

Transition Systems and Bisimulation

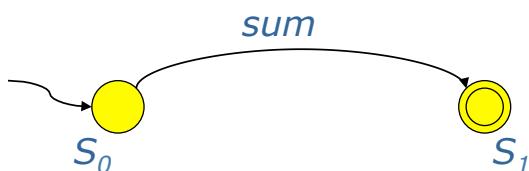
Giuseppe De Giacomo

Service Integration
A.A. 2010/11

Transition Systems

Concentrating on behaviors: SUM two integers

- Consider a program for computing the sum of two integers.
- Such a program has essentially two states
 - the state S_0 of the memory before the computation: including the two numbers to sum
 - the state S_1 of the memory after the computation: including the result of the computation
- Only one action, i.e. “sum”, can be performed



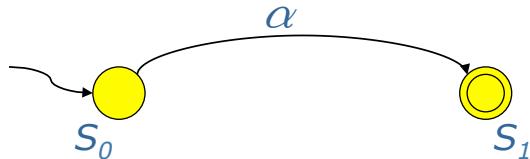
Concentrating on behaviors: CheckValidity

- Consider a program for computing the validity of a FOL formula:
- Also such a program has essentially two states
 - the state S_1 of the memory before the computation: including the formula to be checked
 - the state S_2 of the memory after the computation: including “yes”, “no”, “time-out”
- Only one action, i.e. “checkValidity”, can be performed



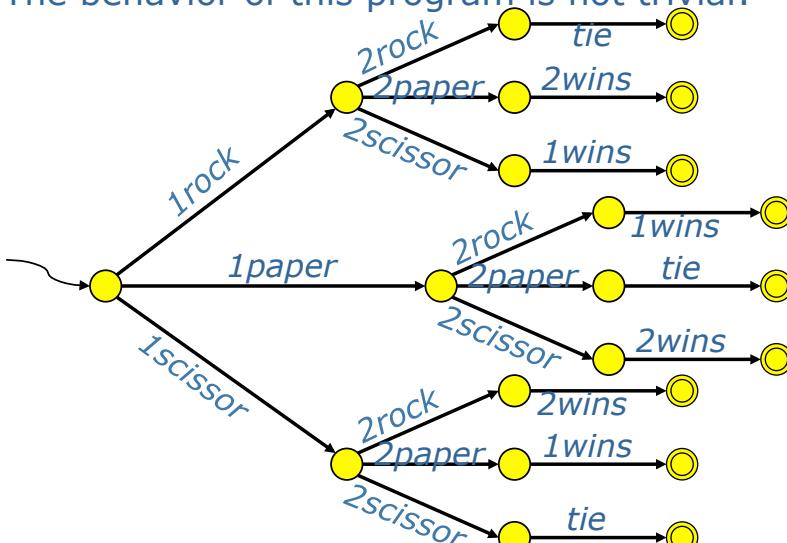
Concentrating on behaviors

- The programs SUM and CheckValidity are very different from a computational point of view.
 - SUM is trivial
 - CheckValidity is a theorem prover hence very complex
- However they are equally trivial from a behavioral point of view:
 - two states S_0 and S_1
 - a single action α causing the transition



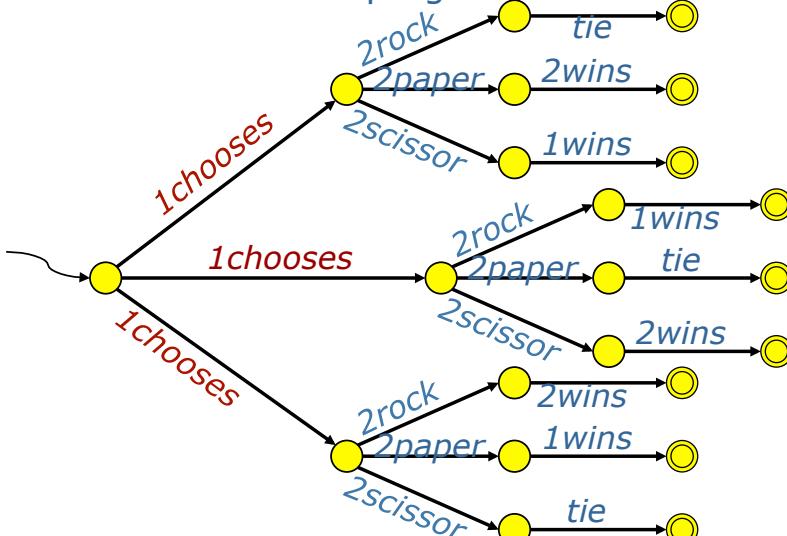
Concentrating on behaviors: RockPaperScissor

- Consider the program RockPaperScissor that allows to play two players the the well-known game.
- The behavior of this program is not trivial:



Concentrating on behaviors: RockPaperScissor (automatic)

- Consider a variant of the program RockPaperScissor that allows one players to play against the computer.
- The behavior of this program is now **nondeterministic**:



Service Integration – aa 2010/11

Giuseppe De Giacomo

7

Concentrating on behaviors: WebPage

<http://www.informatik.uni-trier.de/~ley/db/>



A web page can have a complex behavior!

dbLP.uni-trier.de

COMPUTER SCIENCE BIBLIOGRAPHY

UNIVERSITÄT TRIER

maintained by Michael Ley - [Welcome](#) - [FAQ](#)

Mirrors: [ACM SIGMOD](#) - [VLDB Endow.](#) - [SuSiTE Central Europe](#)

Search

• [Author](#) - [Title](#) - [Advanced](#) - [New: Faceted search \(L3S Research Center, U. Hannover\)](#)

Bibliographies

- Conferences: [SIGMOD](#), [VLDB](#), [PODS](#), [ER](#), [EDBT](#), [ICDE](#), [POPL](#), ...
- Journals: [CACM](#), [TODS](#), [TOIS](#), [TOPLAS](#), [DKE](#), [VLDB](#), [J. Inf. Systems](#), [TPLP](#), [TCS](#), ...
- Series: [LNCS](#)/[LNAL](#), [IFIP](#)
- Books: [Collections](#) - [DB Textbooks](#)
- By Subject: [Database Systems](#), [Logic Prog.](#), [IR](#), ...

Full Text: [ACM SIGMOD Anthology](#)

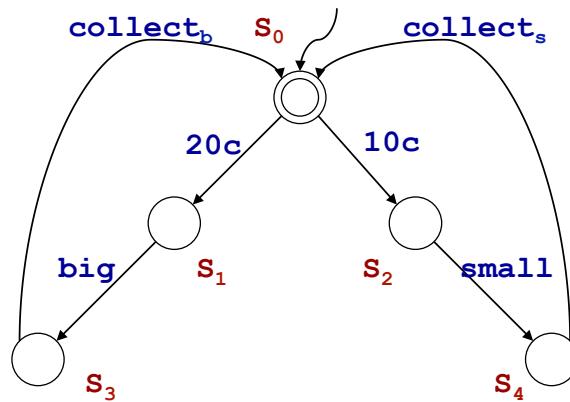
Links

- Computer Science Organizations: [ACM](#) (DL), [SIGMOD](#), [SIGIR](#), [IEEE Computer Society](#) (DL), [IEEE Xplore](#), [IEE](#) ...
- Related Services: [CiteSeer](#), [CS BiTeX](#), [ie-port.net](#), [CoRR](#), [NZ-DL](#), [Zentralblatt MATH](#), [MathSciNet](#), [Erdős Number Base](#), [Math Genealogy](#), [Paul Erdős](#), ...

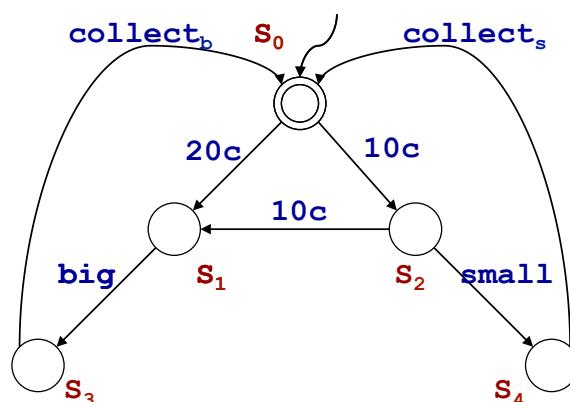
Giuseppe De Giacomo

8

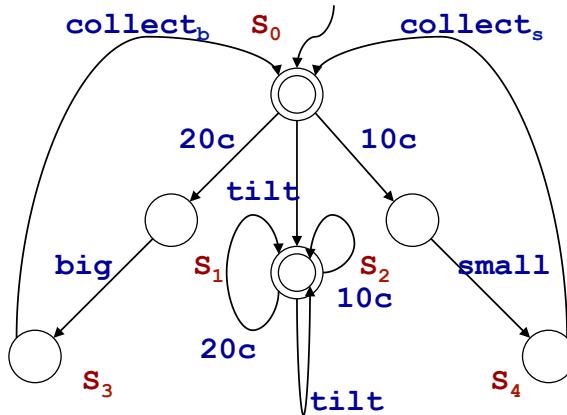
Concentrating on behaviors: Vending Machine



Concentrating on behaviors: Another Vending Machine



Concentrating on behaviors: Vending Machine with Tilt



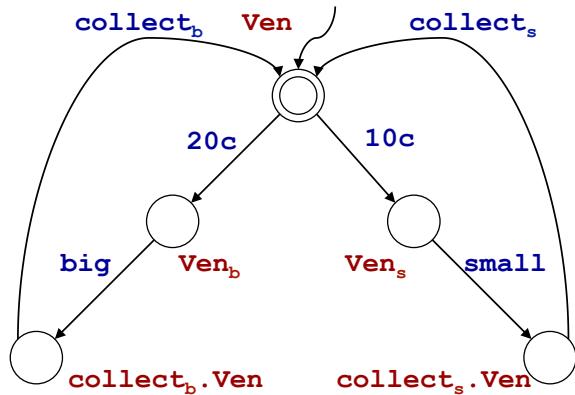
Transition Systems

- A transition system TS is a tuple $T = \langle A, S, S^0, \delta, F \rangle$ where:
 - A is the set of actions
 - S is the set of states
 - $S^0 \subseteq S$ is the set of initial states
 - $\delta \subseteq S \times A \times S$ is the transition relation
 - $F \subseteq S$ is the set of final states
- Variants:
 - No initial states
 - Single initial state
 - Deterministic actions
 - States labeled by propositions other than Final/ \neg Final

(c.f. Kripke Structure)

Process Algebras are Formalisms for Describing TS

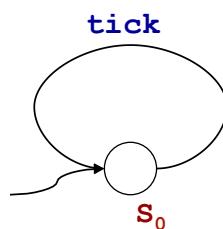
- Trans (a la CCS)
 - $Ven = 20c.Ven_b + 10c.Ven_s$
 - $Ven_b = \text{big.collect}_b.Ven$
 - $Ven_s = \text{small.collect}_s.Ven$
- Final
 - ✓ Ven



- TS may have infinite states - e.g., this happens when generated by process algebras involving iterated concurrency
- However we have good formal tools to deal only with finite states TS

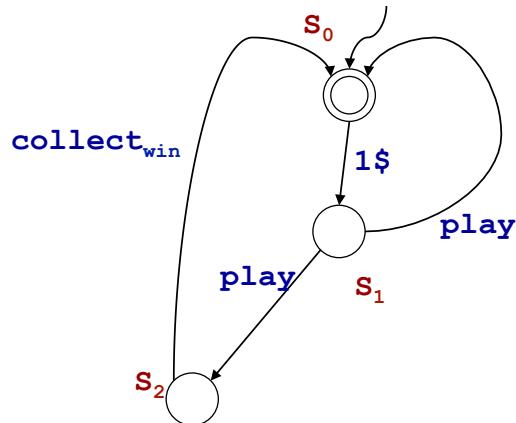
Example (Clock)

TS may describe (legal) nonterminating processes

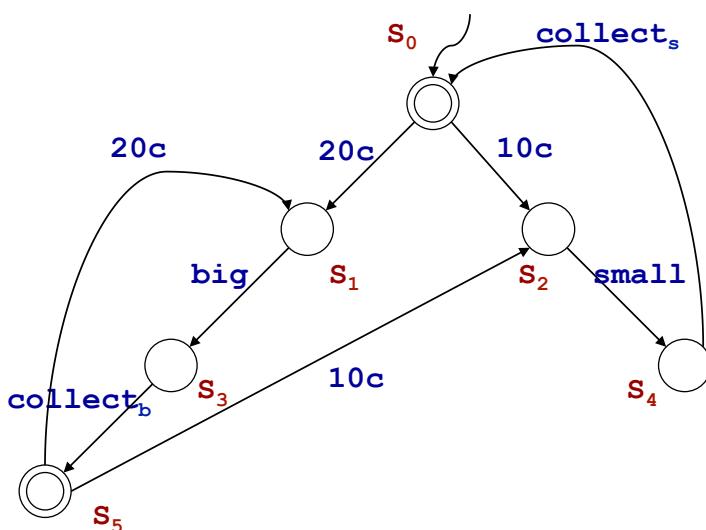


Example (Slot Machine)

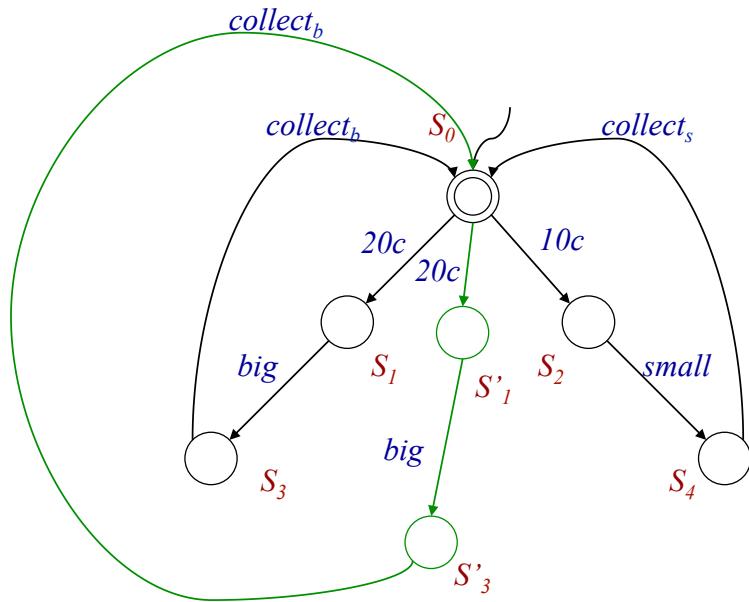
Non-deterministic transitions express
choice that is not under the control of clients



Example (Vending Machine - Variant 1)



Example (Vending Machine - Variant 2)



**Inductive vs Coinductive Definitions:
Reachability, Bisimilarity, ...**

Reachability

- A binary relation R is a **reachability-like relation** iff:
 - $(s,s) \in R$
 - if $\exists a, s'. s \xrightarrow{a} s' \wedge (s',s'') \in R$ then $(s,s'') \in R$
- A state s_0 of transition system S **reaches** a state s_f iff for **all** a **reachability-like relations** R we have $(s_0, s_f) \in R$.
- Notably that
 - **reaches** is a reachability-like relation itself
 - **reaches** is the **smallest** reachability-like relation

Note it is a inductive definition!

Computing Reachability on Finite Transition Systems

Algorithm ComputingReachability

Input: transition system TS

Output: the **reachable-from** relation (the smallest reachability-like relation)

Body

```
R = ∅
R' = {(s,s) | s ∈ S}
while (R ≠ R') {
    R := R'
    R' := R' ∪ {(s,s'') | ∃ s', a. s →a s' ∧ (s',s'') ∈ R }
```

return R'

YdoB

Bisimulation

- A binary relation R is a **bisimulation** iff:

$(s, t) \in R$ implies that

- s is *final* iff t is *final*
- for all actions a
 - if $s \xrightarrow{a} s'$ then $\exists t' . t \xrightarrow{a} t'$ and $(s', t') \in R$
 - if $t \xrightarrow{a} t'$ then $\exists s' . s \xrightarrow{a} s'$ and $(s', t') \in R$

- A state s_0 of transition system S is **bisimilar**, or simply **equivalent**, to a state t_0 of transition system T iff there **exists** a **bisimulation** between the initial states s_0 and t_0 .
- Notably
 - **bisimilarity** is a bisimulation
 - **bisimilarity** is the **largest** bisimulation

Note it is a co-inductive definition!

Computing Bisimilarity on Finite Transition Systems

Algorithm ComputingBisimulation

Input: transition system $TS_S = \langle A, S, S^0, \delta_S, F_S \rangle$ and
transition system $TS_T = \langle A, T, T^0, \delta_T, F_T \rangle$

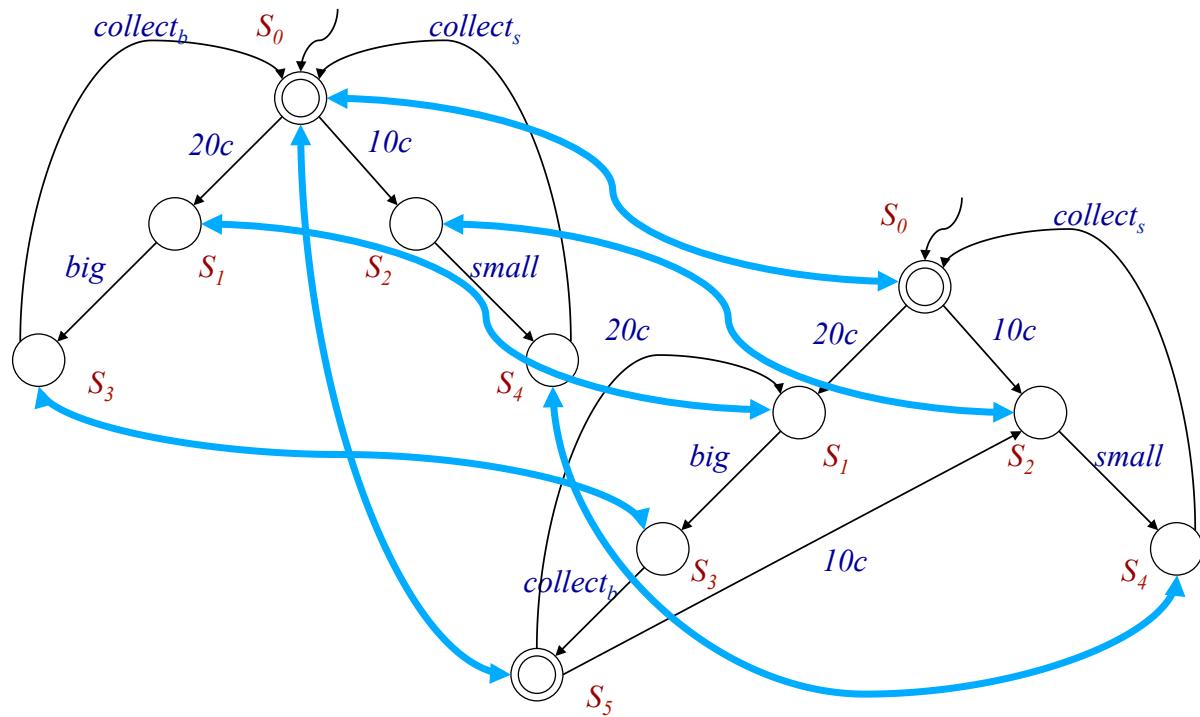
Output: the **bisimilarity** relation (the largest bisimulation)

Body

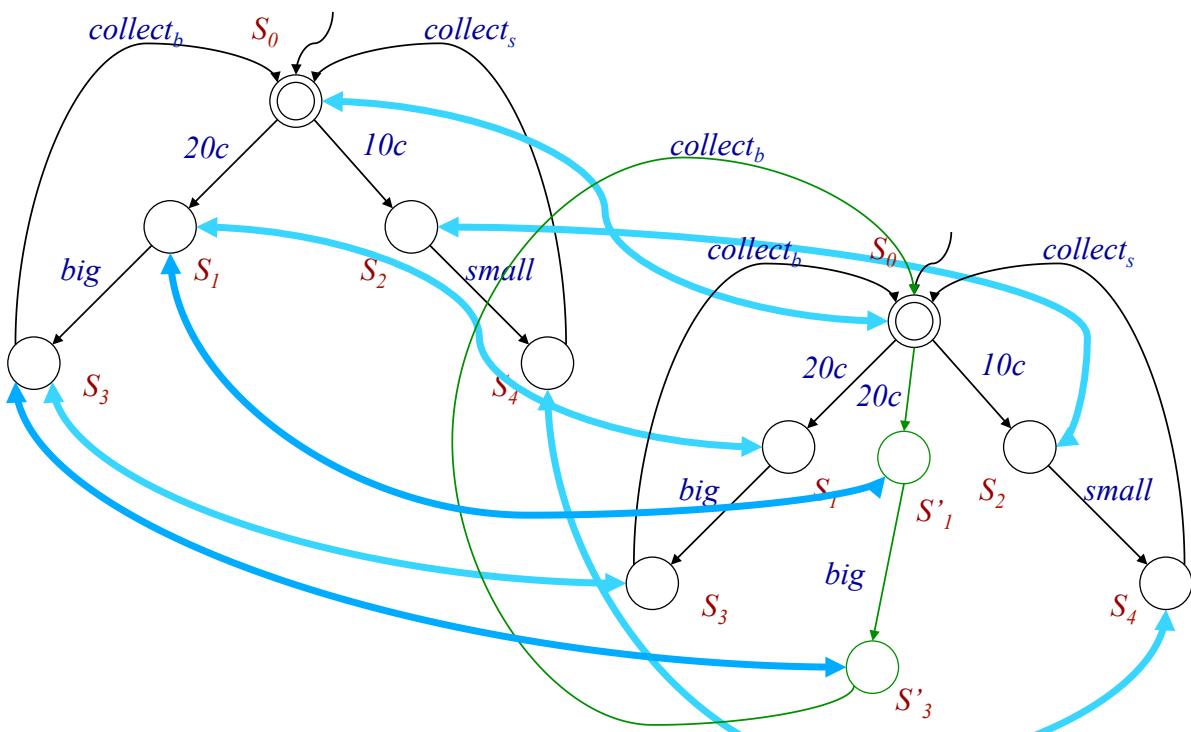
```
R = S × T
R' = S × T - {(s,t) | ¬(s ∈ F_S ≡ t ∈ F_T)}
while (R ≠ R') {
    R := R'
    R' := R' - ( {(s,t) | ∃ s', a. s →_a s' ∧ ∃ t', a. t →_a t' ∧ (s',t') ∈ R' }
                  {(s,t) | ∃ t', a. t →_a t' ∧ ∃ s', a. s →_a s' ∧ (s',t') ∈ R' } )
}
return R'
```

Ydob

Example of Bisimulation



Example of Bisimulation



Automata vs. Transition Systems

- Automata
 - define sets of runs (or traces or strings): (finite) length sequences of actions
- TSs
 - ... but I can be interested also in the alternatives "encountered" during runs, as they represent client's "choice points"

