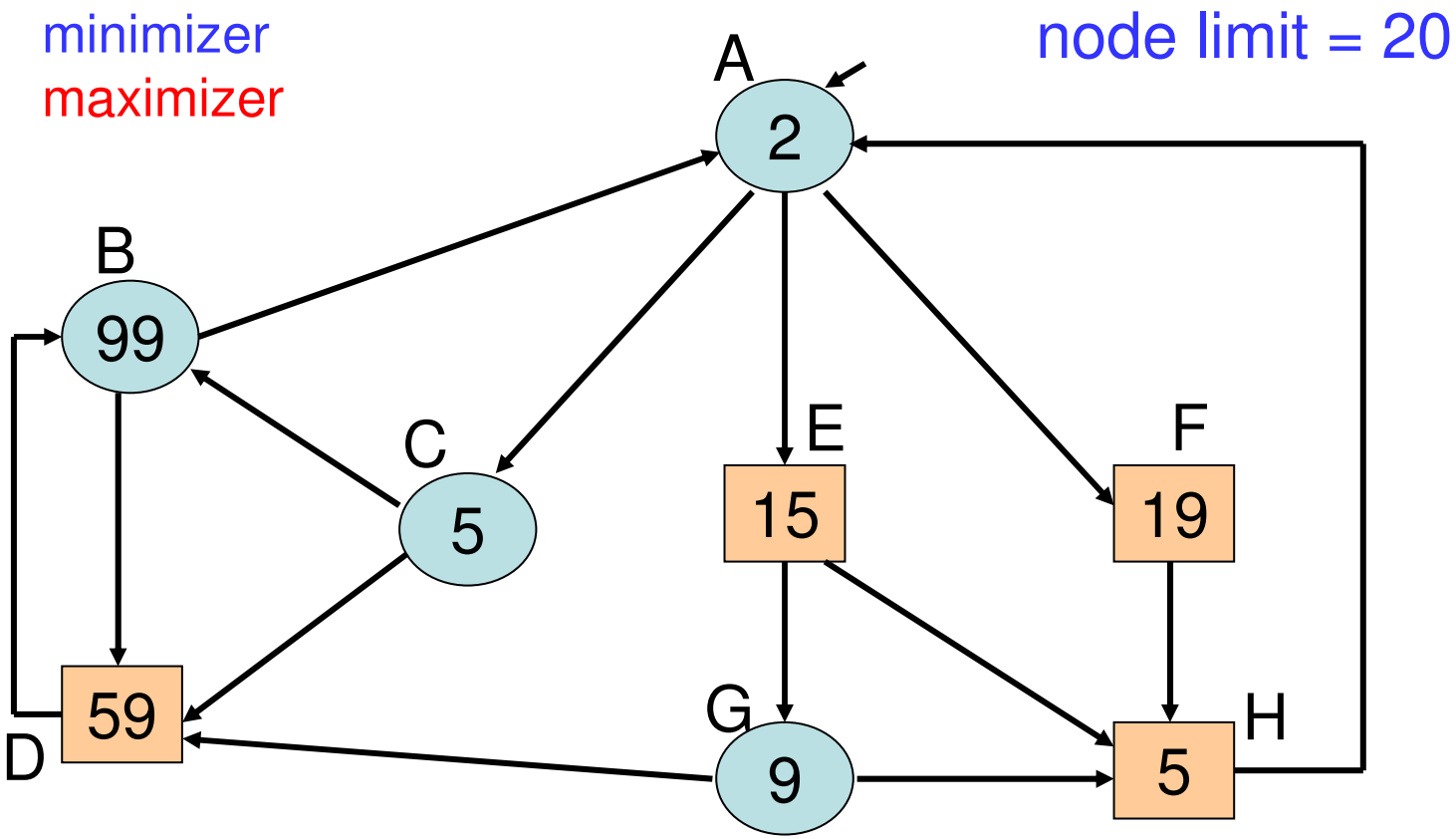
-Component interfaces expose resource requirements (e.g. time, memory, power).

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

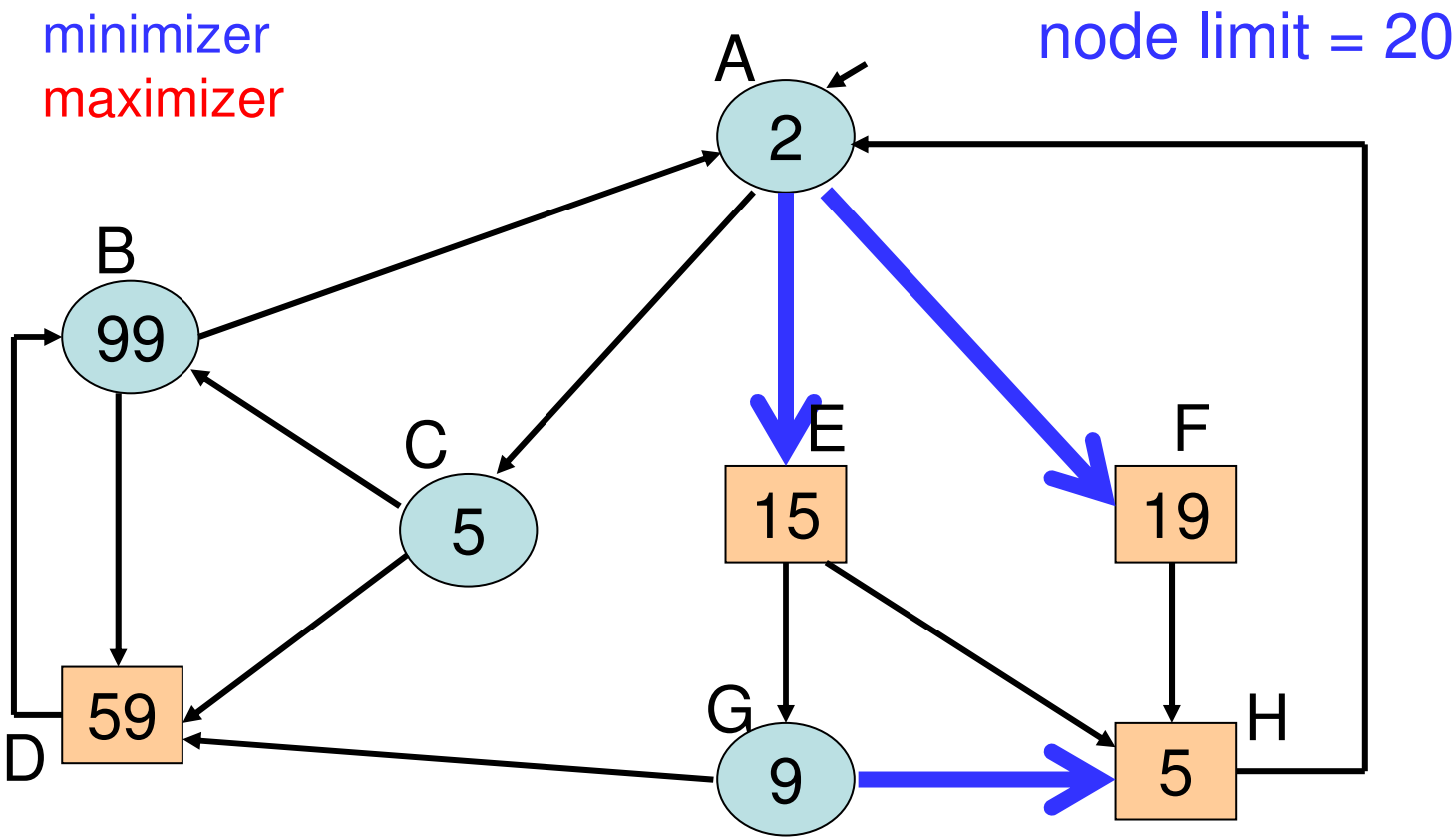
-If the requirements are dynamic, then compatibility can be solved as a graph game with quantitative objectives.

[Chakrabarti et al.]

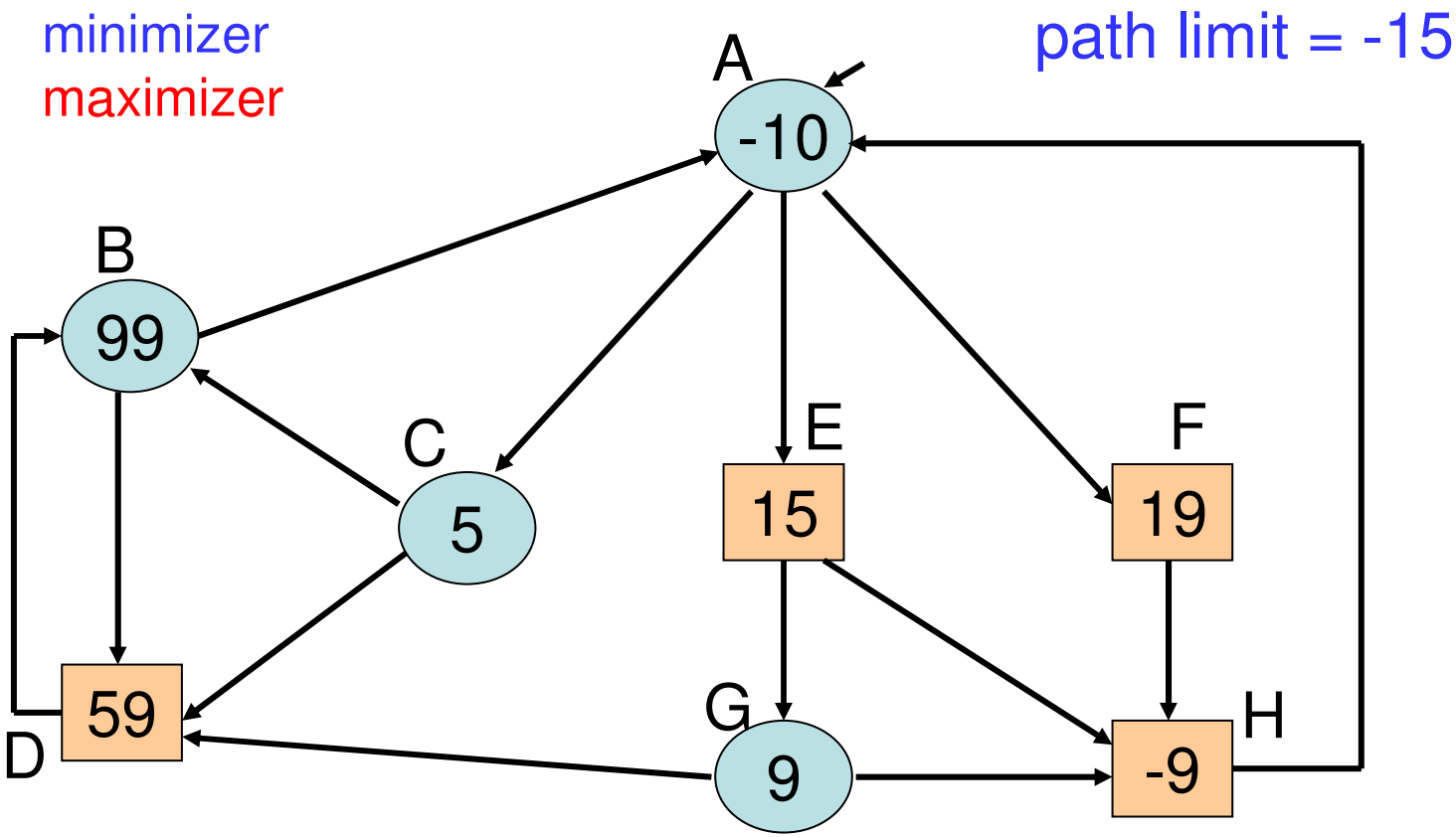
Max Value Constraint



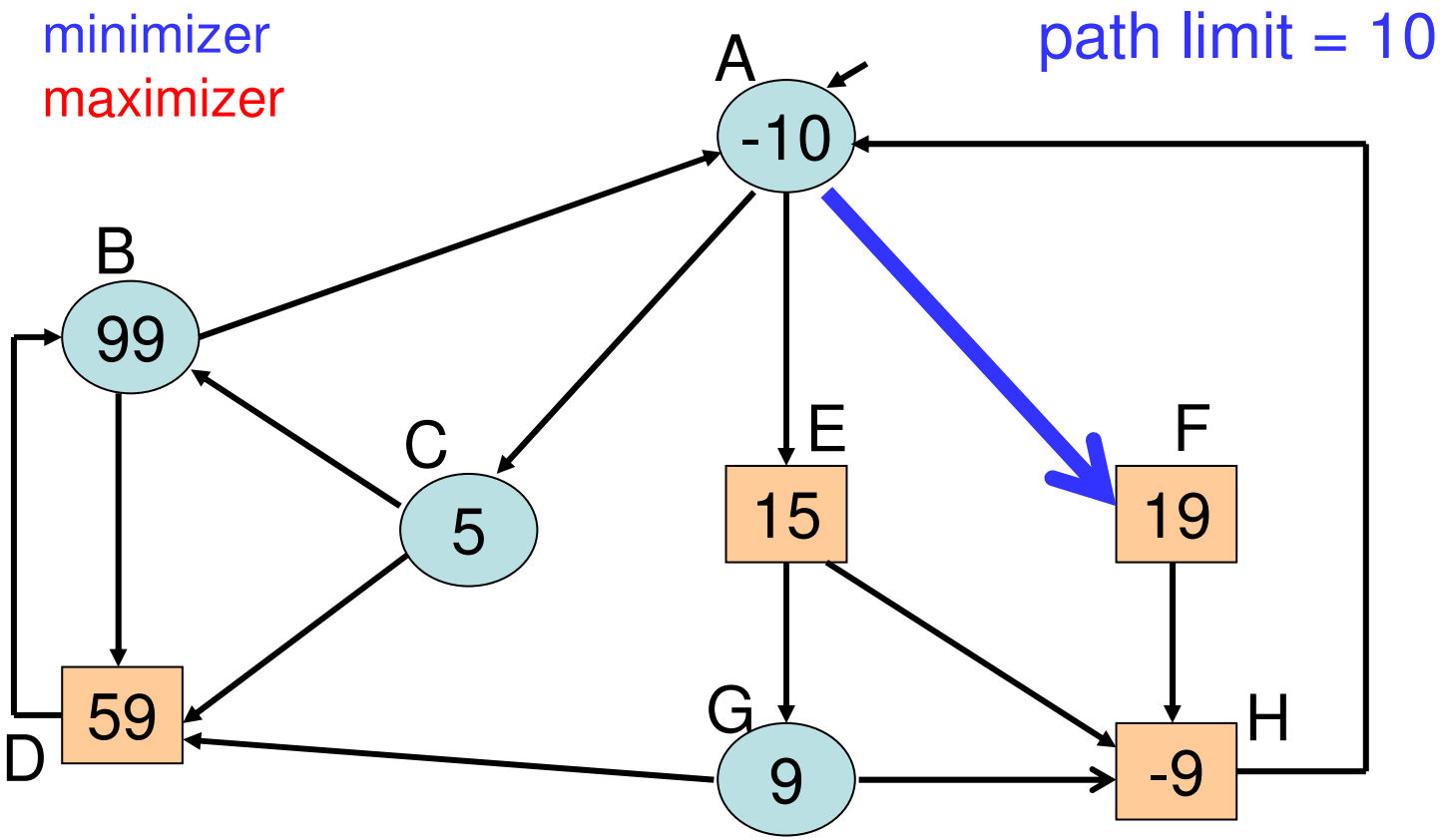
Max Value Constraint



Sum Constraint



Sum Constraint



3.4 System Reliability

-assuming x% of input values are valid,
y% of output values should be valid (limavg)

-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
aaabaaabaaab...	3/4
abaabaaab...	0

Want reliabitity 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.