Design representations

Control Oriented Models

Kris Kuchcinski

Krzysztof.Kuchcinski@cs.lth.se

Models

"A theory has only the alternative of being right or wrong. A model has a third possibility: it may be right, but irrelevant."

Manfred Eigen (1927 -)
Jagdish Mehra (ed.) The Physicist's Conception of Nature, 1973.

Control Flow Models

- FSM (Mealy and Moore)
- FSM extensions
  - Codesign FSM
  - Communicating FSM
- Petri nets
- StateCharts
- Discrete Event
- CCS, CSP, …
- ...

Application Areas

- Reactive systems
- Control functions
- Protocols (telecom, computers, ...)
- ...

FSM basics

- Different communication mechanisms:
  - synchronous (classical FSM's, Moore, Mealy)
  - asynchronous (CCS, Milner '80; CSP, Hoare '85)
- Mealy and Moore state machines FSM = <S, I, O, δ, λ>

  S is set of states

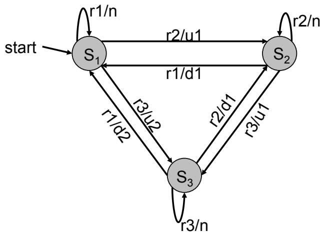
  I is set of inputs (conditions)

  δ is a next-state function, δ: $S \times I \rightarrow S$

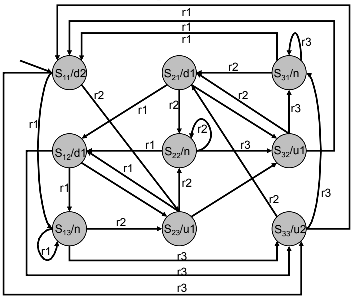
  λ is an output function, λ: $S \times I \rightarrow O$ for Mealy machine

  λ: $S \rightarrow O$ for Moore machine

FSM Model for Elevator Controller Mealy Machine



FSM Model for Elevator Controller
Moore Machine



2012-03-30 7

Moore vs. Mealy machines

- Theoretically, same computational power (almost)
- In practice, different characteristics
- Moore machines:
  - non-reactive (response delayed by 1 cycle)
  - easy to *compose* (always well-defined)
- Mealy machines:
  - reactive (0 response time)
  - hard to *compose* (problem with combinational cycles)

2012-03-30 8

Problems with FSM's

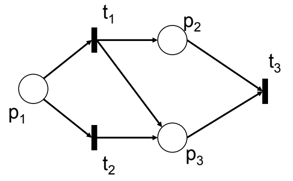
- How to reduce the size of the representation?
- Solution — *hierarchical concurrent finite state machines*
- Example — Harel's StateCharts
- 3 orthogonal exponential reductions
  - hierarchy,
  - concurrency,
  - non-determinism.

2012-03-30 9

Petri Nets

- Model introduced by C.A. Petri in 1962
  Ph.D. Thesis: “Communication with Automata”
- Applications: distributed computing, manufacturing, control, communication networks, transportation…
- Petri nets describe explicitly and graphically:
  - sequencing/causality,
  - conflict/non-deterministic choice,
  - concurrency.
- Asynchronous model (partial ordering)
- Main drawback: *no hierarchy*

Petri Net example



Definition

- A Petri net structure, C, is a four tuple

C = (P, T, I, O)

$P = \{p_1, p_2, ..., p_n\}$ is a finite set of places, $n \geq 0$;

$T = \{t_1, t_2, ..., t_m\}$ is a finite set of transitions, $m \geq 0$;

$P \cap T = \emptyset$;

$I: T \rightarrow P^\infty$ is the input function, a mapping from transitions to *bags* of places;

$0: T \rightarrow P^\infty$ is the output function, a mapping from transitions to *bags* of places;

Petri Net Marking

- A marking m of a Petri net C = (P, T, I, O) is a function from the set of places P to the non-negative integer
  $\mu : P \rightarrow N$
- A marking represents an assignment of *tokens* to the places.
- A marking m can also be defined as an n-vector, $\mu = (\mu_1, \mu_2, ..., \mu_n)$, where n= |P| and $\mu_i \in N$, i = 1, 2, ..., n. The number of tokens in the place p_i is denoted by μ_i.
- A marked Petri net M = (C, μ) is a Petri net structure C = (P, T, I, O) and a marking μ.

Summary

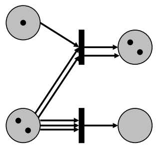
- A (C,μ_0) is a Petri Net Graph N
- places: represent distributed state by holding tokens
  - marking (state) μ is an n-vector $(\mu_1,\mu_2,\mu_3\ldots)$, where μ_i is the non-negative number of tokens in place pi.
  - initial marking (μ_0) is initial state
- transitions: represent actions/events
  - enabled transition: enough tokens in predecessors
  - firing transition: modifies marking
- …and an initial marking μ_0.
- Place/Transition <=> conditions/events

Firing Rules

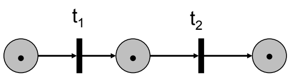
- The execution of a Petri net is carried out by *firing transitions*, which moves tokens from places to places.
- A transition $t_j \in T$ in a marked Petri net C = (P, T, I, O) with marking m is *enabled* if for all $p_i \in I(t_j)$
  $\mu(p_i) \geq \#(p_i, I(t_j))$
- A transition t_j in a marked Petri net with marking μ may fire whenever it is enabled.
  Firing an enabled transition t_j results in a new marking μ' defined by
  $\mu'(p_i) = \mu(p_i) - \#(p_i, I(t_j)) + \#(p_i, O(t_j))$

$\#(p_i, I(t_j))$ $(\#(p_i, O(t_j)))$ – number of occurrences of p_i in $I(t_j)$ $(O(t_j))$

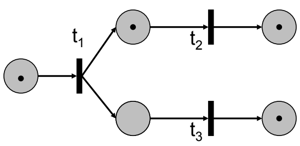
An Example



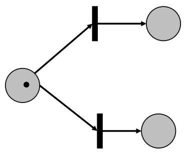
Sequencing



Concurrency

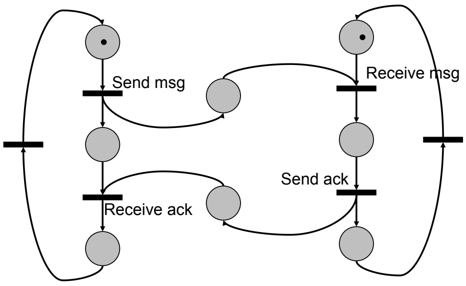


Conflict

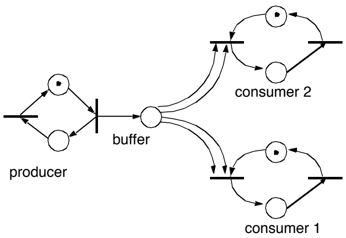


The firing order is not irrelevant.

Communication Protocol



Produces/Consumer



Properties of Petri Nets

- Most important analysis problems for Petri nets:
  - reachability and coverability
  - liveness
  - boundness
  - safeness
  - conservation

2012-03-30 22

Reachability

- Marking μ is *reachable* from marking μ_0 if there exists a *sequence of firings* s = μ_0 t_1 μ_1 t_2 μ2… μ that transforms μ_0 to μ.
- The reachability problem is decidable.

2012-03-30 23

Liveness

- *Liveness*: from any marking any transition can become fireable
- Liveness implies deadlock freedom, not viceversa

2012-03-30 24

Boundness

- *Boundedness*: the number of tokens in any place cannot grow indefinitely
  - (1-bounded also called *safe*)
- Application: places represent buffers and registers (check there is no overflow)

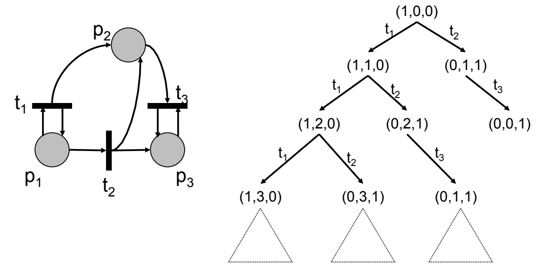
Conservation

- *Conservation*: the total number of tokens in the net is constant

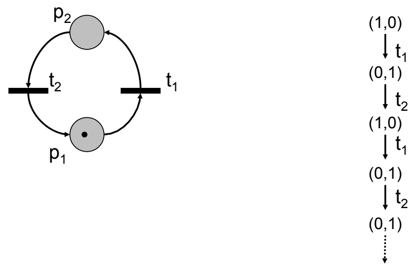
Analysis Techniques

- State Space Analysis techniques
  - Reachability Tree or Coverability Graph
- Structural analysis techniques
  - Incidence matrix
  - T- and S- Invariants

The Reachanility Tree



Infinite Tree



Reachability Tree

- We will try to limit the tree to the finite size (notice, however, that it will usually result in the lost of information)
- We introduce three types of nodes
  - frontier
  - terminal
  - duplicate

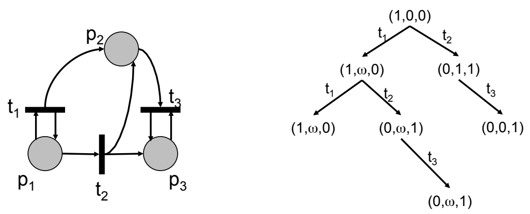
An extended marking

- Let us assume that after sequence of transitions σ we will end up in marking μ' from μ
  and $\mu' > \mu$
  $\mu' = \mu + (\mu'-\mu)$ and $(\mu' - \mu) > 0$
  since transition firing can be repeated it can lead to μ''
  $\mu'' = \mu' + (\mu'-\mu)$
  or $\mu'' = \mu' + 2(\mu'-\mu)$
  after n times we can produce a marking
  $\mu' + n(\mu'-\mu)$
  which, in fact, produces infinite marking.

Infinite Marking

- infinite number of markings is represented by ω symbol with following properties:
  - $\omega + a = \omega$
  - $\omega - a = \omega$
  - $\alpha < \omega$
  - $\omega \leq \omega$

Finite Reachability Tree



Analysis Techniques Based on Reachabilty Tree

- it can be used for analysing several properties such as
  - safeness and boundness
  - conservation
  - coverability
- it cannot be used, in general, to solve problems such as
  - reachability
  - liveness
  - determine which firing sequences are possible
- limitations => loss information by the use of ω symbol

Matrix Equations

- An alternative definition of Petri nets
  - instead of defining (P, T, I, O), we define (P, T, D^-, D^+), where two matrices D^- and D^+represent input and output function
  - we define
    $D^-[i, j] = \#(p_i, I(t_j))$
    $D^+[i, j] = \#(p_i, O(t_j))$
  - the transition t_j is represented by the unit *m*-vector e[j]

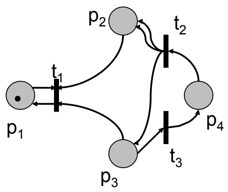
Matrix Equations (cont'd)

- transition t_j is enabled in a marking μ if
  $\mu \geq e[j] \cdot D^-$
- the result of firing transition t_j in marking μ, if it is enabled, is

$$\delta(\mu, t_j) = \mu - e[j] \cdot D^- + e[j] \cdot D^+ = \mu + e[j] \cdot (- D^- + D^+) = \mu + e[j] \cdot D$$

where $D = D^+ - D^-$

Matrix Equations - Example



$$D^- = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$D^+ = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & -1 & -1 & 0 \\ 0 & 2 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

Examples

- firing t_3

$$\mu' = [1 \quad 0 \quad 1 \quad 0] + [0 \quad 0 \quad 1] \times \begin{bmatrix} 0 & -1 & -1 & 0 \\ 0 & 2 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix} = [1 \quad 0 \quad 1 \quad 0] + [0 \quad 0 \quad -1 \quad 1] = [1 \quad 0 \quad 0 \quad 1]$$

- firing $t_3\,t_2\,t_3\,t_2\,t_1$

$$\mu' = [1 \quad 0 \quad 1 \quad 0] + [1 \quad 2 \quad 2] \times \begin{bmatrix} 0 & -1 & -1 & 0 \\ 0 & 2 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix} = [1 \quad 0 \quad 1 \quad 0] + [0 \quad 3 \quad -1 \quad 0] = [1 \quad 3 \quad 0 \quad 0]$$

Examples (cont' d)

- To determine if the marking (1, 8, 0, 1) is reachable from the marking (1, 0, 1, 0), we have the equation

$$[1 \quad 8 \quad 0 \quad 1] = [1 \quad 0 \quad 1 \quad 0] + x \times \begin{bmatrix} 0 & -1 & -1 & 0 \\ 0 & 2 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

$$[0 \quad 8 \quad -1 \quad 1] = x \times \begin{bmatrix} 0 & -1 & -1 & 0 \\ 0 & 2 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

- which has a solution x = (0, 4, 5) and sequence the s = $t_3t_2t_3t_2t_3t_2t_3t_2t_3$.

Matrix Equations - Problems

- matrix D by itself does not properly reflect the structure of the Petri net - *self loops* disappear.
- lack of sequencing information in the firing vector.
- a solution of equation $\mu' = \mu + x \cdot D$ is *necessary* for reachability but it is not *sufficient*

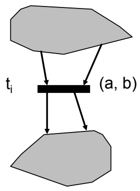
Summary of Petri Nets

- Graphical formalism
- Distributed state (including buffering)
- Concurrency, sequencing and choice made explicit
- Structural and behavioral properties
- Analysis techniques available

Petri Nets Extensions

- Add interpretation to tokens and transitions
  - Colored nets (tokens have value)
- Add time
  - time/timed Petri Nets (deterministic delay)
    - type (duration, delay)
    - where (place, transition)
    - control (weak, strong)
  - Stochastic PNs (probabilistic delay)
  - Generalized Stochastic PNs (timed and immediate transitions)
- Add hierarchy
  - Place Chart Nets

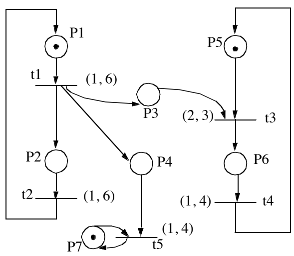
Time Petri Nets (TPN's)



- a ($0 \leq a$), is the minimal time that must elapse, starting from the time at which transition t_i is enabled, until this transition can fire,
- b ($0 \leq b \leq \infty$), denotes the maximal time during which transition t_i can be enabled without being fired.

Reference: B. Berthomieu and M. Diaz, *Modeling and Verification of Time Dependent Systems Using Time Petri Nets*, IEEE Trans. on Software Engineering, vol. 17, no. 3, March 1991.

TPN example



Some Properties of TPN's

- The reachability and boundness problems for TPN's are undecidable.
- There exist subclasses of TPN's which are bound.

Literature

- Tadao Murata, "Petri Nets: Properties, Analysis and Applications", Proceedings of IEEE, vol. 77, no. 4, April 1989.
- Bernard Berthomieu and Michel Diaz, "*Modeling and Verification of Time Dependent Systems Using Time Petri Nets*", IEEE Trans. on Software Engineering, vol. 17, no. 3, March 1991.
- D. Harel, et. al., "STATEMATE: A Working Environment for the Development of Complex Reactive Systems", IEEE Trans. on Software Engineering, vol. 16, no. 4, April 1990.