A Framework for Comparing Models of Computation

[1998 by Edward Lee & Alberto Sangiovanni-Vincentelli]

Tobias Welp

University of California at Berkeley
EECS Department

Agenda:

☞ **1. Introduction**

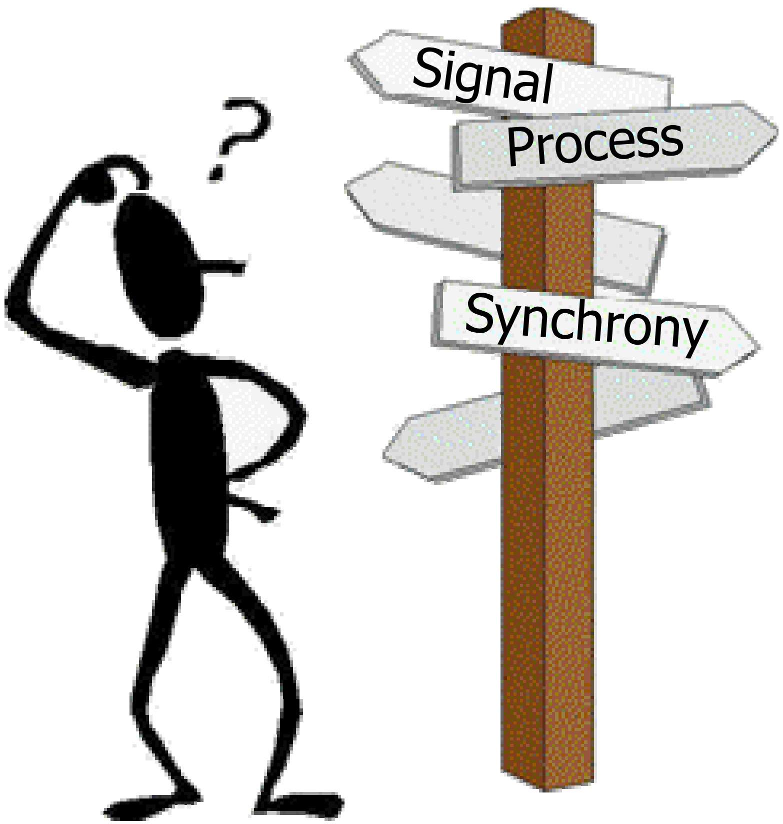
2. Tagged Signal Model

3. Modeling of Time and Causality

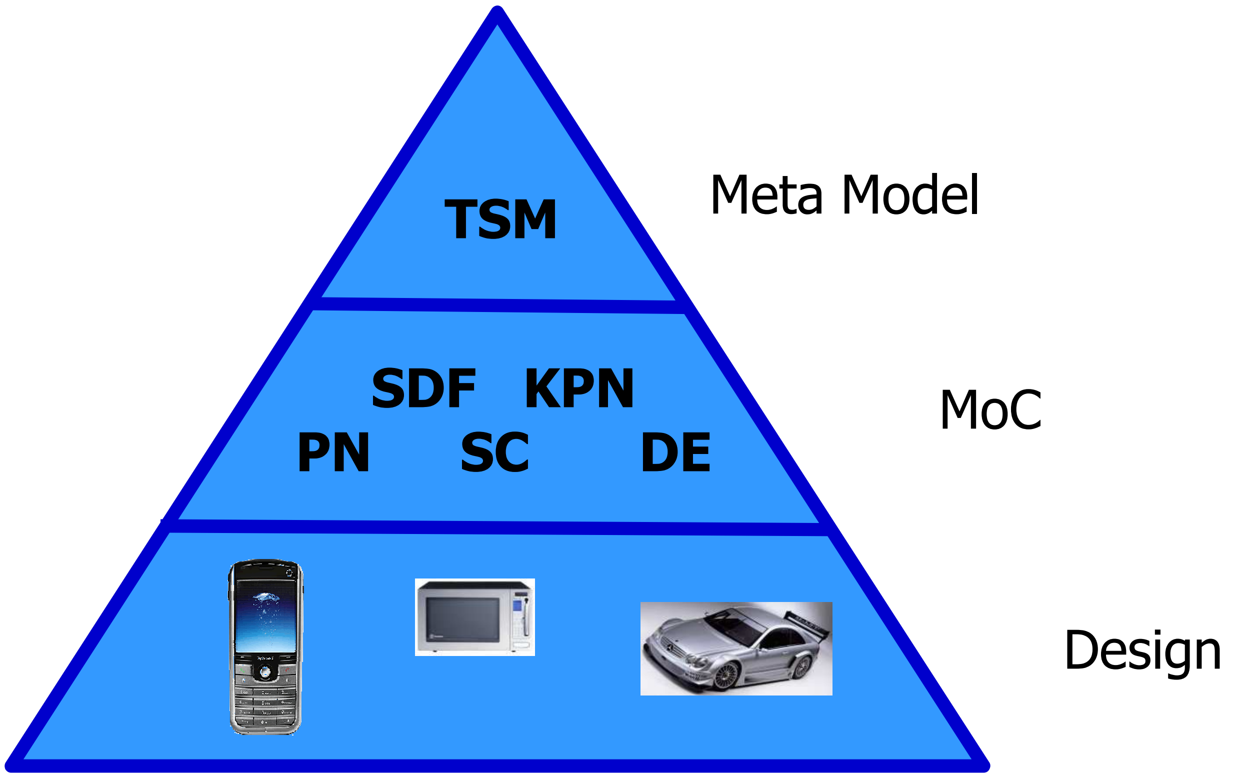
4. Transformations of Tag Systems

5. Summary

Confusion



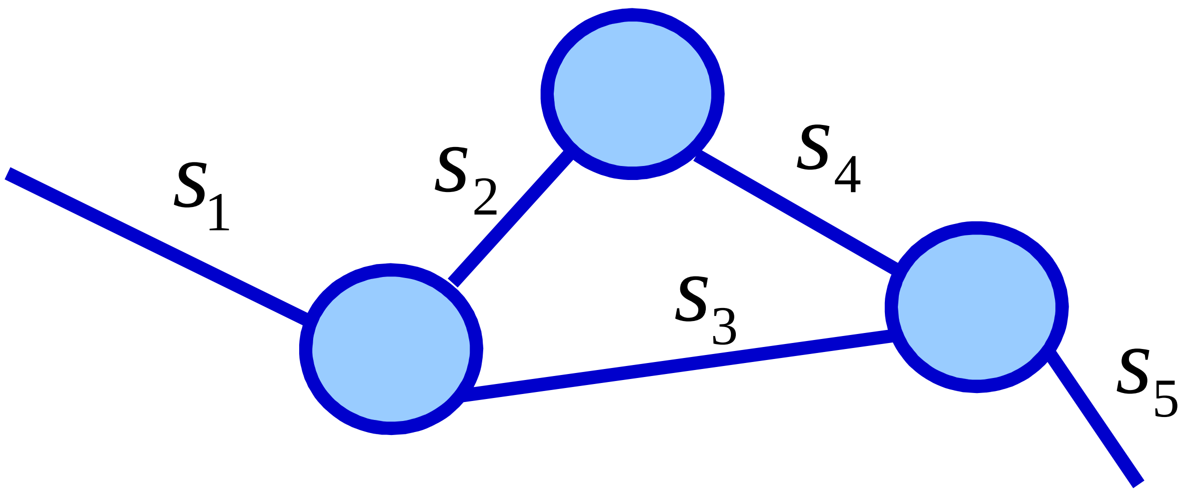
Meta Model



Agenda:

1. Introduction
2. **Tagged Signal Model**
3. Modeling of Time and Causality
4. Transformations
5. Summary

Signals



Preliminary Definitions

Values $V = \{v_1, v_2, \ldots\}$

Tags $T = \{t_1, t_2, \ldots\}$

Event $e \in T \times V$ e.g. $e_1 = (t_2, v_1)$

Signal $s = \{e_1, e_{10}, e_3 \ldots\}$

e.g. $s = \{(t_2, v_2), (t_6, v_3), (t_1, v_3)\}$

Functional Signals

We call a signal S *functional* if for all two events

$$e_i = (t, v_i), e_j = (t, v_j) \in S$$

follows that $v_i = v_j$

Quiz: 1.

$$S = \{(1,3), (2,2), (3,3)\}$$

Functional?

- ☒ Yes
- ○ No

Functional Signals

We call a signal S *functional* if for all two events

$$e_i = (t_i, v_i), e_j = (t_i, v_j) \in S$$

follows that $v_i = v_j$

Quiz: 2.

$$S = \{(\color{blue}{\blacksquare}, a), (\color{blue}{\blacksquare}, b), (\color{cyan}{\blacksquare}, a)\}$$

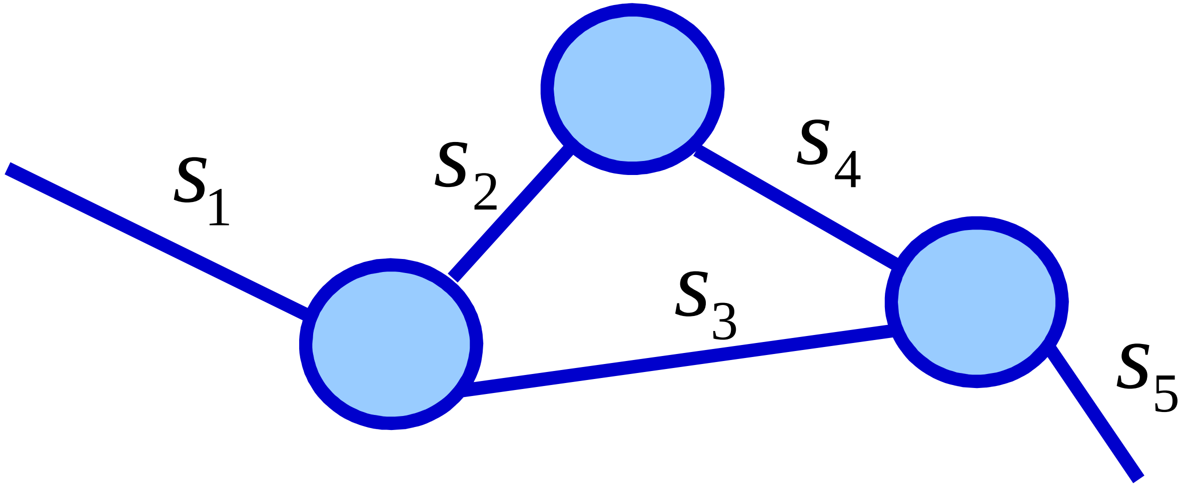
Functional?

○ Yes

☒ No

Tuples of Signals

$$s^N = (s_1, s_2, \ldots, s_N)$$



Set of all those tuples: S^N

Empty Signals

$$\lambda = \varnothing \in S$$

$$\Lambda = \underbrace{\{\lambda, \lambda, \ldots, \lambda\}}_{N} \in S^N$$

By definition: $s \cup \lambda = s$

$$s^N \cup \Lambda = \{s_1 \cup \lambda, s_2 \cup \lambda, \ldots, s_N \cup \lambda\} = s^N$$

Bottom Signal

From the discussion of synchronous languages, we know the bottom symbol

$$\perp$$

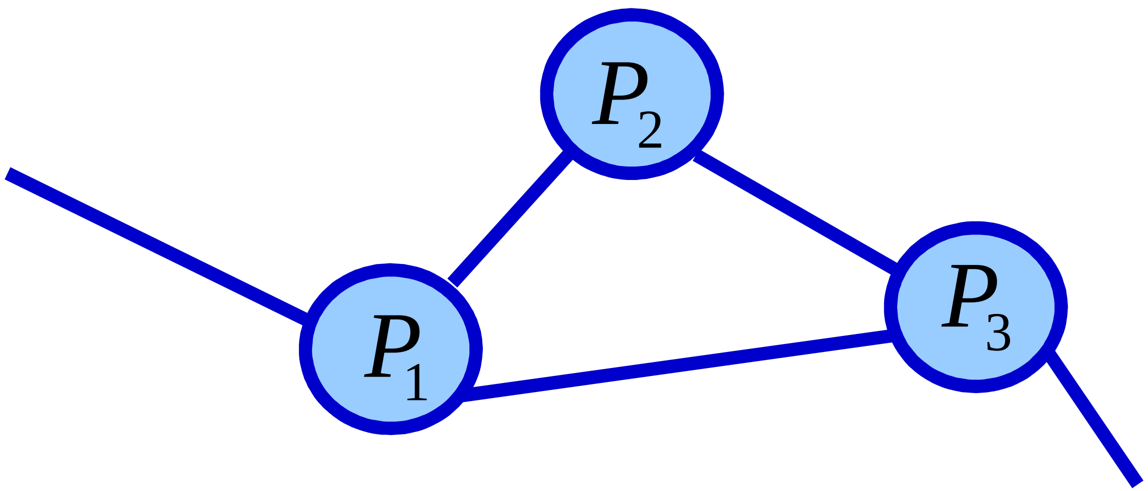
standing for the absence of a signal at a time step.

3. Holds $\lambda = (t_i, \perp)$?

Yes

No

Processes



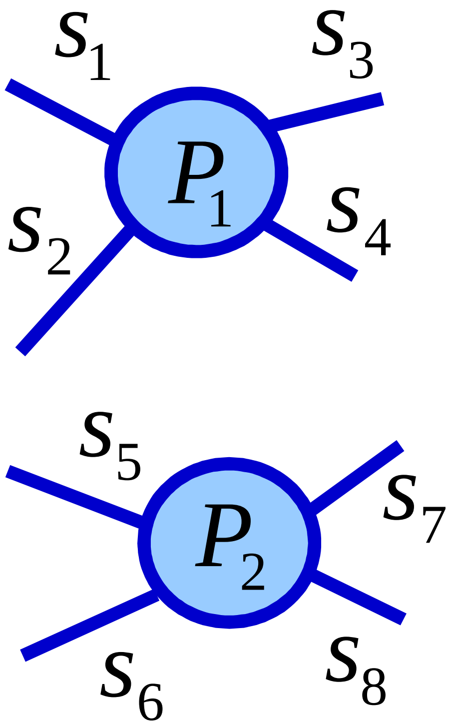
Process

$$P \subseteq S^N$$

s^N satisfies P if $s^N \in P$

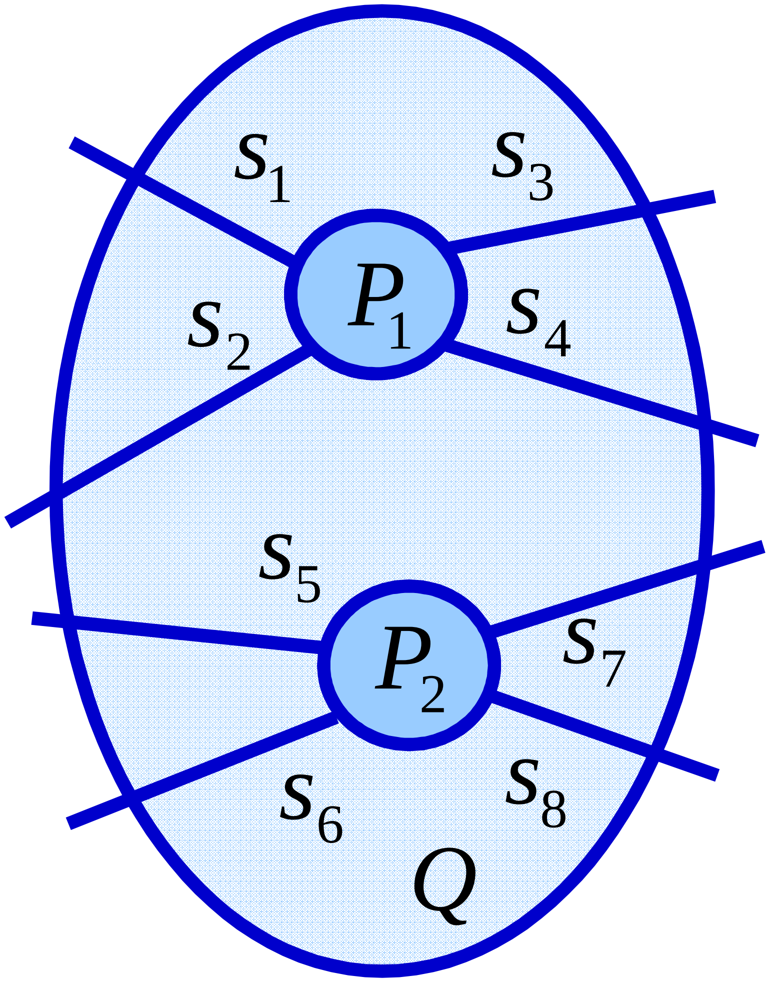
We call such an s^N a behavior of P

Composition



The composite should only allow behaviors allowed both by P_1 and P_2 .

Composition (cont'd)

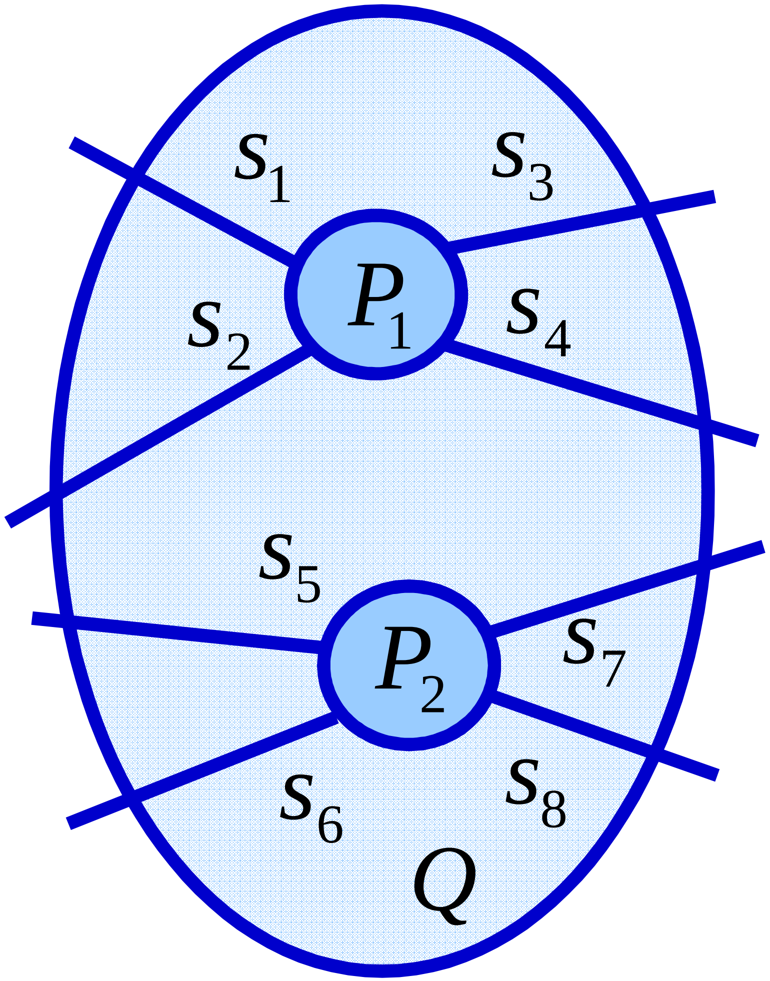


Therefore,

$$Q = P_1 \cap P_2$$

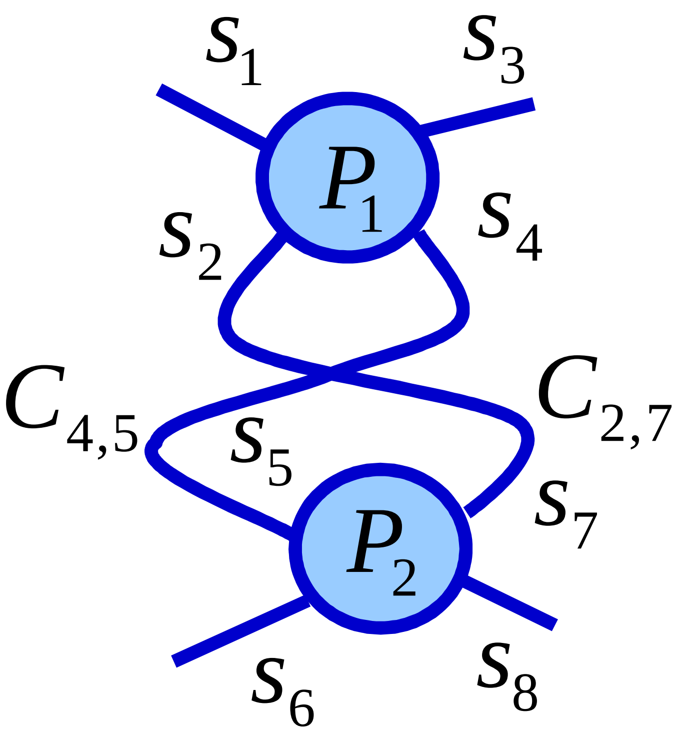
Is there a problem?

Composition (cont'd)



$$Q = P_1 \times P_2$$

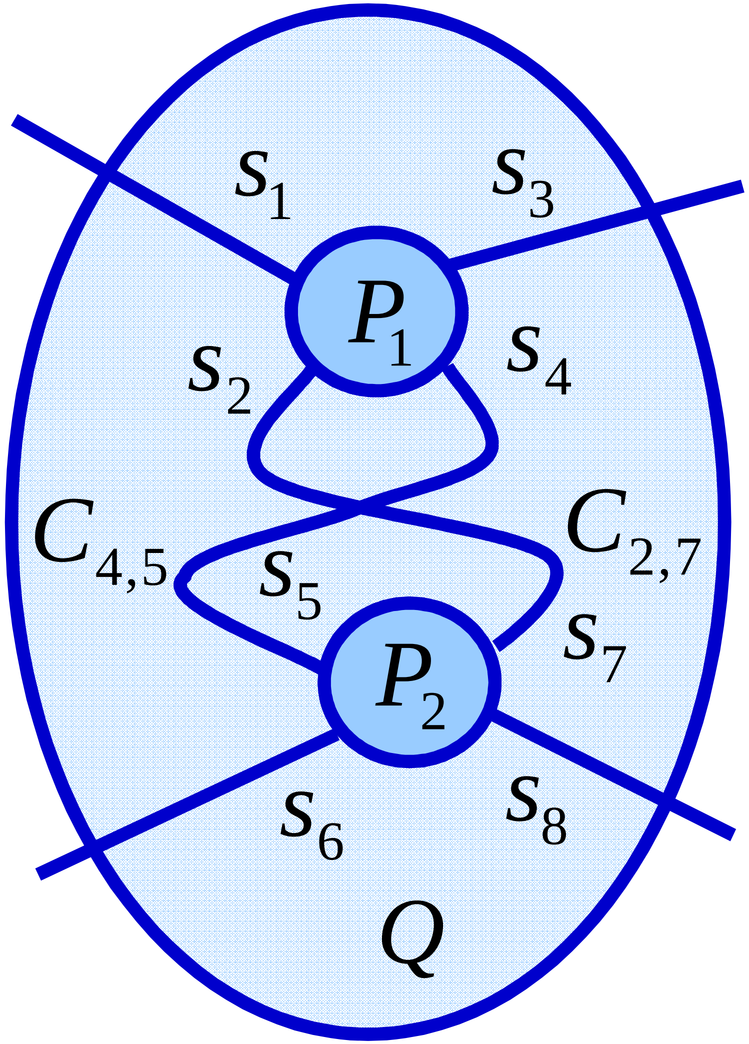
Connections



$$C_{4,5} = \{(s_4, s_5) : s_4 = s_5\}$$

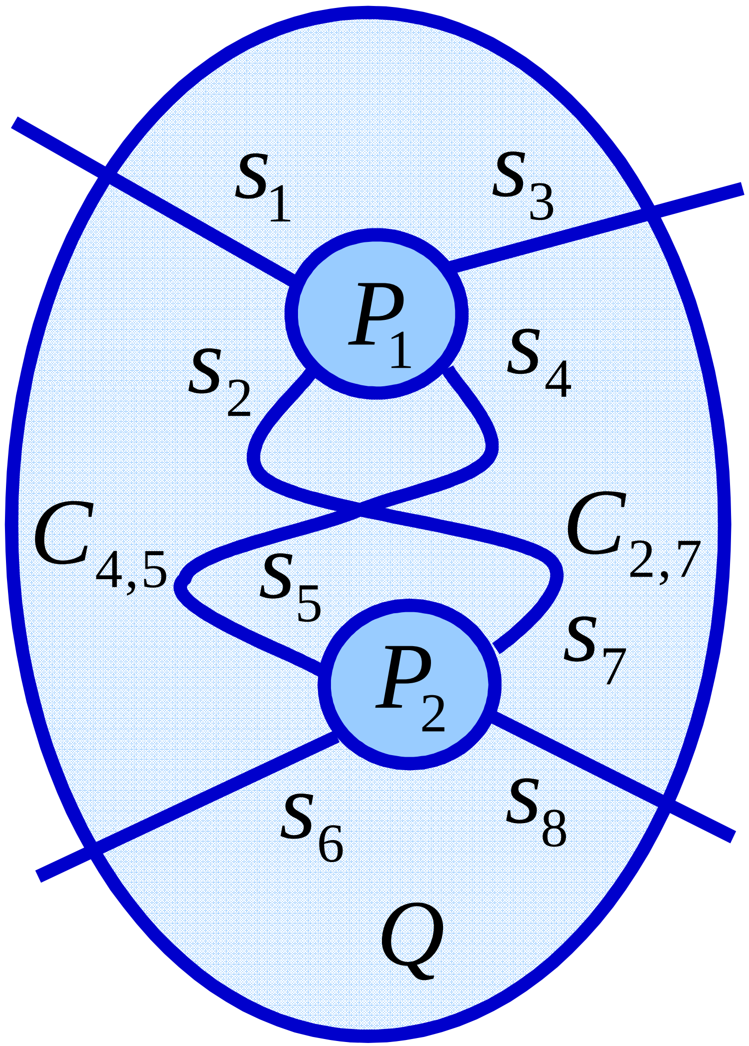
$$C_{2,7} = \{(s_2, s_7) : s_2 = s_7\}$$

Composition of Interacting Processes



$$Q = P_1 \times P_2 \times C_{4,5} \times C_{2,7}$$

Projection



We want to hide

$$s_1, s_2, s_5, s_7$$

Define Projection function:

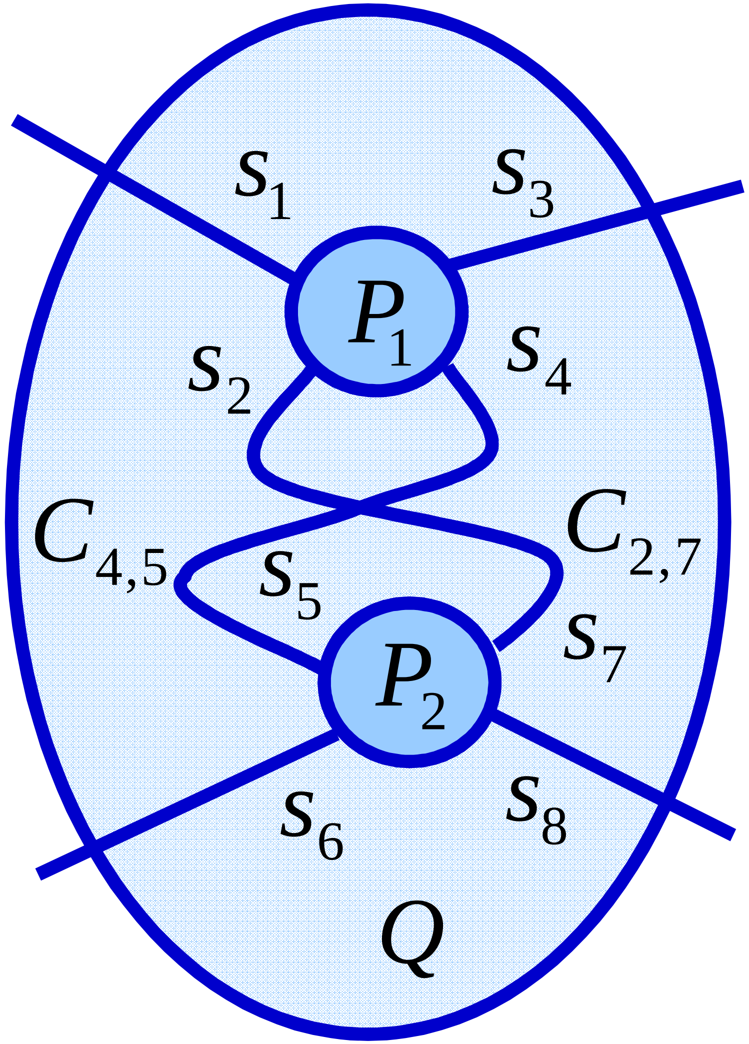
Let $I = (i_1, i_2, \ldots, i_M)$

$$s^N = (s_1, s_2, \ldots, s_N)$$

then $\pi_I : S^N \to S^M$ s.t.

$$\pi_I(s^N) = (s_{i_1}, s_{i_2}, \ldots, s_{i_M})$$

Projection



We want to hide

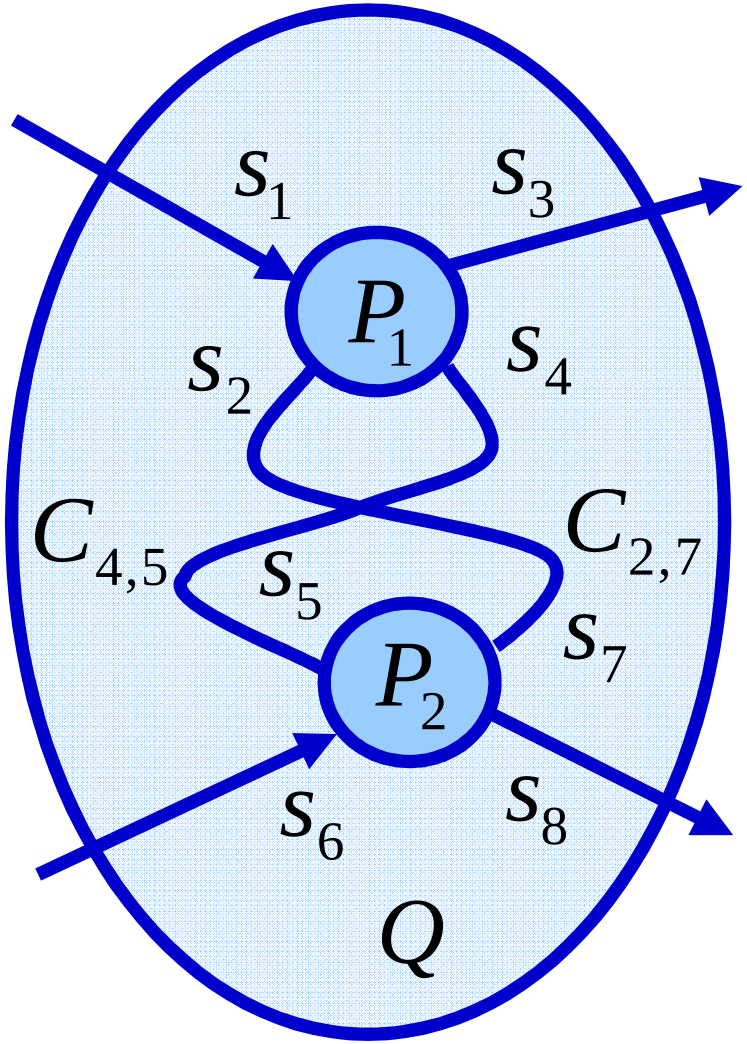
$$s_2, s_4, s_5, s_7$$

Projection function:

Let $I = (1,3,6,8)$

then $Q' = \pi_I(Q)$

Inputs



Set of all possible Inputs

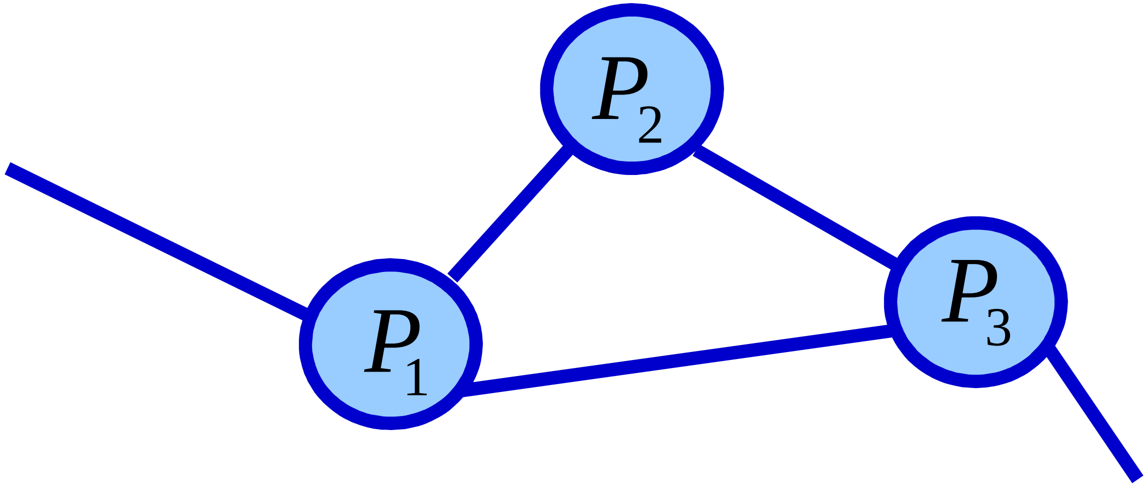
$$B$$

One specific input:

$$A$$

$A \times P$ are the possible behaviors

Determinacy



Functional Process

Let I be the indices of the inputs, let O be the indices of the outputs.

Then, a process is functional if

$$\pi_I(s) = \pi_I(s') \Rightarrow \pi_O(s) = \pi_O(s)$$

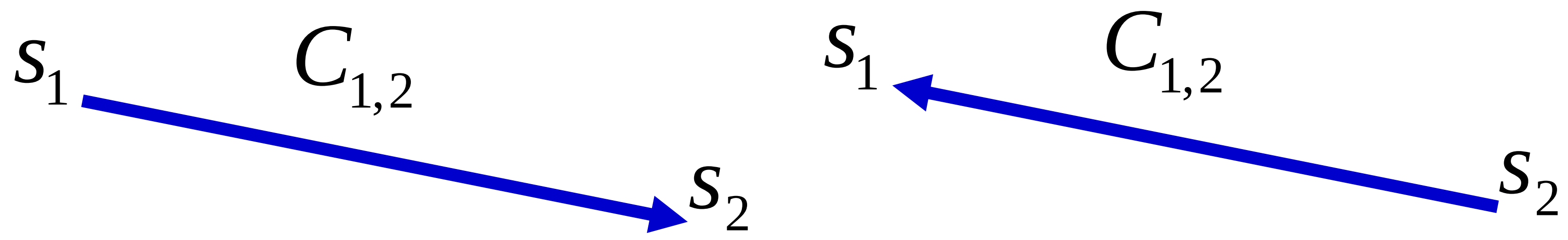
$$\forall s, s' \in P$$

Then there exists F s.t. $\pi_O(s) = F\left(\pi_I(s)\right)$

$$\forall s \in P$$

Functional Process (cont'd)

A process may be functional with respect to different assignments for I and O:

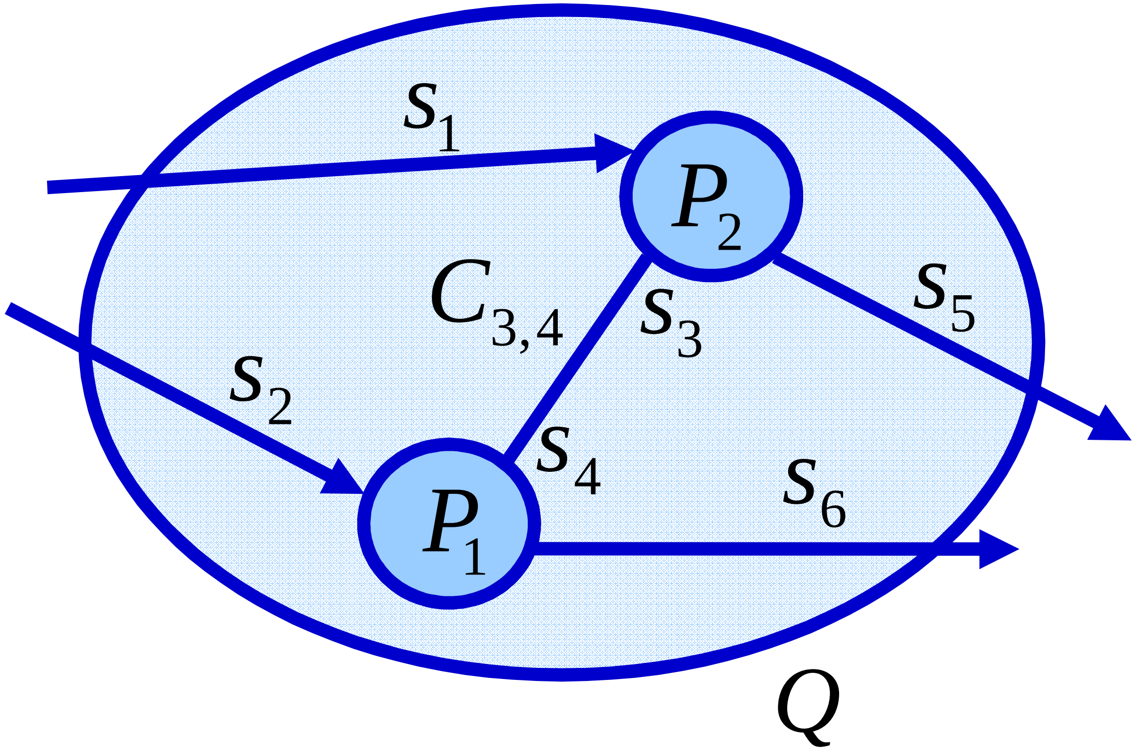


Determinacy

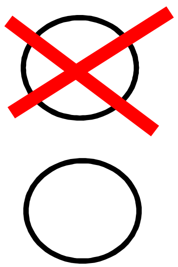
A process is determinate iff

$$\left| A \times P \right| \leq 1$$

Determinacy



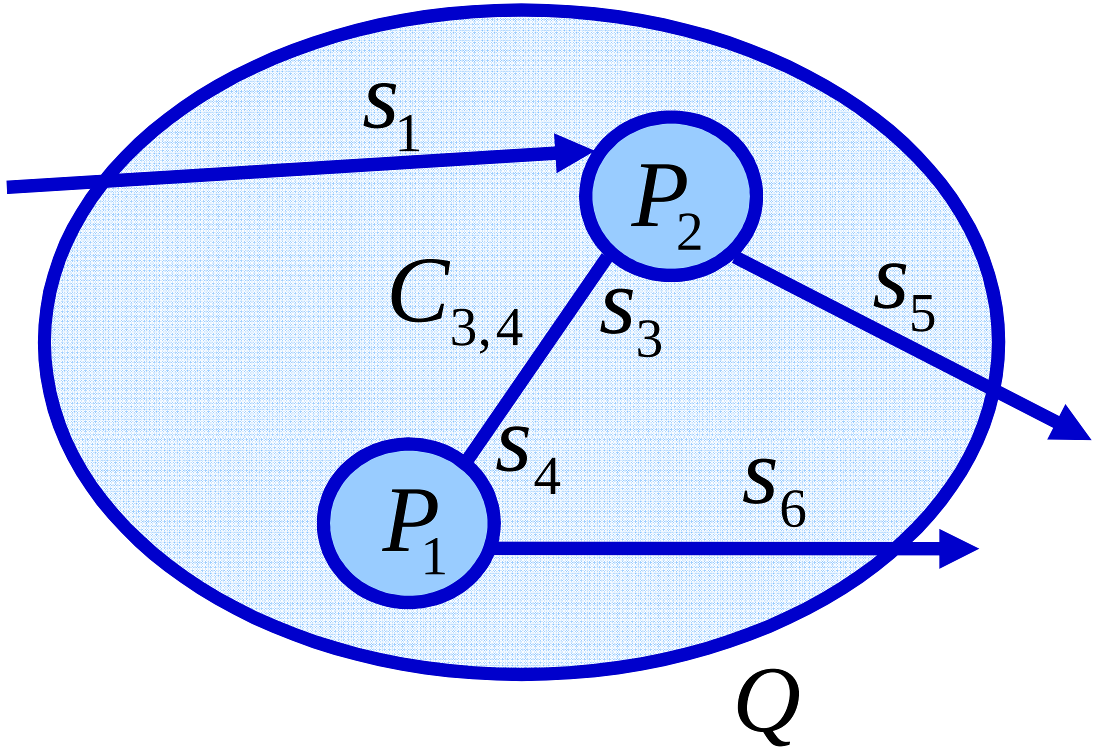
4. Q determinant if P_1 and P_2 determinant?



Yes

No

Determinacy



5. Q determinant if P_1 and P_2 determinant?

- [x] Yes
- [ ] No

Agenda:

1. Introduction
2. Tagged Signal Model
3. ☞ **3. Modeling of Time and Causality**
4. Transformations of Tag Systems
5. Summary

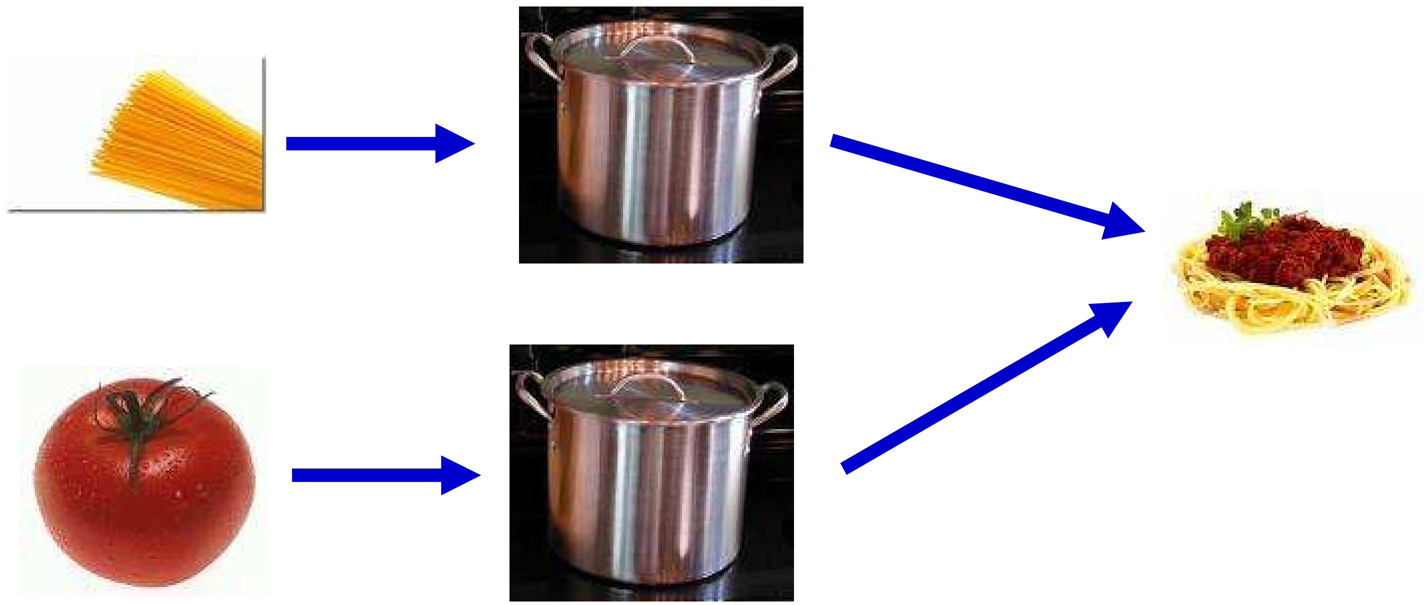
The Role of Tags

The natural interpretation

P_K = {((, 0),(, 15 min))}

Is this interpretation generally useful?

Global Order vs. Partial Order



It does not matter in which order the items are cooked. Maybe they are cooked simultaneously?

Establishing Order

Ordering of tags:

$$t_1 \le t_2$$

Relation is

- reflexive
- transitive
- antisymmetric

T with such a relation $\le$ is called ordered.

It is globally ordered if $\forall i, j : t_i \le t_j$ or $t_j \le t_i$

partially ordered otherwise.

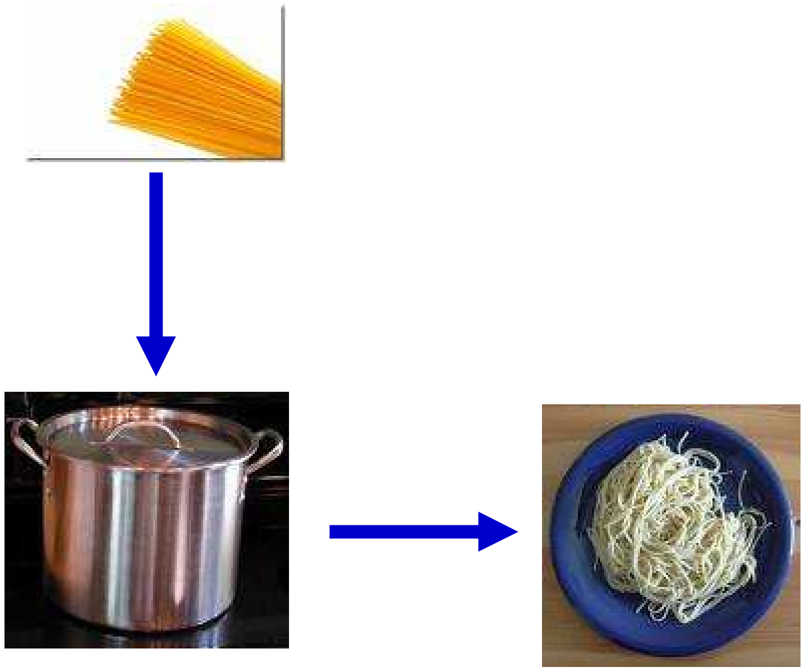
Establishing Order (cont'd)

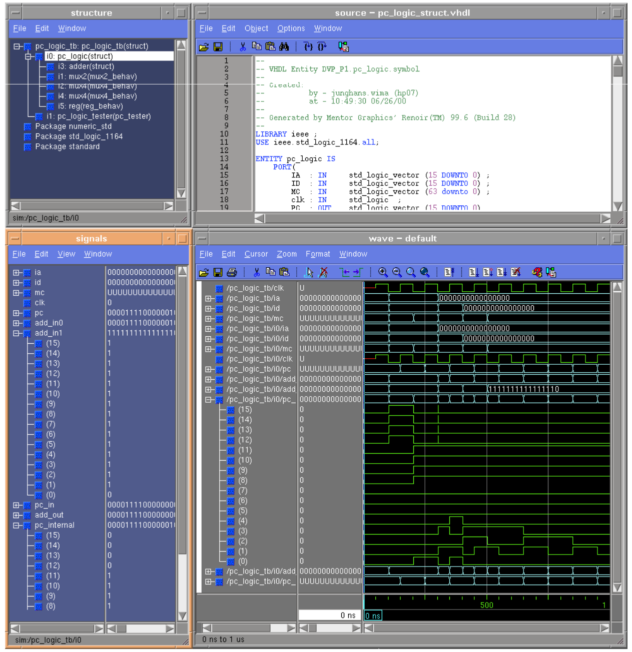
Event ordering depends on tag ordering:

$$e_2 = (t_2, v_2)$$
$$e_1 = (t_1, v_1)$$

$$t_1 \leq t_2 \rightarrow e_1 \leq e_2$$

Timed Models of Computation





Metric Time

The tags adhere to a metric:

$$d(t,t') = d(t',t)$$

$$d(t,t') = 0 \Leftrightarrow t' = t$$

$$d(t,t') \geq 0$$

$$d(t,t') + d(t',t'') \geq d(t,t'')$$

Frequently used: $d(t,t') = |t_1 - t_2|$

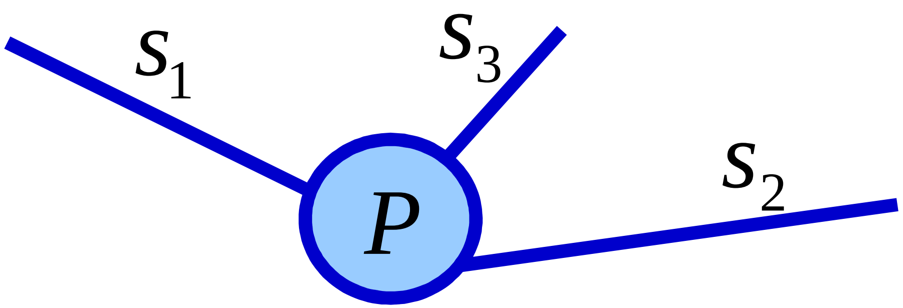
Discrete Event

Given: P

$s^N \in P$

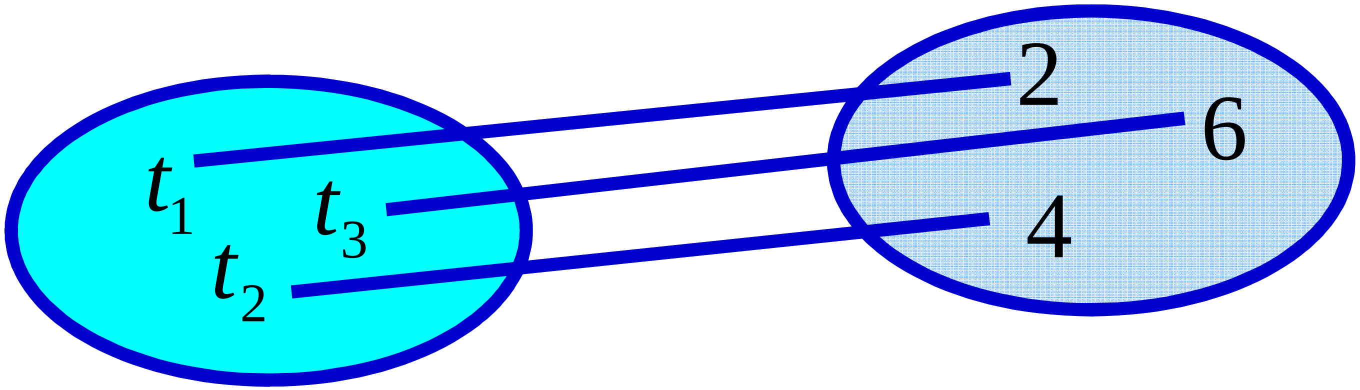
$t(s^N) \subseteq T$

$t(s^N)$ is order isomorphic to a subset of integers.



$t_1 < t_2$

$t_2 < t_3$



Circuit Simulators

	t_0	$t_0+\Delta$	$t_0+2\Delta$	t_1
a	1	1	1	1
b	1	1	1	1
x	0	0	1	1
y	0	1	1	1

```
architecture foo_a of foo_e is
begin
      x <= a and y;
      y <= a or b,
end foo_a;
```

Circuit Simulators (cont'd)

	t_0	$t_0+\Delta$	$t_0+2\Delta$	t_1
a	1	1	1	1
b	1	1	1	1
x	0	0	1	1
y	0	1	1	1

```
architecture foo_a of foo_e is
begin
      x <= a and y;
      y <= a or b;
end foo_a;
```

The tag consists of two parts:

$$t = (t_1, t_2) \in N \times N$$

Simulation Time (t_1) — Delta Time (t_2)

Circuit Simulators (cont'd)

There is no order isomorphism between $N \times N$ and a subset of integers.

This is because there can be an infinite number of delta steps between two simulation steps. Consider, e.g.

```
architecture foo_a of foo_e is
begin
      a <= not a;
end foo_a;
```

Circuit Simulators (cont'd)

It can be proven that if a fix point is found, that it is unique

→ Determinacy

The fix point iteration converges under the condition that there is a delay on feedback loops.

Synchrony

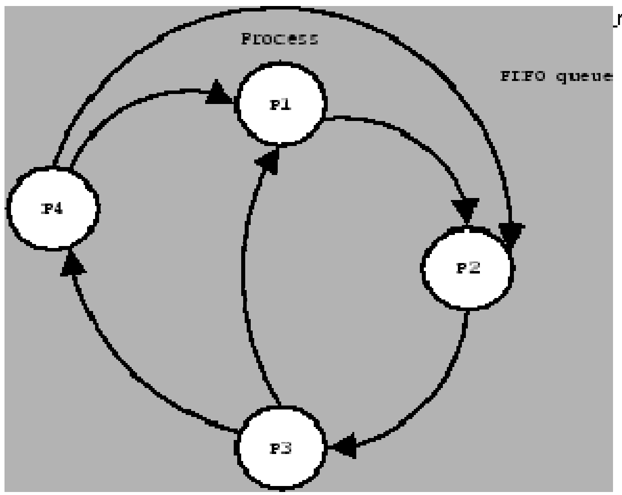
Greek: sun ↔ together
khronos ↔ time

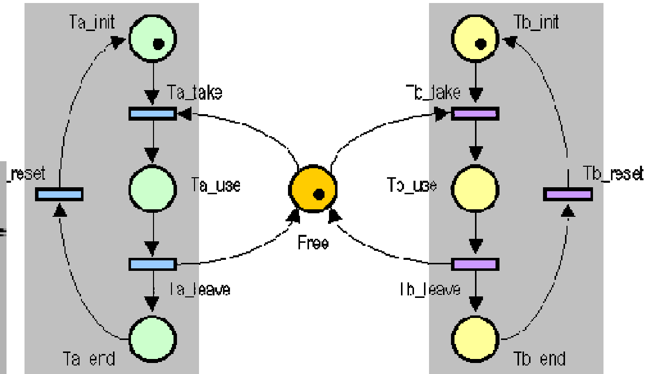
- Two events are synchronous have they have the same tag.
- Two signals are synchronous if for each event in the one signal, there is a synchronous event in the other signal.
- A tuple of signals is synchronous if each pairs of signals are synchronous.
- A process is synchronous if all its behaviors are synchronous.

Synchrony: Results

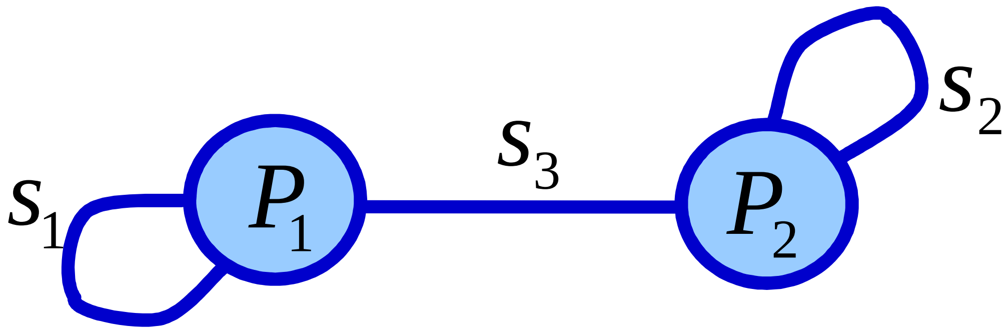
- Synchronous languages like Esterel are synchronous if $\perp$ is considered to be a value.
- Synchronous Data Flow is **not** synchronous.

Untimed Models of Computation





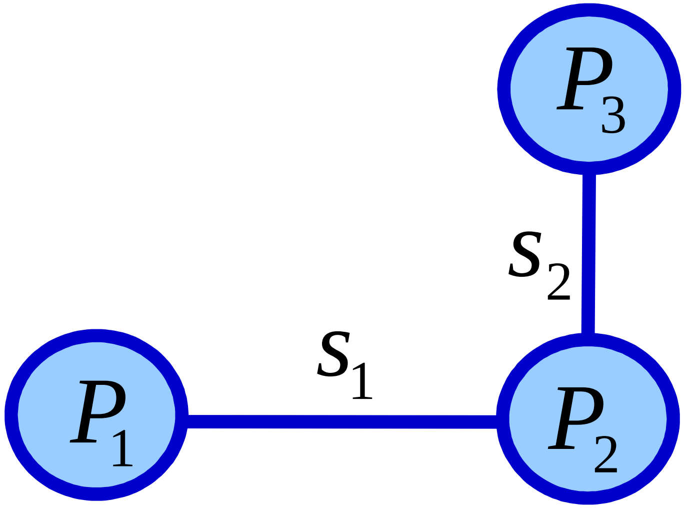
Rendezvous



$T(s_1), T(s_2), T(s_3)$ are each totally ordered

Rendezvous: $T(e_1) = T(e_2) = T(e_3)$

Kahn Process Networks

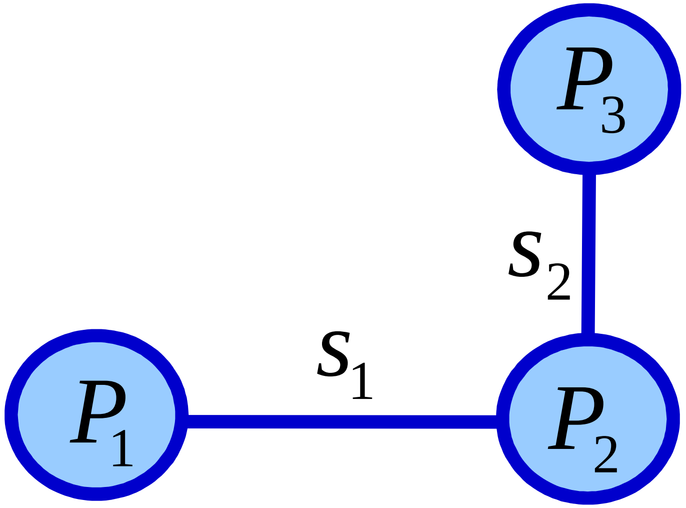


Channels of Kahn Process Networks are FIFO. Therefore,

$$T(s_1), T(s_2)$$

are totally ordered.

Kahn Process Networks



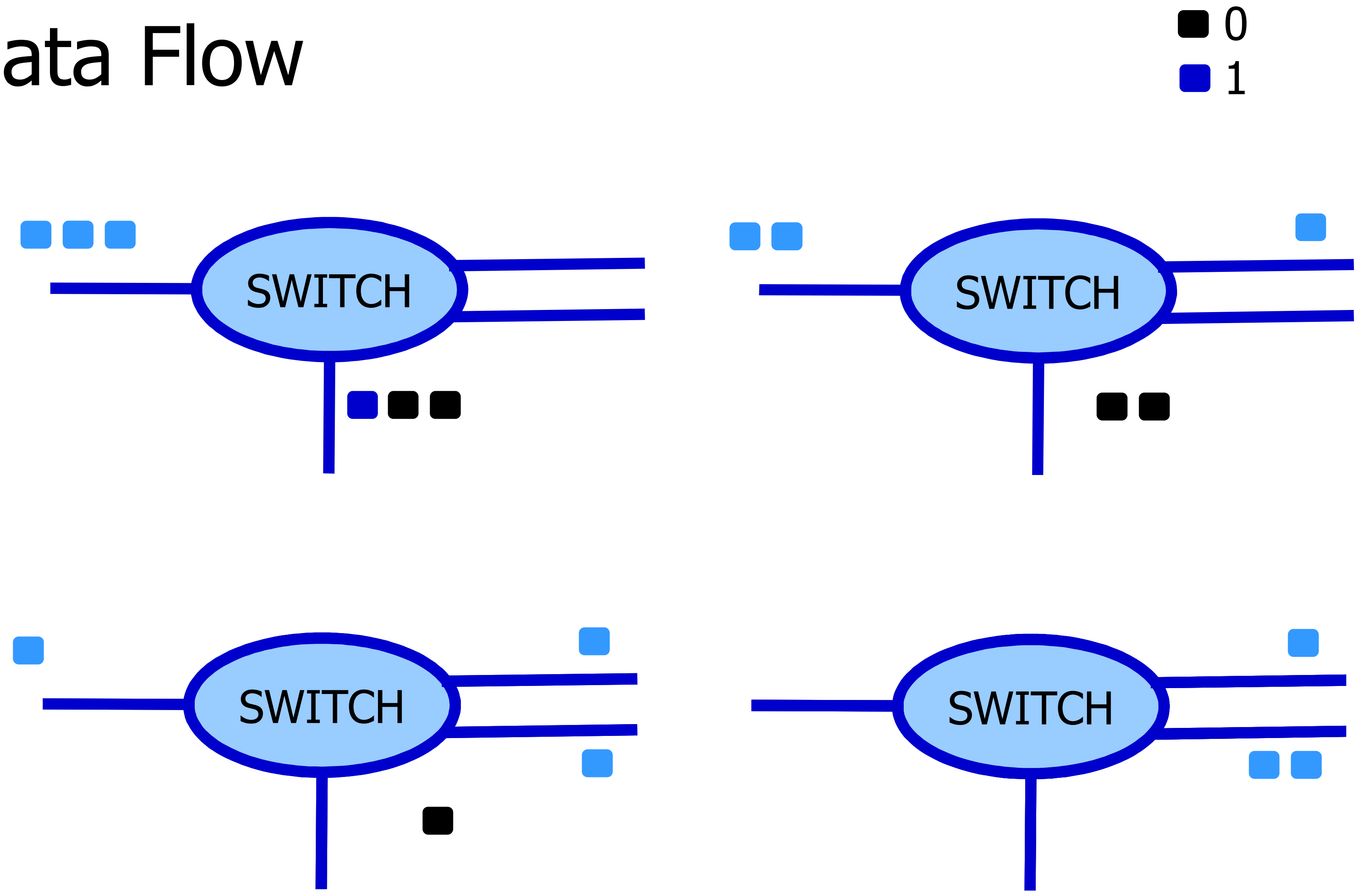
Define sequence $\sum(s)$

Then, a Kahn Process is defined by:

$$P = \left\{ s^N \in S^N : F\left(\sum \pi_I\left(s^N\right)\right) = \sum \pi_O\left(s^N\right) \right\}$$

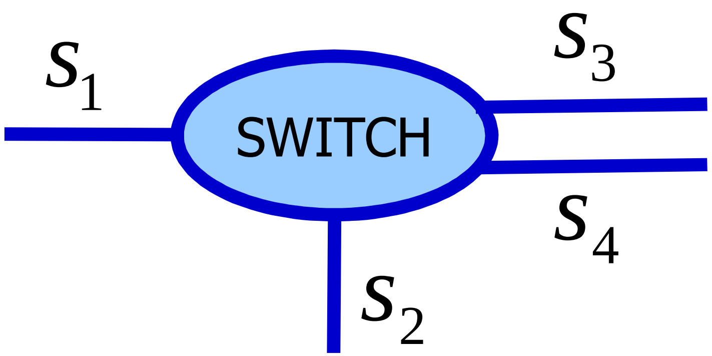
What is the constraint on F s.t. the KPN is deterministic?

Data Flow



Data Flow

$$b_k = \sum_{i=1}^{k} v_{2,i}$$



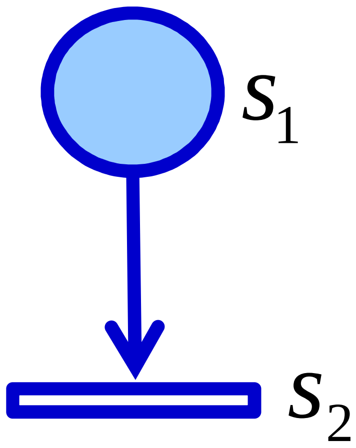
$$v_{3,m} = v_{1,b_m}$$

$$v_{4,n} = v_{2,n-b_n}$$

$$e_{3,m} > e_{1,b_m}$$

$$e_{4,n} > e_{2,n-b_n}$$

Petri Nets



Very similar to data flow.

Notable exception: $T(s_1), T(s_2)$

are **not** totally ordered

Semantics in TSM: $f : s_2 \rightarrow s_1 \quad f(e) < e_2 \quad \forall e \in s_2$

Agenda:

1. Introduction
2. Tagged Signal Model
3. Modeling of Time and Causality
4. **Transformations of Tag Systems**
5. Summary

System Refinement

Suppose we have two tag systems T, T' and an order preserving mapping

$$f : T \to T'$$

Then we can refine the system by replacing each tag t in each event in every process by

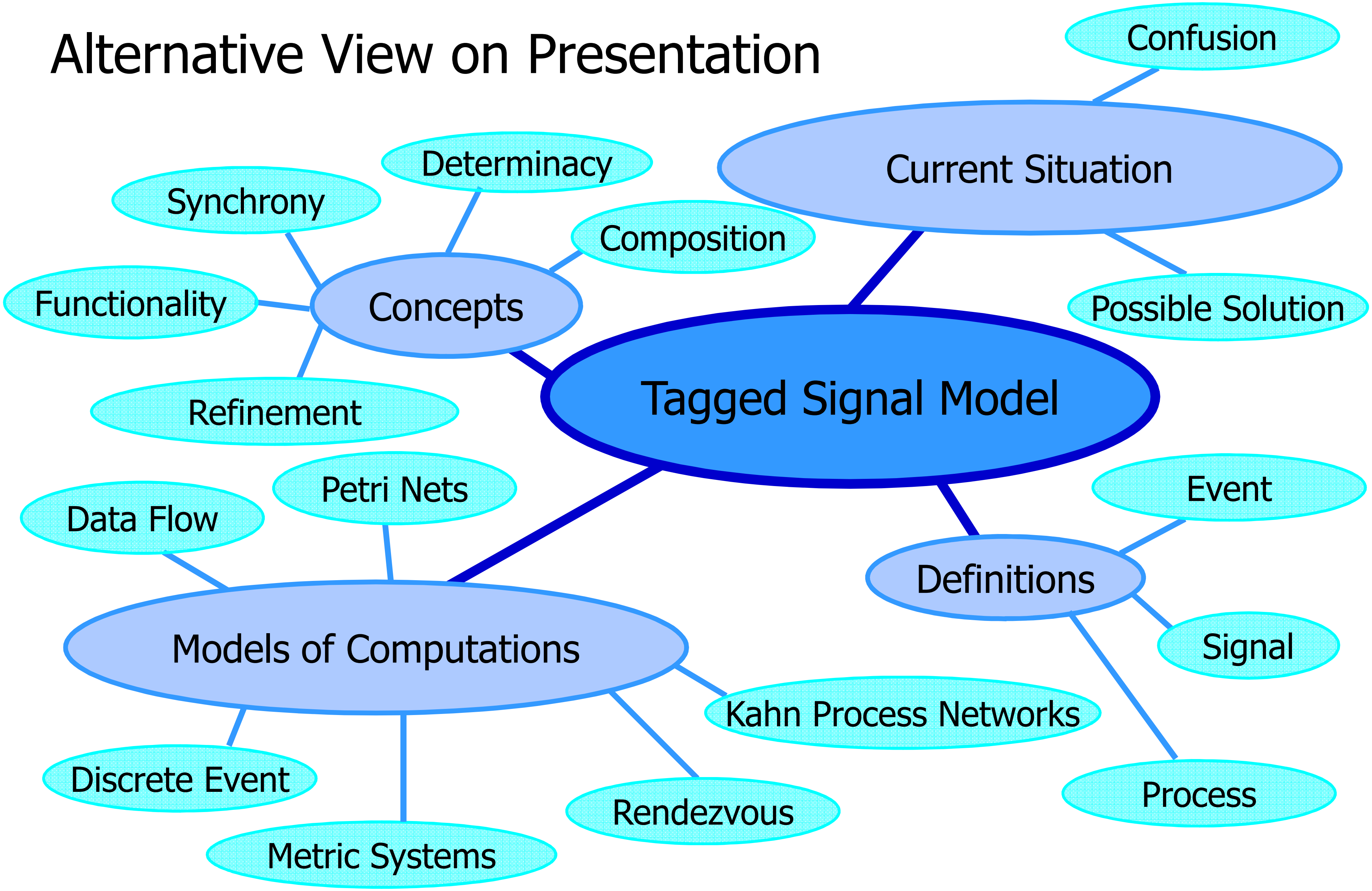
$$f(t)$$

Obviously, this preserves the designs properties.

Agenda:

1. Introduction
2. Tagged Signal Model
3. Modeling of Time and Causality
4. Transformations of Tag Systems
5. **Summary**

Alternative View on Presentation



Thank you

for your attention

Slides: twelp@berkeley.edu

References:

1. Edward A. Lee, Alberto Sangiovanni-Vincentelli: *A Framework for Comparing Models of Computation*, TCAD 1998