Modeling Tools

- Why prefer one model over another?
  - Clarity
  - Expressiveness/Completeness
  - Analytic possibilities
  - Implementation possibilities

Review of Basic Petri Net Constructs

<table>
<thead>
<tr>
<th>Control Flow</th>
<th>Signalling</th>
<th>Mutual Exclusion (&quot;Conflict&quot;)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Decision (&quot;Choice&quot;)</th>
<th>Resource Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Boundedness of Petri Nets

- Def: A Petri Net is bounded wrt a specific initial state if the set of states reachable from the initial state is finite.

Contrast:

- Boundedness
  - In general, boundedness is a "good thing".
  - It is essential if the system is to be realized with finite memory.
  - It allows the system to be analyzed as a finite-state machine.
  - However, a type of unboundedness can be useful for mathematical analysis, e.g. in the form of firing counters.

When Boundedness is Undesirable

- The system being modeled might not be finite-state, e.g.:
  - Producer/Consumer problem with no limit on production.
  - Artifacts for mathematical analysis, e.g. in the form of firing counters.

Firing Counters

should be unbounded as a necessary condition to be free of deadlock
Boundedness of Individual Places

- Def: A set of places (circular nodes) in a Petri net is simultaneously **unbounded** w.r.t. an initial state if
  - $(\forall n \in \{0, 1, 2, 3, \ldots\})$ there is a reachable state in which each place has $\geq n$ tokens.

Theorem (Karp & Miller, 1966)

- There is an algorithm for deciding whether any given set of places in a Petri net is simultaneously unbounded.
- Note: Karp & Miller did not state this result in terms of Petri nets; they used a different, but more-or-less equivalent model called **Vector Addition Systems**.

Vector Addition Systems

- Consider representing the states of a Petri net as vectors:
  - The vector components are identified with places.
  - Firing a transition has the effect of adding a vector (which may have negative components) to the state (which never has negative components).
  - The enabling rule requires that state components stay non-negative.

Monotonicity: Key Insight for Vector Addition Systems & Petri Nets

- This insight is what makes certain decidability results possible, and is also what creates certain computational limitations on these particular models.
- Let $q_0 \xrightarrow{t_1} q_1 \xrightarrow{t_2} \ldots q_{n-1} \xrightarrow{t_n} q_n$ be a firing sequence leading from a state $q_0$. Let $q'_0 \geq q_0$. Then there is also a firing sequence $q'_0 \xrightarrow{t_1} q'_1 \xrightarrow{t_2} \ldots q'_{n-1} \xrightarrow{t_n} q'_n$ (where $(\forall i) q'_i = q_i + (q'_0 - q_0)$).

Monotonicity

- In particular, if
  $$q_0 \xrightarrow{t_1} q_1 \xrightarrow{t_2} \ldots q_{n-1} \xrightarrow{t_n} q_n$$
  where $q_n > q_0$ (strict inequality in one or more components),
  then there exist sets of reachable states in which those components are without bound.

Reachability-Tree Algorithm

- Construct a tree with the initial state as root.
- Construct successive nodes for each firable transition, as if constructing a state diagram.
- Whenever a node is added that has a predecessor which is pointwise $\leq$ this node, set to $\infty$ any place that is $<$ in the predecessor. If the result is just a repeat, that branch ends.
- This process will terminate. Nodes having sets of places with $\approx$ indicate sets of places that are simultaneously unbounded.
Reachability-Tree Algorithm

Unboundedness of a Counter for a Transition is a Necessary, but not Sufficient, Condition for the Transition to have an Infinite Firing Sequence

Monotonicity Revisited

- Monotonicity is what makes the reachability tree possible.
- Monotonicity also points out a modeling limitation of Petri nets:
  - For unbounded Petri nets, it is not possible to devise a "0-testing structure".

Hypothetical 0-testing structure

Generalizing Petri Nets

- Adding inhibitory arcs:
  - For finite-state systems: ok, can simulate without inhibitory arcs anyway.
  - For unbounded systems: can destroy essential decidability properties (now can simulate a Turing machine)
- "Colored" tokens (see Kurt Jensen, 3 vols., Springer, 1997)
- Program variables
- Enabling predicates on transitions
Inhibition

Simulating Inhibitory Arcs

This transformation only works if the place X in question is bounded by 1.

Simulating Inhibitory Arcs

State-to-State Reachability Problem

- This is the question of deciding whether one state can be reached from another.
- This problem lay open for about 20 years, and was answered in the affirmative by Ernst Mayr.
- The algorithm is complex.
- Richard Lipton earlier showed that this problem is exponential-space hard.