Petri Nets

- **Definitions**
  - **Source Transition**: a transition without any input place
    » is unconditionally enabled
  - **Sink Transition**: a transition without any output place
    » consumes but does not create any tokens
  - **Self-Look**: $P$ is both an input and output place of $T$
  - **Pure Petri Net**: does not contain self-loops
  - **Ordinary Petri Net**: all of the arc weights are unity, i.e. one.
  - **Infinite Capacity Net**: assumes that each place can accommodate an unlimited number of tokens
  - **Finite Capacity Net**: max. token-capacity $K(P)$ defined for each $P$
  - **Strict Transition Rule**: finite capacity net with additional rule that the number of tokens in each output place $P$ of $T$ cannot exceed its capacity $K(P)$ after firing $T$. 
Petri Nets

- Modeling Constructs
  - Concurrency
  - Precendence
  - Conflict, choice or decision
    - function: “exclusive OR”
    - only one transition can fire
    - weight: probability of taking that arc
Petri Nets

- Modeling Constructs
  - Synchronization
    » AND
    » joining several paths into a single path

![Diagram of Petri Nets with synchronization example]
Example

Fig. 8. A Petri net showing a dataflow computation for $x = (a + b)/(a - b)$. 

$$x = \frac{a + b}{a - b}$$
Example

Fig. 9. A simplified model of a communication protocol.
Petri Nets

- Modeling Constructs
  - Time
    » need new concept => timed transition
    » timed transition has firing delay $T$
    » when transition is enabled, wait $T$, then fire
      ▹ tokens are consumed and created at the firing instance
    » timed Petri Net symbol

- Stochastic Petri Net
  - $T$ is not fixed
  - $T = \text{random variable with } exponential \text{ distribution}$
Petri Nets

- Generalized Stochastic Petri Nets (GSPN)
  Adds extra constructs
  - Mixed transitions
    » stochastic and instantaneous transitions
  - Multiple Arcs

same as

» needs 2 tokens to fire
Petri Nets

- Generalized Stochastic Petri Nets (cont.)
  - Inhibitory Arcs
    » token inhibits firing
    » obviously no token transfer
    » watch for deadlocks!

  - Multiple Inhibitory Arcs
    » needs at least $N$ tokens to inhibit firing
    » less than $N$ tokens $\Rightarrow$ transition is firable
Petri Nets

- **Reachability**
  - fundamental basis for studying the dynamic properties of any system
  - firing of enabled transition will change token distribution
  - sequence of firings results in sequence of markings
  - marking $M_n$ is reachable from $M_0$ if there exists a sequence of firings that transforms $M_0$ into $M_n$
  - firing sequence is denoted by
    - $\sigma = M_0 t_1 M_1 t_2 \ldots t_n$ or simply $\sigma = t_1 t_2 \ldots t_n$
    - in this case $M_n$ is reachable from $M_0$ by $\sigma$
  - the set of all possible markings reachable from $M_0$ in a net $(N,M_0)$ is denoted by $R(N,M_0)$ or simply $R(M_0)$
  - the set of all possible firing sequences from $M_0$ in a net $(N,M_0)$ is denoted by $L(N,M_0)$ or simply $L(M_0)$
Petri Nets

- Reachability Graph
  - Petri Net with initial marking
    \[ M(t_0) = \{m_1, m_2\} = \{2, 0\} \]
  - Reachability Graph

  » add transitions to graph and…
  » Markov chain
Petri Nets

- Reachability Graph
  - Petri Net with initial marking
    \[ M(t_0) = \{m_1, m_2, m_3\} \]
  - Reachability Graph

\[
\begin{align*}
p_1 & \quad \quad \quad \quad p_2 \\
p_3 & \quad \quad \quad \quad 0.0.2 \\
1.1.0 & \quad \quad \quad \quad 0.1.1 \\
& \quad \quad \quad \quad 1.0.1
\end{align*}
\]
**Petri Nets**

- **Boundedness**
  - A Petri net \((N, M_0)\) is said to be \textit{k-bounded} (or simply \textit{bounded}) if the number of tokens in each place does not exceed a finite number \(k\) of any marking reachable from \(M_0\), i.e., \(M(p) \leq k\) for every place \(p\) and every marking \(M \in R(M_0)\)
  - example of 2-bound net
**Petri Nets**

- **Liveness**
  - closely related to the complete absence of deadlock in OS
  - A Petri net \((N,M_0)\) is said to be *live* (or equivalently \(M_0\) is said to be a *live* marking of \(N\)) if, no matter what marking has been reached from \(M_0\), it is possible to ultimately fire *any* transition of the net by progressing through some further firing sequence.

A live Petri net guarantees deadlock-free operation, no matter what firing sequence is chosen. However, this property is costly to verify, e.g. for large systems.
Petri Nets

- How did we get the net of the candy machine?
  - identify places needed

![Petri Net Diagram]

0  5  15
10  20
**Petri Nets**

- Example: candy machine
  - identify paths from places to places and the events that get you there (interpret the numbers as “deposit x cents”).

Diagram:

- Place 0: Initial state
- Place 5: 5 cents
- Place 10: 10 cents
- Place 15: 15 cents
- Place 20: 20 cents

Arrows:
- 0 → 10: 10 cents
- 10 → 0: 10 cents
- 0 → 5: 5 cents
- 5 → 0: 5 cents
- 5 → 15: 10 cents
- 15 → 5: 10 cents
- 0 → 15: 15 cents
- 15 → 0: 15 cents
- 10 → 20: 10 cents
- 20 → 10: 10 cents
- 10 → 15: 5 cents
- 15 → 10: 5 cents

Get 15c candy

Get 20c candy
Petri Nets

- Example: candy machine
  - transition events: “deposit x cents”

get 15c candy

get 20c candy
Petri Nets

- Example: candy machine
  - final Petri net
GSPN

- gspn model name (opt. param. list)
  - 1. List all places and initial marking
    » place-name expr for init num of tokens
  - 2. List all timed trans. and rates
    » trans-name ind expr for rate
    » trans-name dep place-name expr for base rate
  - 3. List instant. trans. and branch weights
    » trans-name ind expr for weight
    » trans-name dep place-name expr for base weight
  - 4. List all place to trans. arcs
    » place-name trans-name expr for mult.
  - 5. List all trans. to place arcs
    » trans-name place-name expr for mult.
  - 6. List all inhibitory arcs

(See language description)
Some general notes

- Recall: reachability graph is Markov.
- Most functions compute CDF of “time to absorption” in reachability graph.
- Must ensure net is “dead" at desired point, e.g.:
  - when 1st token enters “Failure" place,
  - when exactly k-of-N nodes are faulty,
  - when exactly k-of-N nodes are still up,
- Need Inhibitory arcs from “Failure” back to all timed transitions.
  - Causes net to become dead at instant of failure.
  - Otherwise absorption could occur well after failure.
**GSPN**

- **Useful Functions**
  - `etokt (t; model name, place-name {; args})`
    » Expected num of tokens in place at time t.
  - `etok (model name, place-name {; args})`
    » Steady state average of same thing (no t parameter).
  - `preemptyt (t; model name, place-name {; args})`
    » Probability place is empty at time t,
    » Useful for tracking failure modes,
    » Warning: Do not use ( 1 - preemptyt ) !!!
  - `preempty (model name, place-name {; args})`
    » Steady state average of same thing (no t parameter).
**GSPN**

- **Useful Functions**
  - `tput`, `tputt`, `taveputt`
    - Difference is point-in-time of analysis.
    - **Function:**
      - The “throughput” of a transition
      - The “firing rate” of the transition
    - More useful in Performance models (jobs/sec).
    - `tput`: throughput for transition
    - `tputt`: throughput for transition at time `t`
    - `taveputt`: time-averaged throughput of a transition during interval `(0,t)`
GSPN

Useful Functions

- util, utilt, taveutil
  - Difference is point-in-time of analysis
  - Function:
    - The “utilization” of a timed transition
    - The fraction of time it is enabled.
    - Also useful in Performance models (proc. util).
  - util: utilization for a transition
  - utilt: utilization for a transition at time t
GSPN Example

- K-of-N System: Model A
* SYSTEM: K of N SYSTEM. ALTERNATE MODEL DEMONSTRATION
* MODELS: GSPN

epsilon results $1.0 \times 10^{-11}$
epsilon basic $1.0 \times 10^{-13}$
format 3

*------------------------- MODEL DEFINITION -- MODEL A

gspn KofN_A (K,N)
*
* 1. INITIAL MARKING M(0) ...................................... P_NAME TOKENS
  n_up N
  n_dn 0
  end
*
* 2. TIMED TRANSITIONS ........... T_NAME ind RATE (or) T_NAME dep P_NAME RATE
  flt dep n_up lambda
  end
*
* 3. INSTANT. TRANSITIONS .... T_NAME ind WEIGHT (or) T_NAME dep P_NAME WEIGHT
  end
*
* 4. PLACE - TRANS ARCS ................................... P_NAME T_NAME MULT
  n_up flt 1
  end
*
* 5. TRANS - PLACE ARCS ..................................... T_NAME P_NAME MULT
  flt n_dn 1
  end
*
* 6. INHIBITORY ARCS ...................................... P_NAME T_NAME MULT
  n_dn flt (N-K+1)
  end
GSPN Example

- K-of-N System: Model B
*------------------------- MODEL DEFINITION -- MODEL B

gspn KofN_B (K,N)
*
* 1. INITIAL MARKING M(0) ........................................... P_NAME TOKENS
 n_up N
 n_dn 0
 SYS_FAIL 0
end
*

* 2. TIMED TRANSITIONS .......... T_NAME ind RATE (or) T_NAME dep P_NAME RATE
 flt dep n_up lambda
end
*

* 3. INSTANT. TRANSITIONS .... T_NAME ind WEIGHT (or) T_NAME dep P_NAME WEIGHT
 fail_sys ind 1
end
*

* 4. PLACE - TRANS ARCS ......................................... P_NAME T_NAME MULT
 n_up flt 1
 n_dn fail_sys (N-K+1)
end
*

* 5. TRANS - PLACE ARCS .......................................... T_NAME P_NAME MULT
 flt n_dn 1
 fail_sys SYS_FAIL 1
end
*

* 6. INHIBITORY ARCS ............................................. P_NAME T_NAME MULT
 SYS_FAIL flt 1
end
GSPN Example

- K-of-N System: Model C
*------------------------- MODEL DEFINITION -- MODEL C

gspn KofN_C (K,N)
*
* 1. INITIAL MARKING M(0) ...................................... P_NAME TOKENS
n_up     N
n_dn     0
sys_up   1
SYS_FAIL 0
end
*

* 2. TIMED TRANSITIONS ........... T_NAME ind RATE (or) T_NAME dep P_NAME RATE
flt dep n_up lambda
end
*

* 3. INSTANT. TRANSITIONS .... T_NAME ind WEIGHT (or) T_NAME dep P_NAME WEIGHT
fail_sys ind 1
end
*

* 4. PLACE - TRANS ARCS ............................. P_NAME T_NAME MULT
n_up   flt  1
sys_up fail_sys 1
end
*

* 5. TRANS - PLACE ARCS .............................. T_NAME P_NAME MULT
flt   n_dn  1
fail_sys SYS_FAIL 1
end
*

* 6. INHIBITORY ARCS ................................. P_NAME T_NAME MULT
n_up   fail_sys  K
SYS_FAIL flt  1
end