# Weakly Bounded Petri Nets

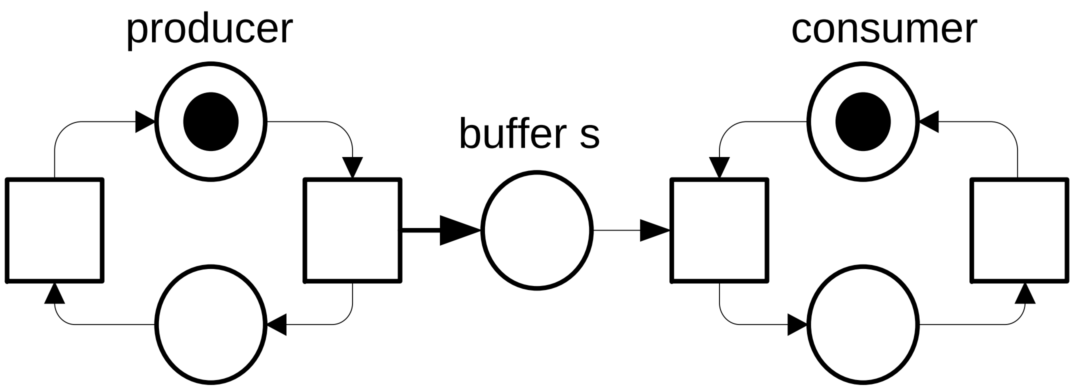
Jörg Desel

KU Eichstätt-Ingolstadt

Evry, March 5th, 2010, séminaire MeFoSyLoMa

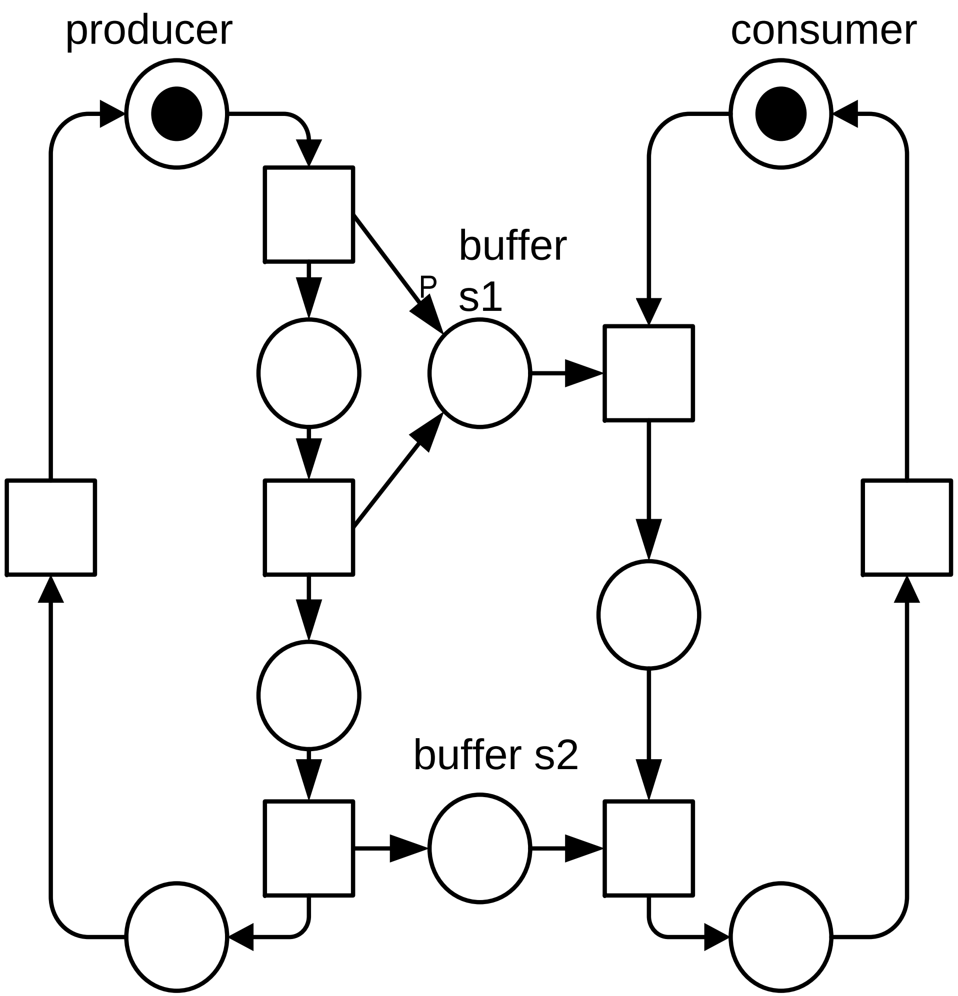
# Weakly Bounded Petri Nets

**Attention:**
**Work in Progress!**



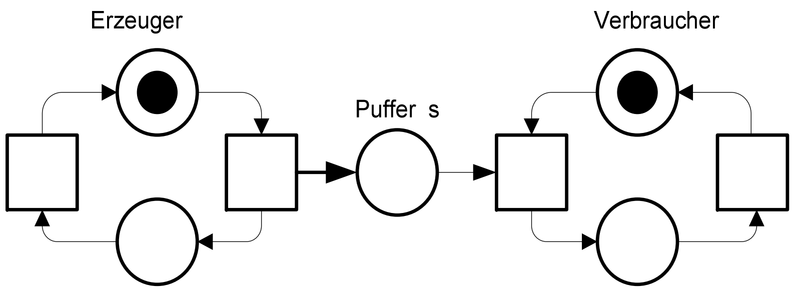

Is this Petri net bounded?

No, the place s is unbounded!

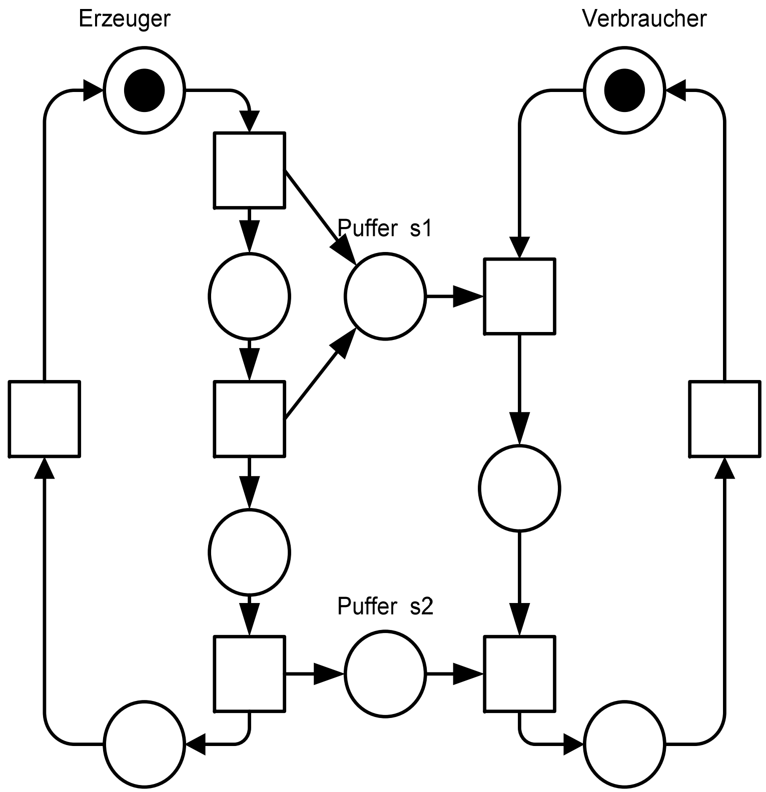



Places s1 and s2 are unbounded

The place s1 is **„worse unbounded“**



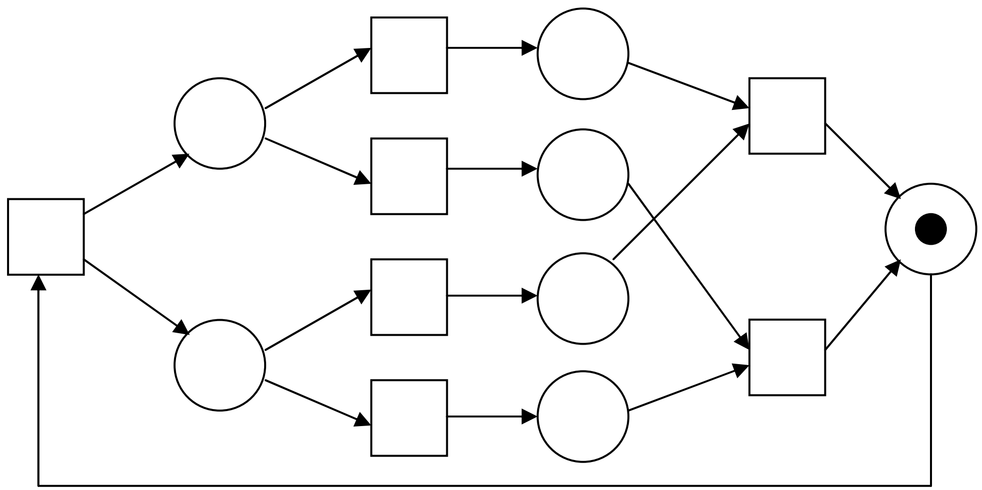

weakly bounded




**not**
weakly bounded

Why **weakly bounded ?**

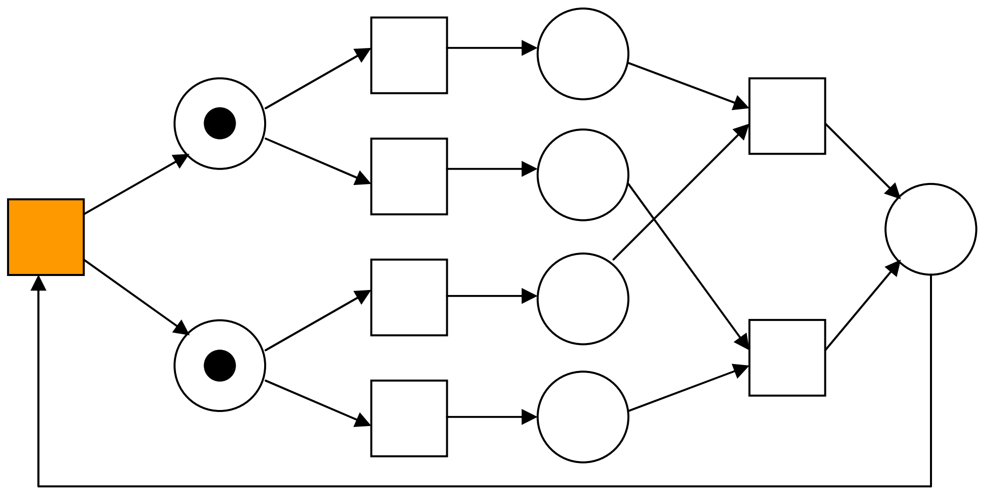
Analogy to **weak liveness:**



**not** live

Why **weakly bounded ?**

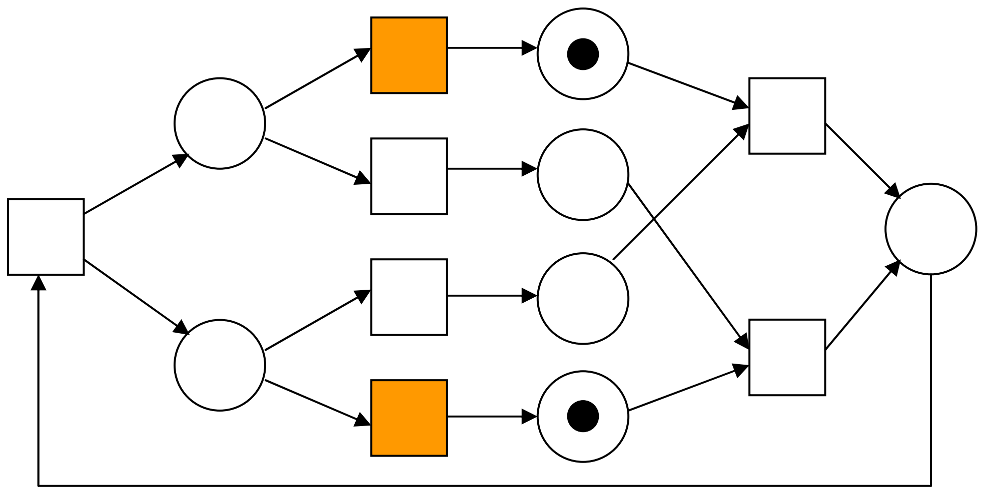
Analogy to **weak liveness:**



**not** live

Why **weakly bounded ?**

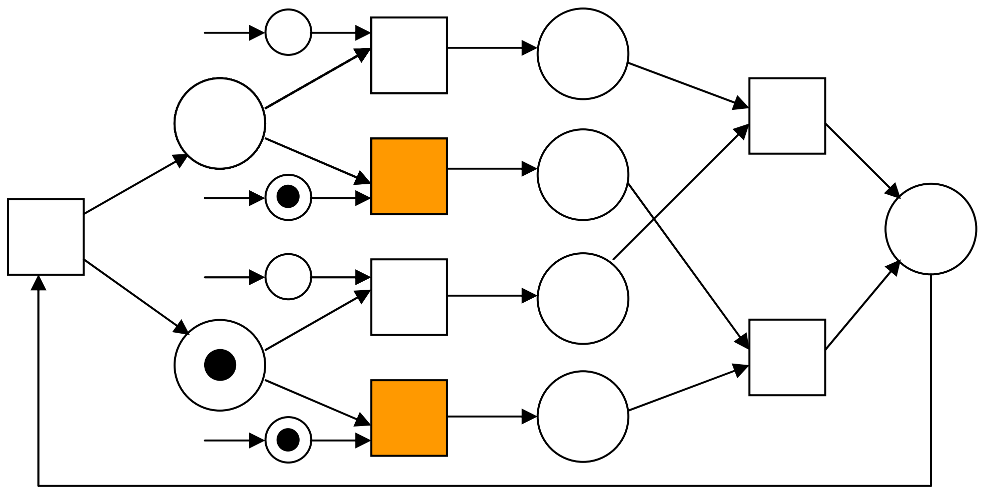
Analogy to **weak liveness:**



**not** live

Why **weakly bounded ?**

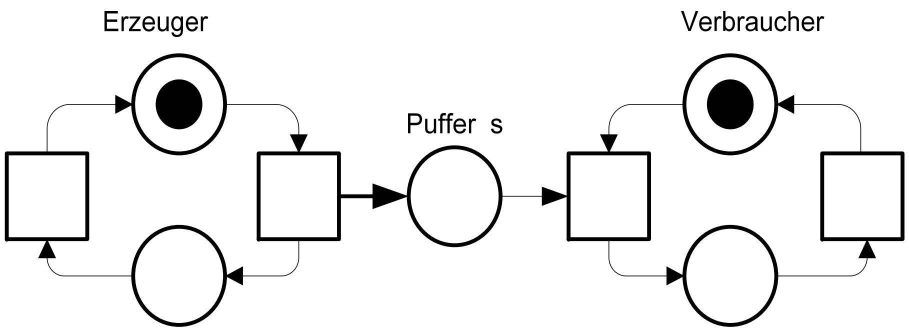
Analogy to **weak liveness:**



**weak liveness:**
*choices* can be controlled such that the controlled net behaves lively

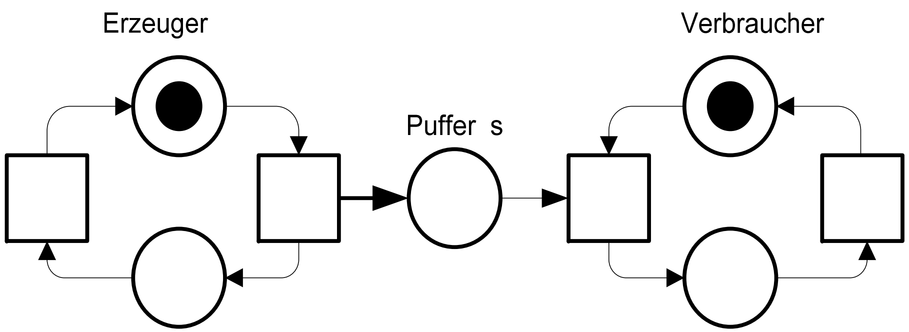
Why **weakly bounded ?**

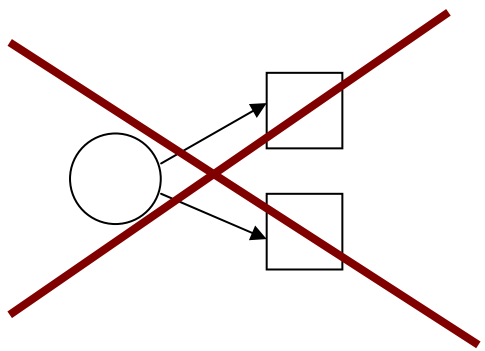
Analogy to **weak liveness:**




**weak boundedness**: *<u>concurrency</u>* can be controlled such that the controlled net behaves boundedly

# Petri nets without branching places



# Petri nets without branching places

## Suggestion for a definition of weak boundedness:

- We are allowed to determine the (relative) speed of the components

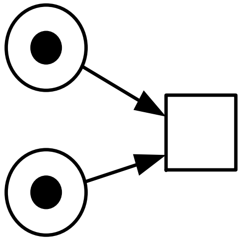
  For each occurrence sequence, we are allowed to change the order of concurrent transitions

  Necessary requirement:
  **no component is (or becomes) inactive**
  i.e., we assume progress (weak liveness)

# Petri nets without branching places

## Progress assumption:

If transition t is enabled then t eventually occurs



An occurrence sequence will be called **progressing**,
if it satisfies the progress assumption

# Petri nets without branching places

## Definition

A place s is called **weakly k-bounded**
if each progressing occurrence sequence can be permuted
such that in the resulting occurrence sequence
s carries never more than k tokens

A Petri net is called **weakly k-bounded**
if all ist places are weakly k-bounded.

A Petri net is called **weakly bounded**
if each progressing occurrence sequence can be permuted
such that in the resulting occurrence sequence
only finitely many markings are reached

# Petri nets without branching places

## Observation

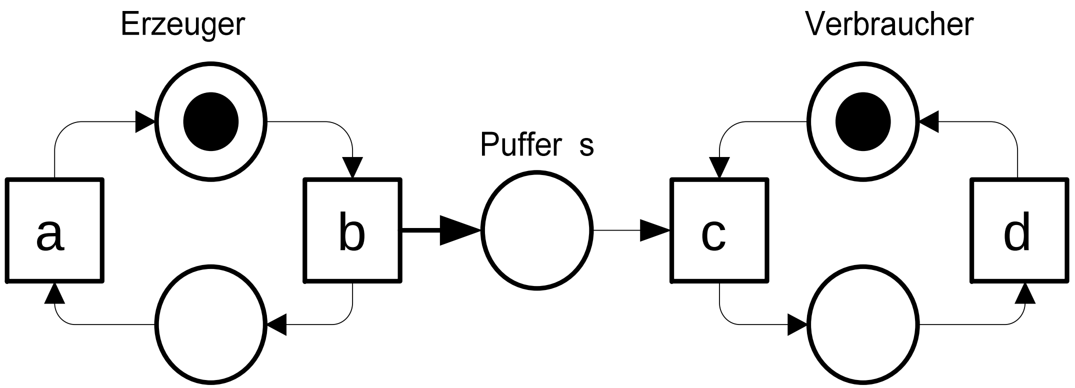
**weak k-boundedness** implies **weak boundedness**
(if the set of places is finite)

**weak boundedness** implies **weak k-boundedness** for some k

This does not hold if further assumptions are made
for example:

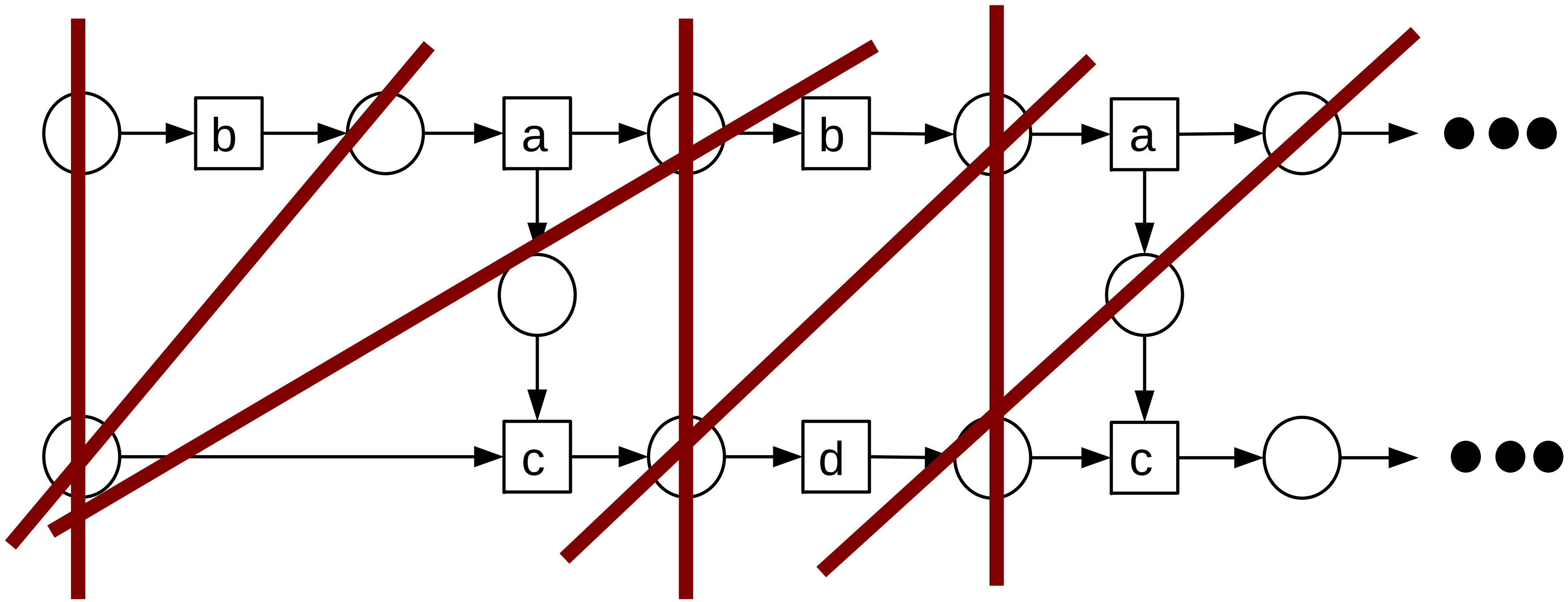
- if the consumer is generally faster than the producer
  (talking about the average speed)

# Example

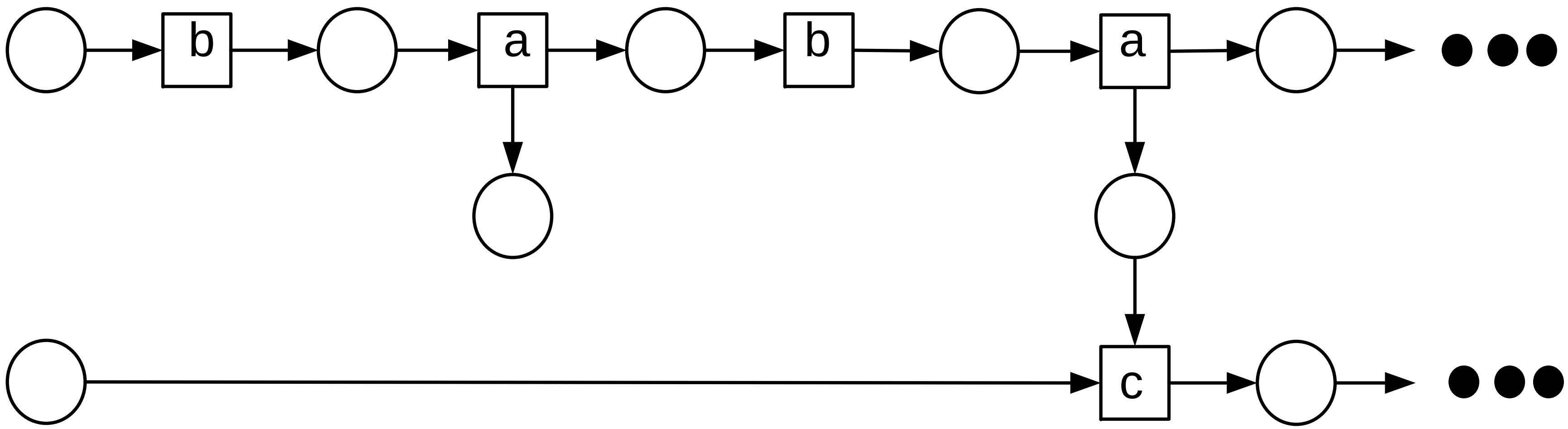



Occurrence sequence: b a b a c d b a b a c d b a b a c d …
Permutation: b a c d b a c d b a c d b a c d b a …

# Definition using partially ordered occurrence nets?



**but ….**



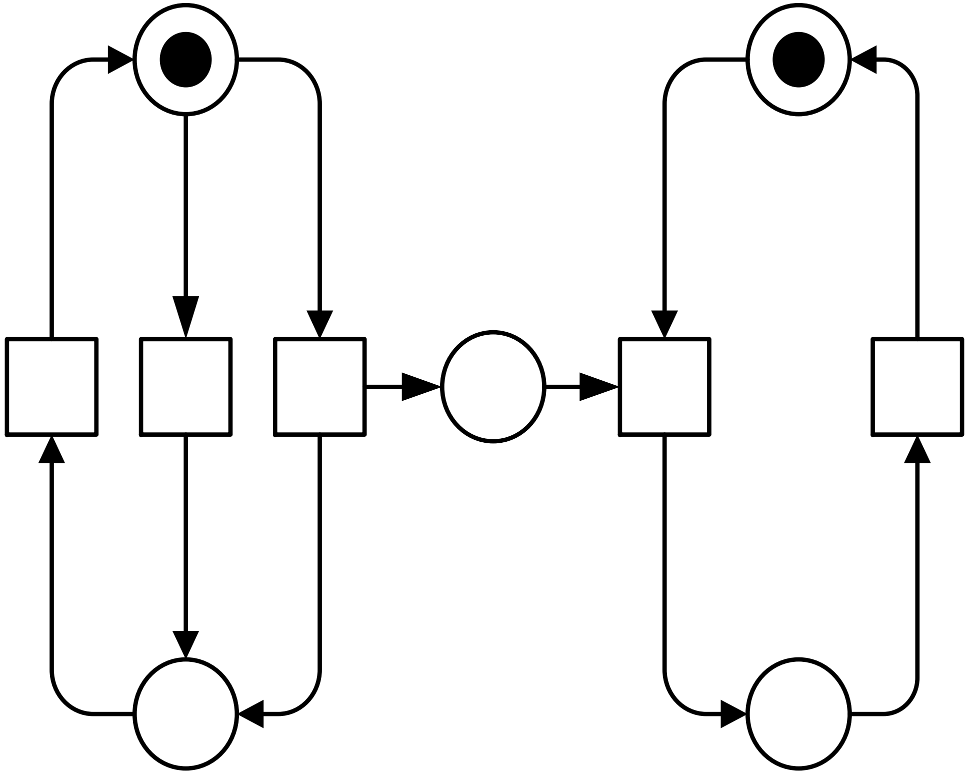
# Petri nets with branching places

**Idea (Cortadella, Kondratyev et. al.):**

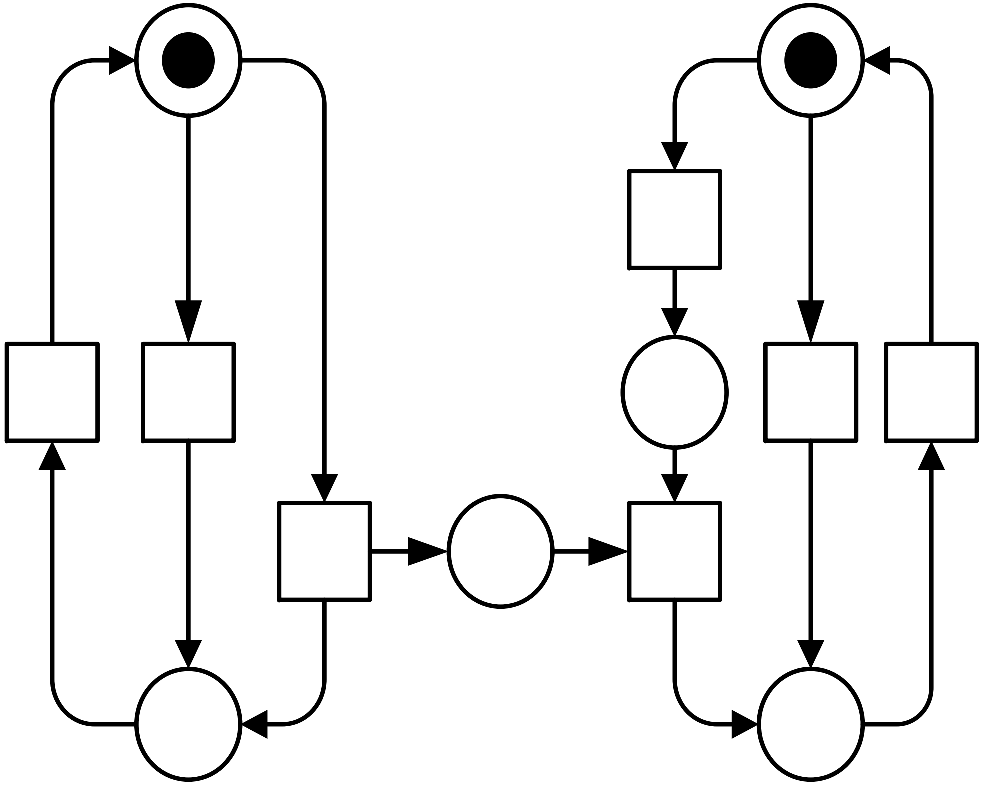
Petri nets model the control flow of concurrent programs which are executed sequentially (e.g. on one circuit)

The relative speed of the components can be controlled.
Buffers are modelled by weakly bounded places.

Choices depend on (unknown) data.
So choices can **not** be controlled



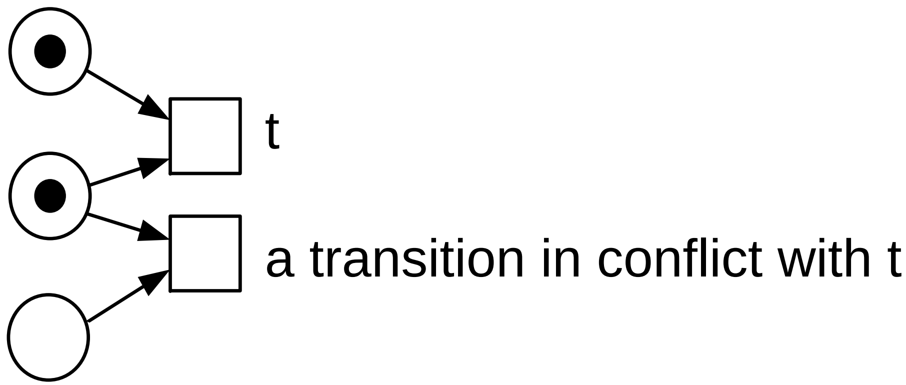
weakly bounded



weakly bounded ???

# Progress assumption:

If  t is enabled then
either t occurs or a transition which is in conflict with t

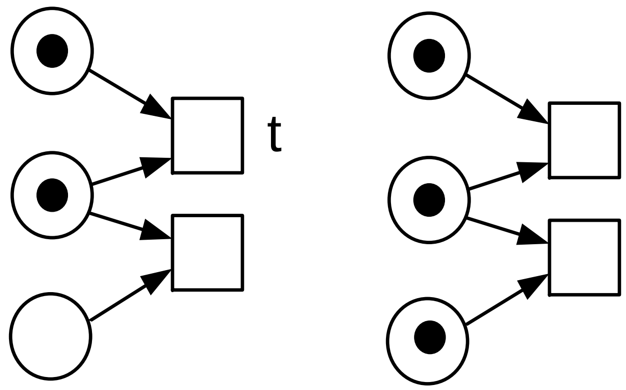



An occurrence sequence is **progressing**
if it satsifies the progress assumption.

## Fairness:

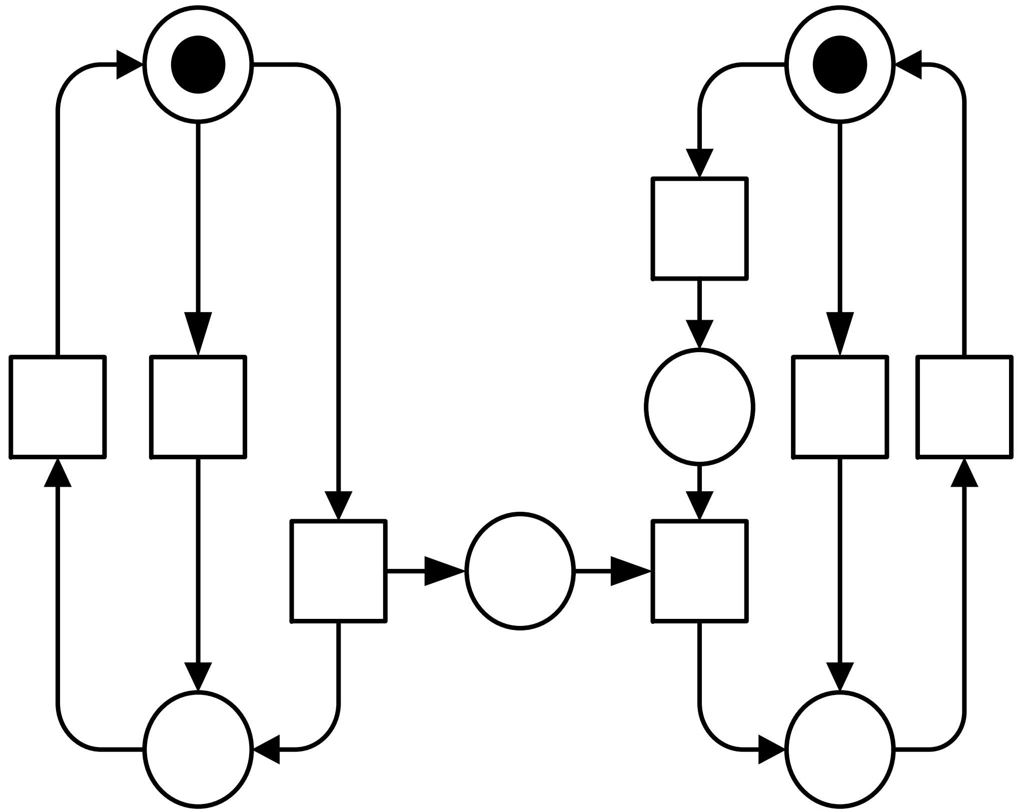
Each possible alternative will be selected eventually
(each loop terminates …)

t occurs
eventually




both transitions
occur eventually

An occurrence sequence is called **fair**
if it satisfies the fairness assumption



weakly bounded !!!

# Petri nets with branching places

## Definition

A place s is called **weakly k-bounded**,
if each progressing fair occurrence sequence can be permuted
where the order of alternatives (decision of choices) is kept
such that in the resulting occurrence sequence
s carries never more than k tokens

A Petri net is called **weakly k-bounded**
if all ist places are weakly k-bounded

A Petri net is called **weakly bounded**,
if each progressing fair occurrence sequence can be permuted
where the order of alternatives (decision of choices) is kept
such that in the resulting occurrence sequence
only finitely many markings are reached

# A result

restricting assumptions:

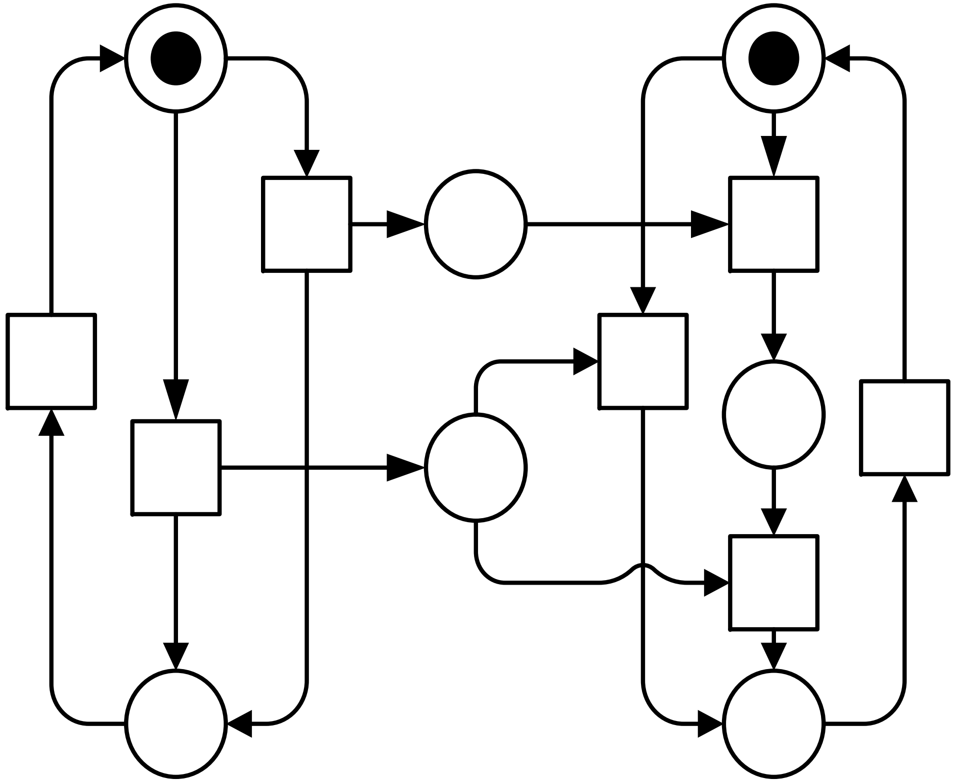
finitely many live state machines
\+ buffer places

a connected Petri net

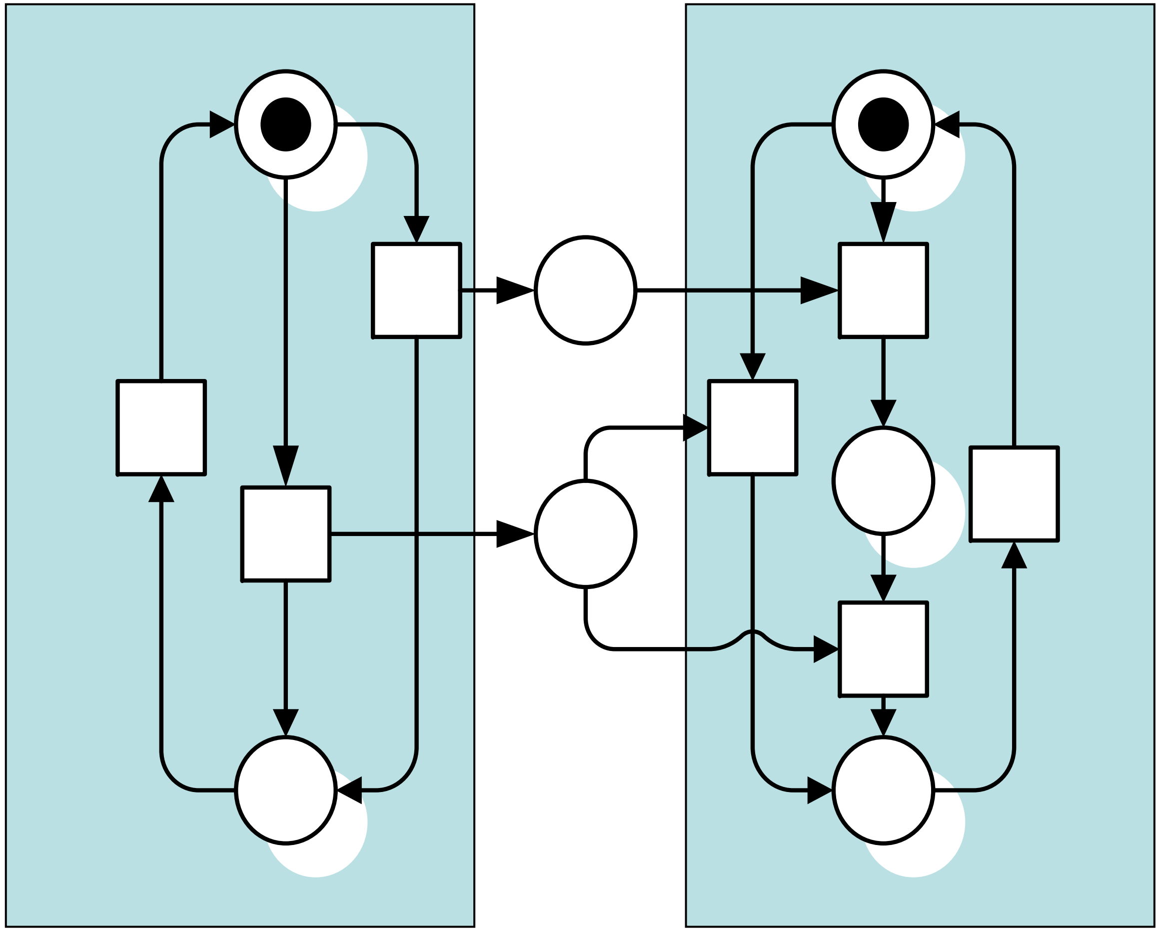
the net (composed state machines + buffers) is live

choices are either free-choice (data dependent, if-then-else)
or controlled by buffer places (select statement)

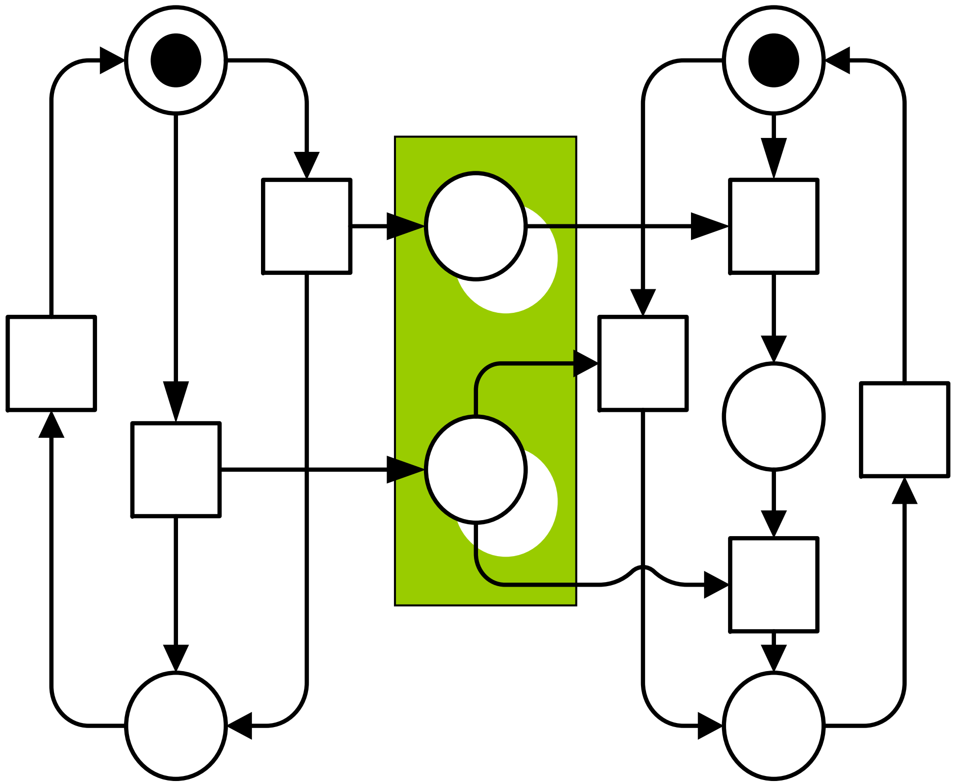
Name: **coupled state machines**



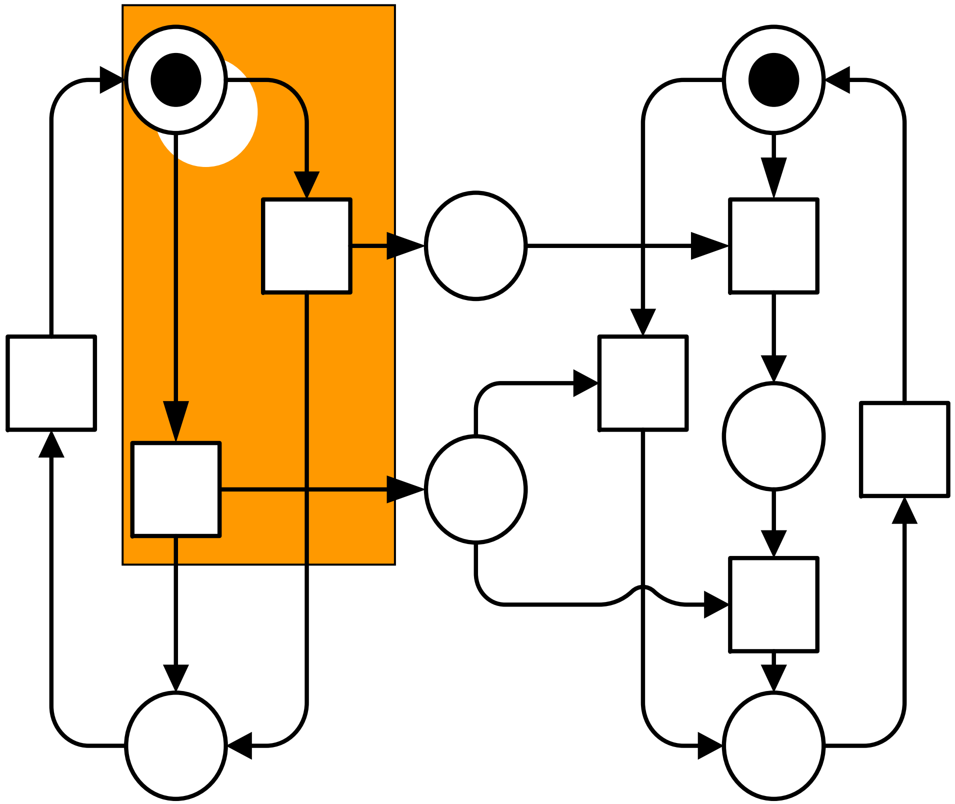
A coupled state machine



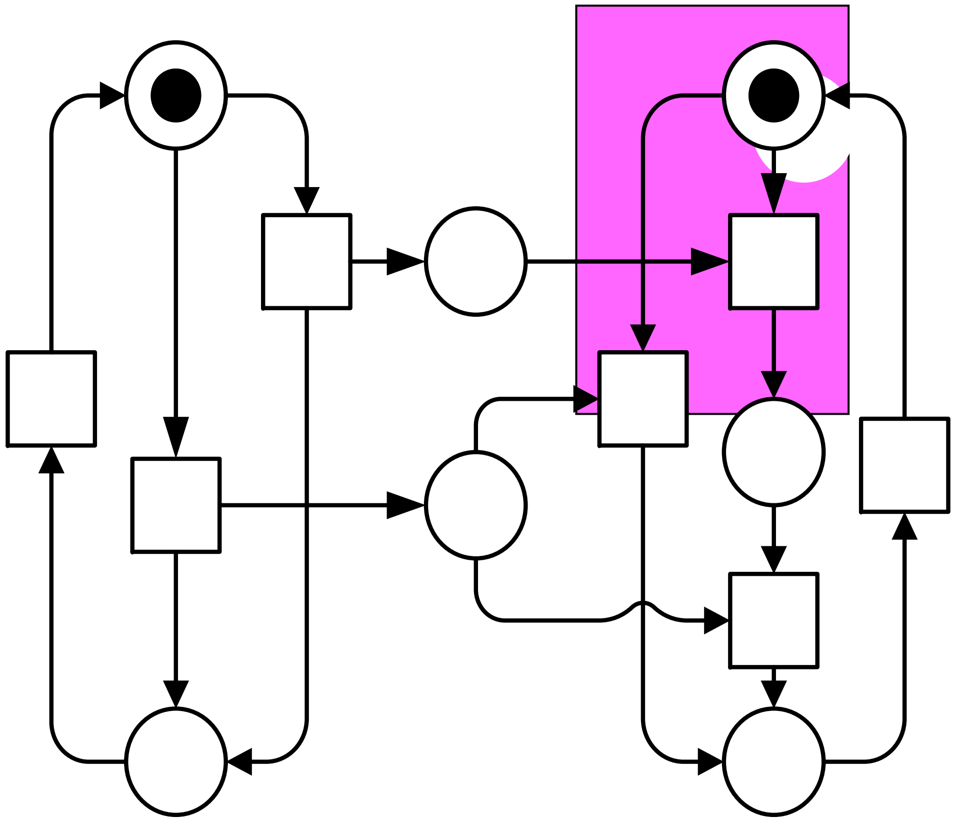
state machines



Buffer places



data dependent choice



buffer controlled choice

## The Result

A coupled state machine is weakly bounded

if and only if

The rank of ist incidence matrix equals $|T| - |A^{\bullet}| + |A| - 1$

where

T – set of transitions

A – set of free-choice alternatives

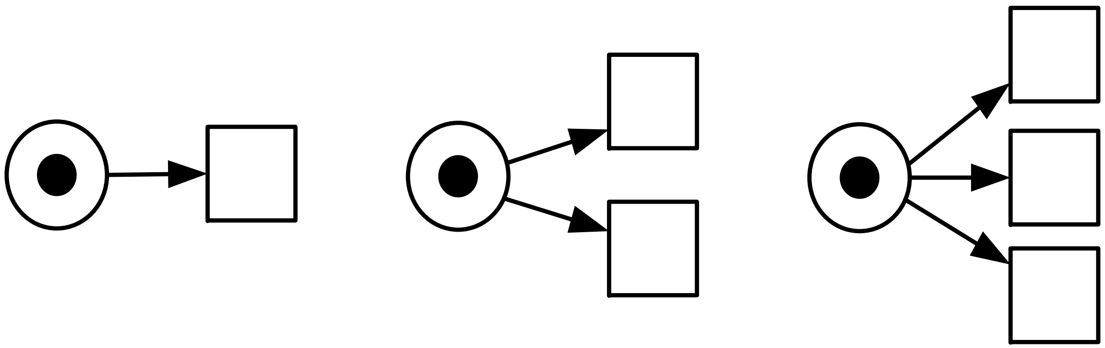
## The result

Another formulation of

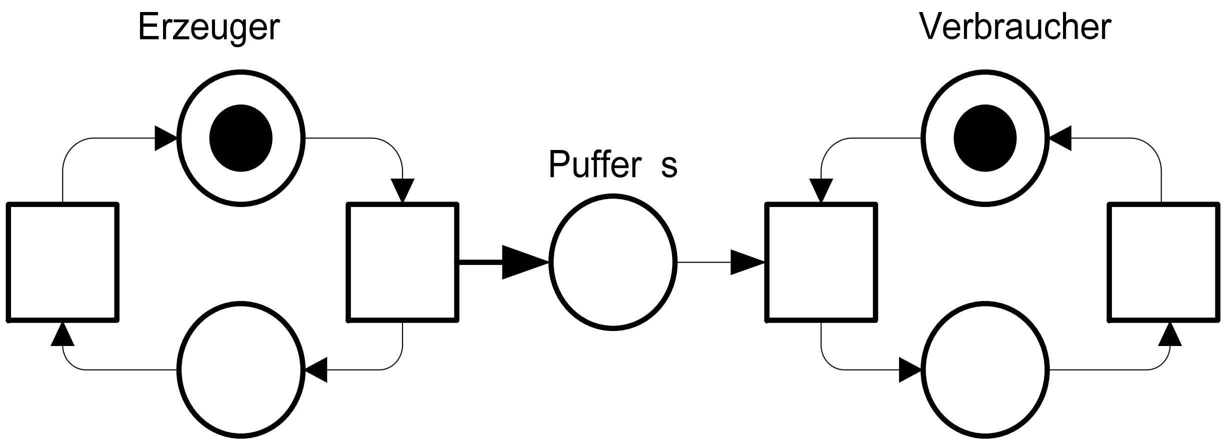
The rank of ist incidence matrix equals $| T | - | A \bullet | + | A | - 1$:

$|$ Linearly independent T-invariants $| =$

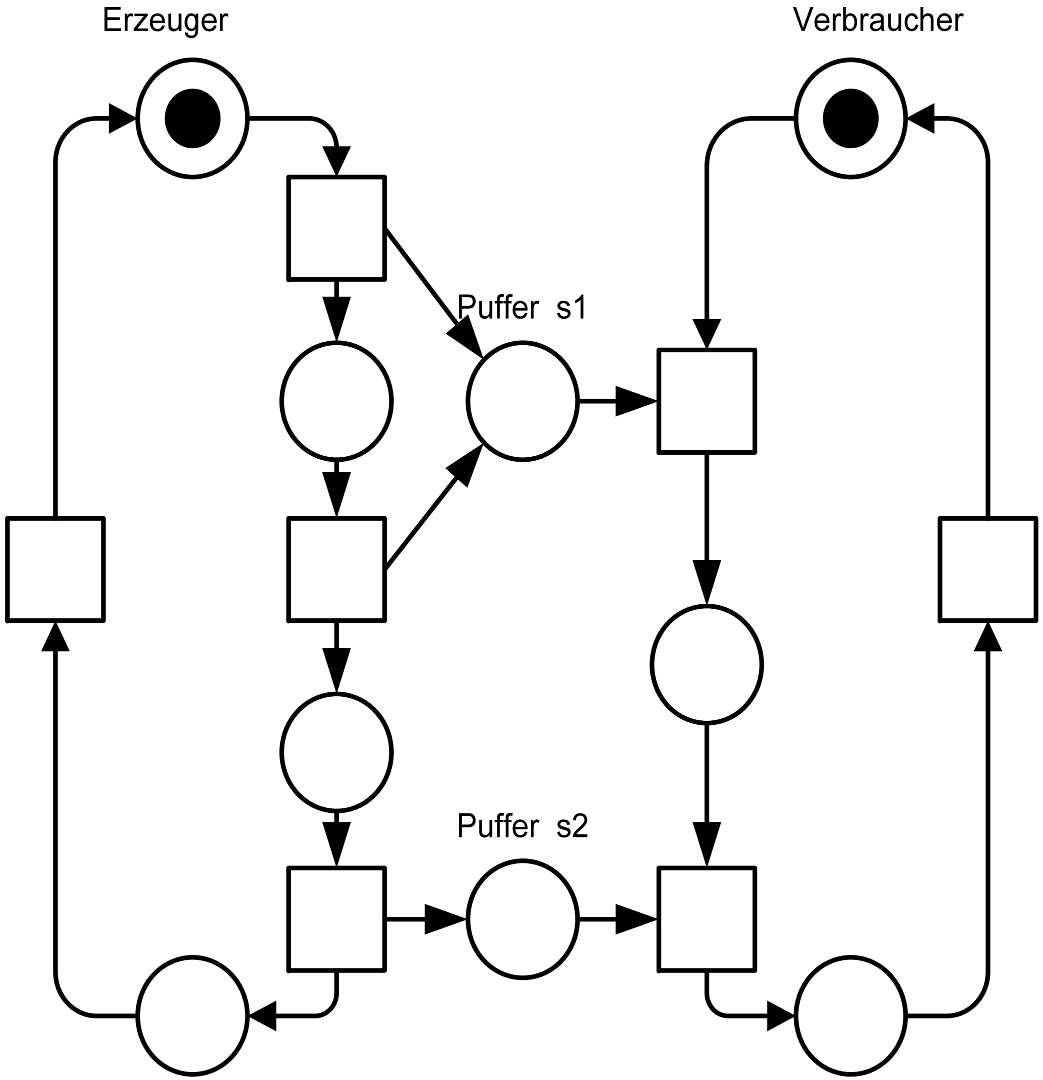
1 + number of free-choice alternatives



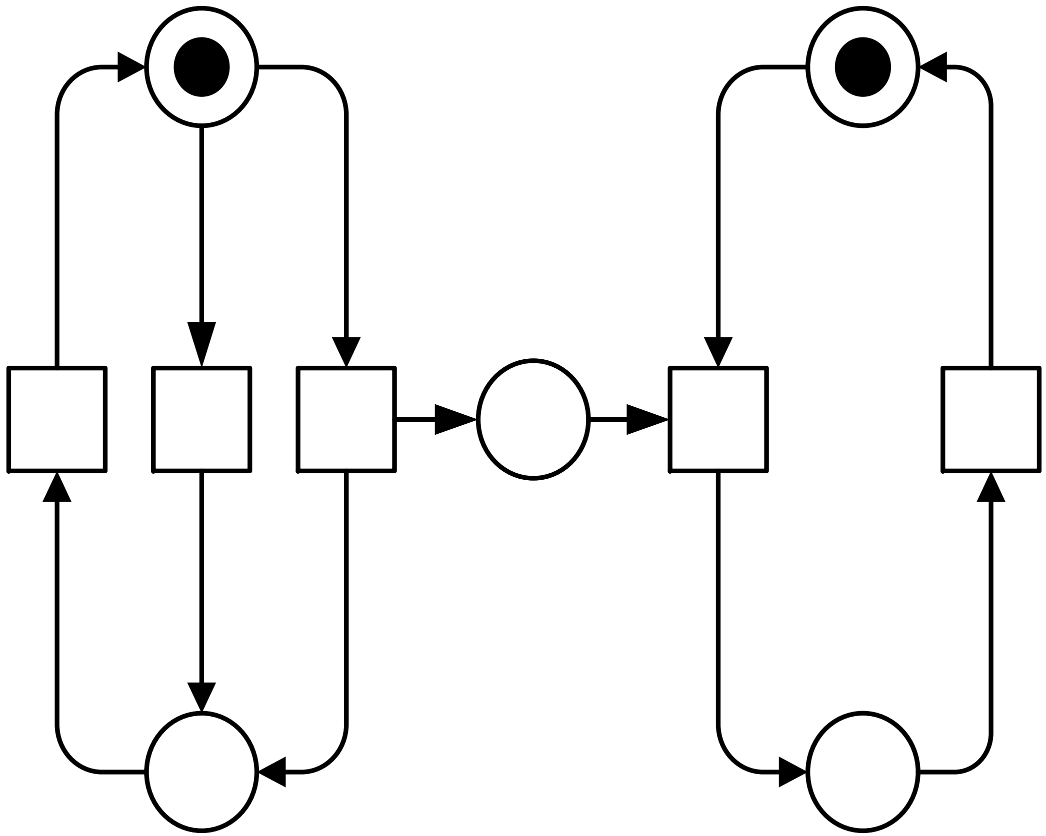
0 Alternativen 1 Alternative 2 Alternativen



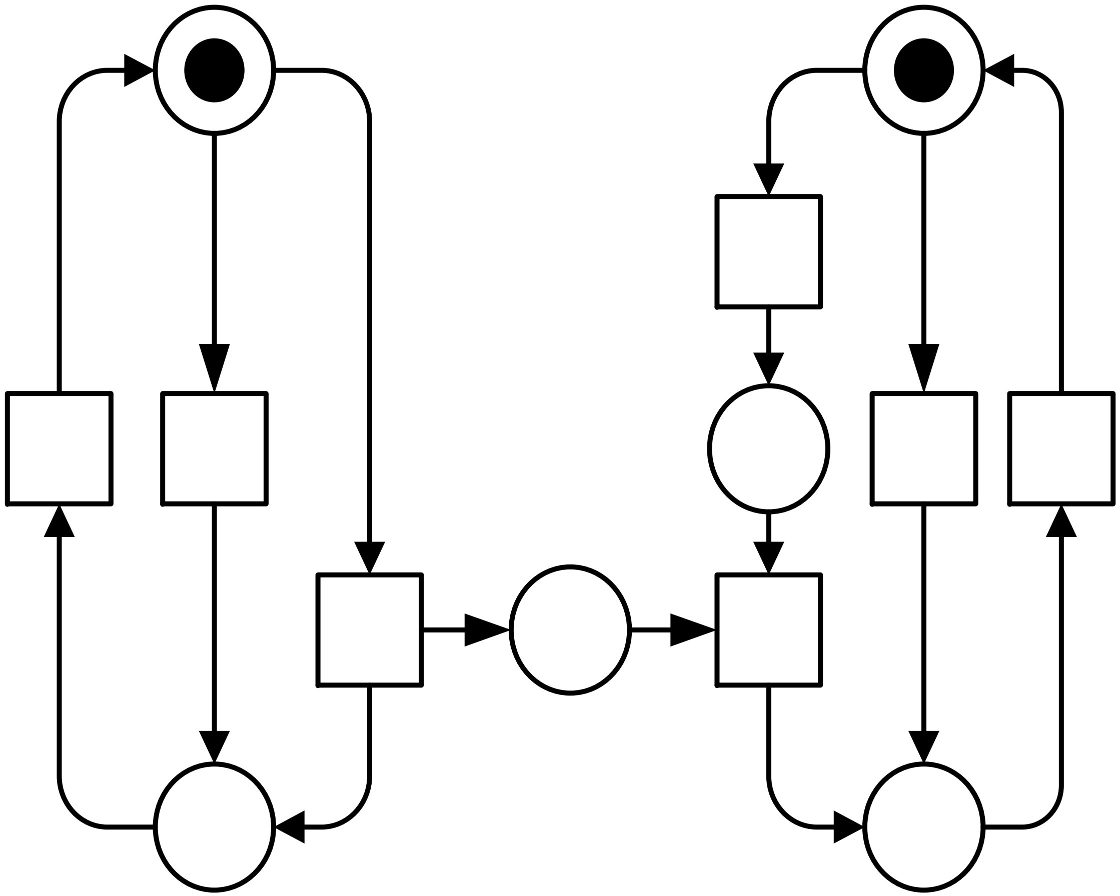

1 T-invariant = 1 + 0 alternativs



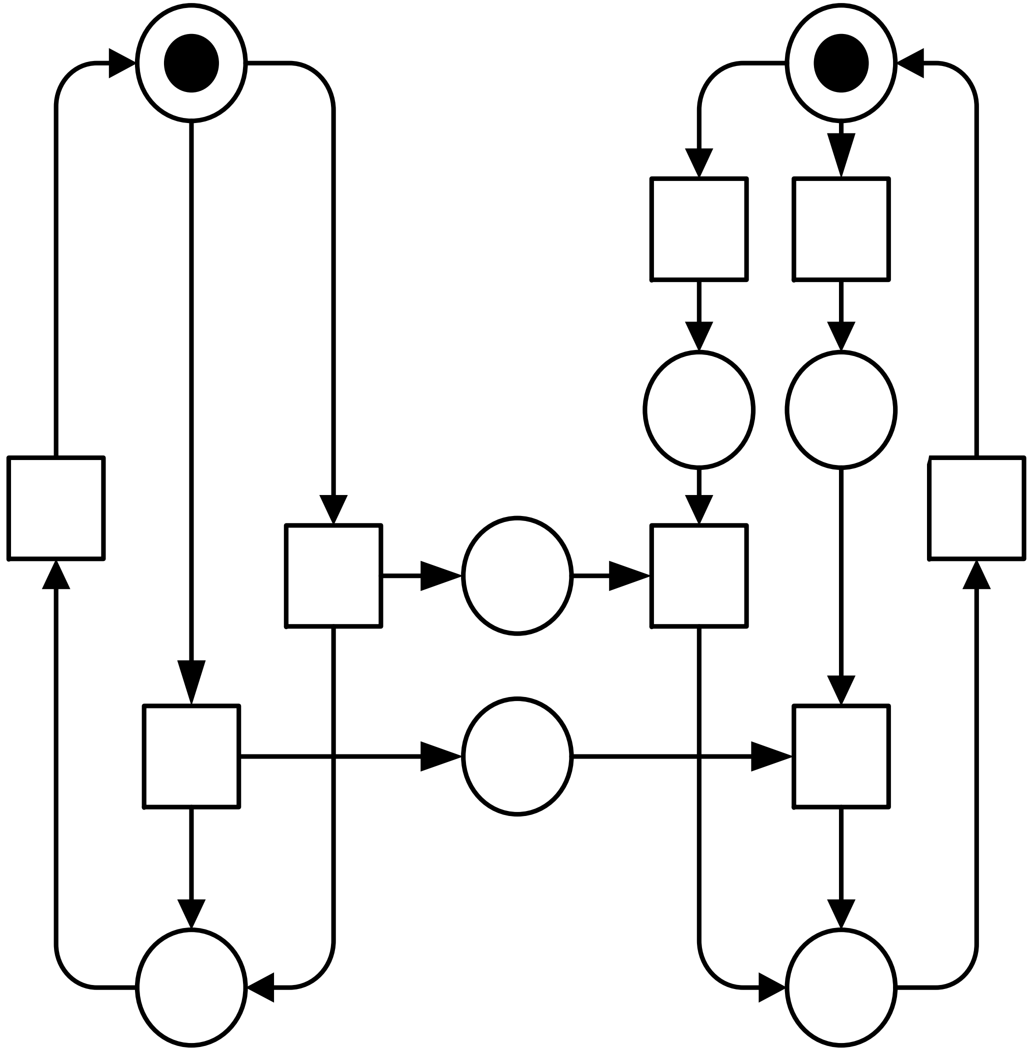

0 T-invariants $\neq$ 1 + 0 alternativs



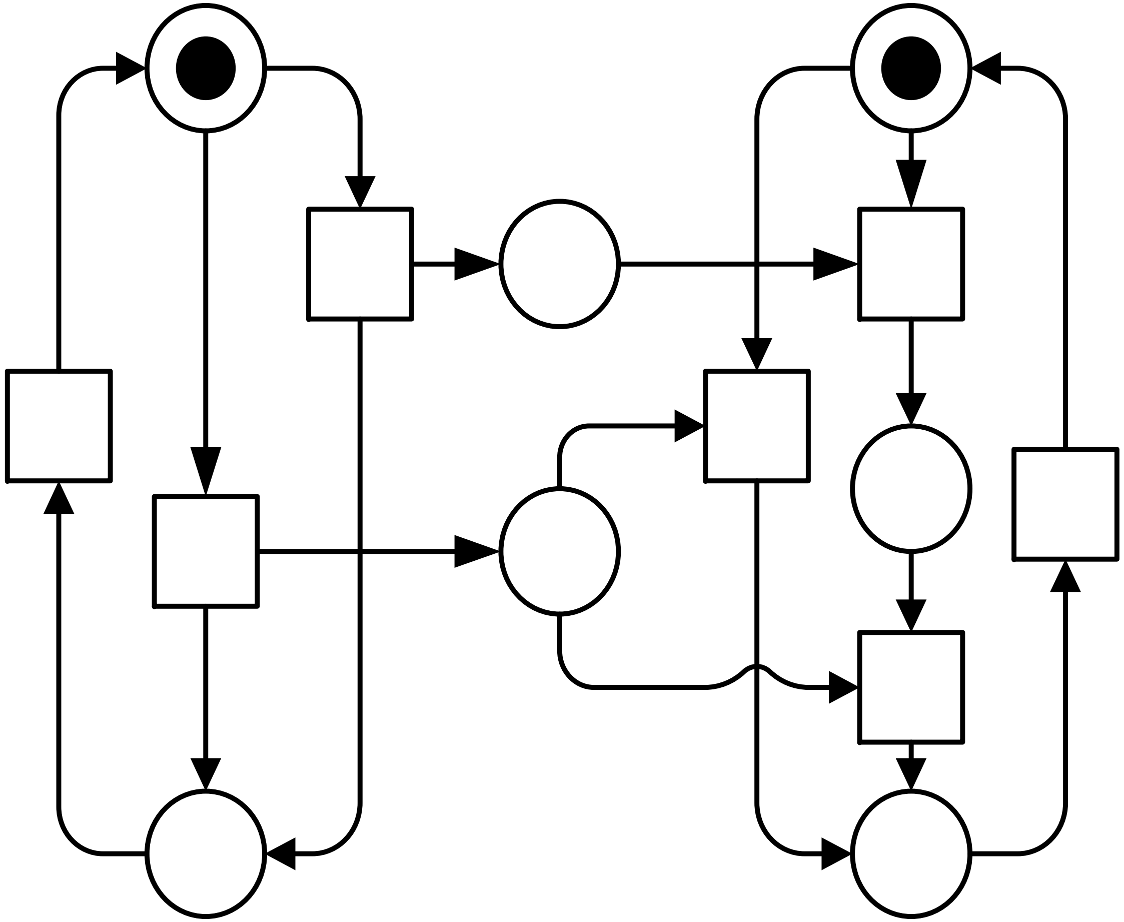
2 T-invariants = 1 + 1 alternativs



3 T-invariants   =  1 + 2 alternativs



2 T-Invariants $\neq$ 1 + 2 alternativs



2 T-invariants   =  1 + 1 alternativs

This Petri net is weakly bounded
but not weakly k-bounded for any k

# Schedulability Analysis of Petri Nets Based on Structural Properties

Cong Liu[1], Jörg Desel[2] ,
Alex Kondratyev[3], Yosinori Watanabe[3],
Alberto Sangiovanni-Vincentelli[1]

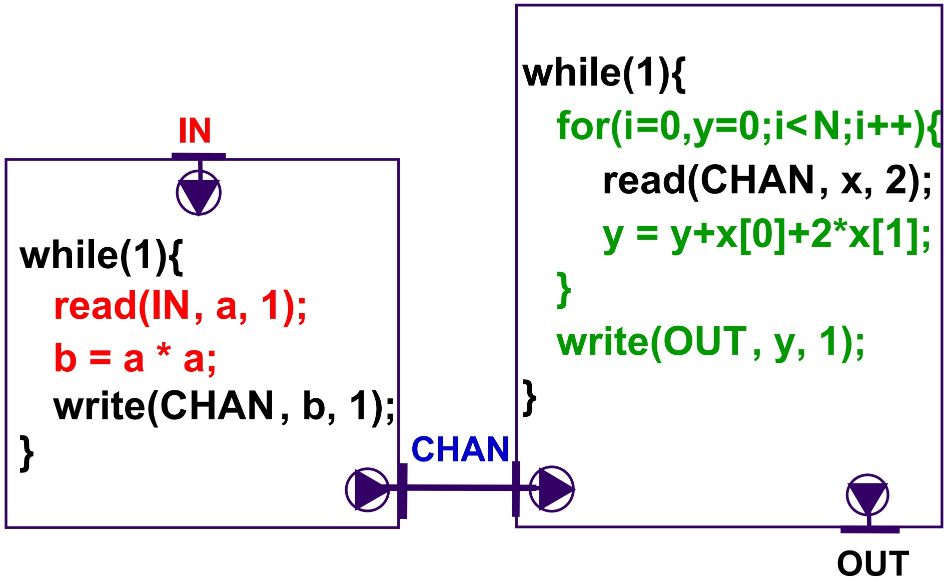
[1]University of California, Berkeley, USA
[2]Katholische Universität Eichstätt-Ingolstadt, Germany
[3]Cadence Berkeley Laboratories, USA

ACSD 2006 / Fundam. Inform. 86(3): 325-341 (2008)

# Scheduling Concurrent Programs

- The problem:

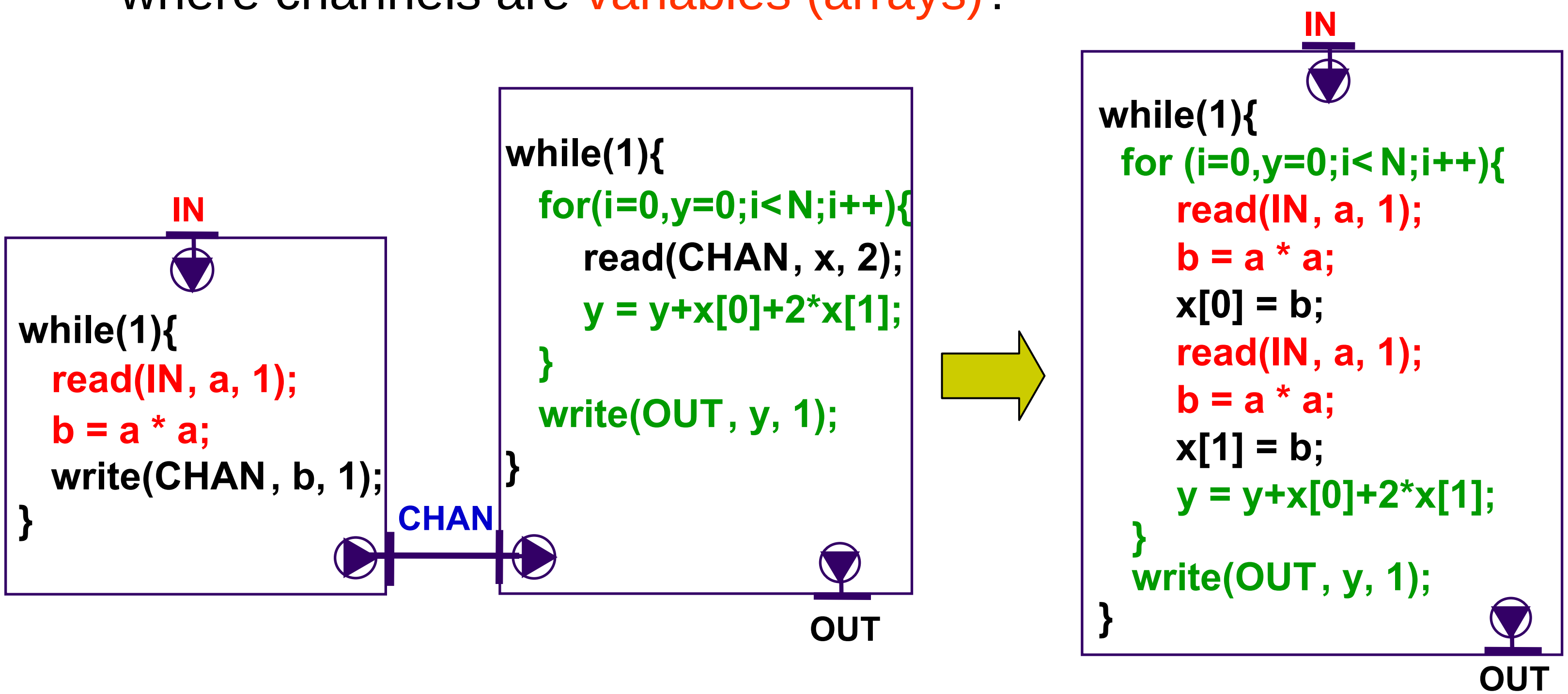
  Given a set of concurrent non-terminating processes communicating through channels with infinite capacity,
  is there a sequential execution where
  channels are bounded?



# Scheduling Concurrent Programs

- The problem:

  Given a set of concurrent non-terminating processes communicating through channels with infinite capacity,
  is there a single process comprising the concurrent processes where channels are variables (arrays)?

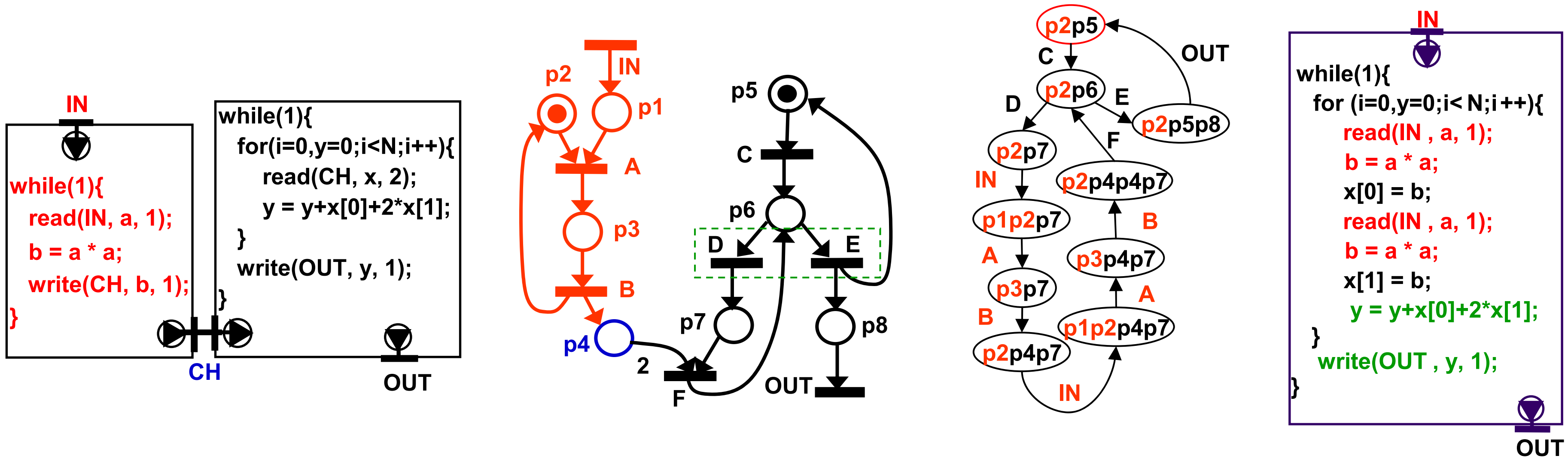


# Scheduling Classification

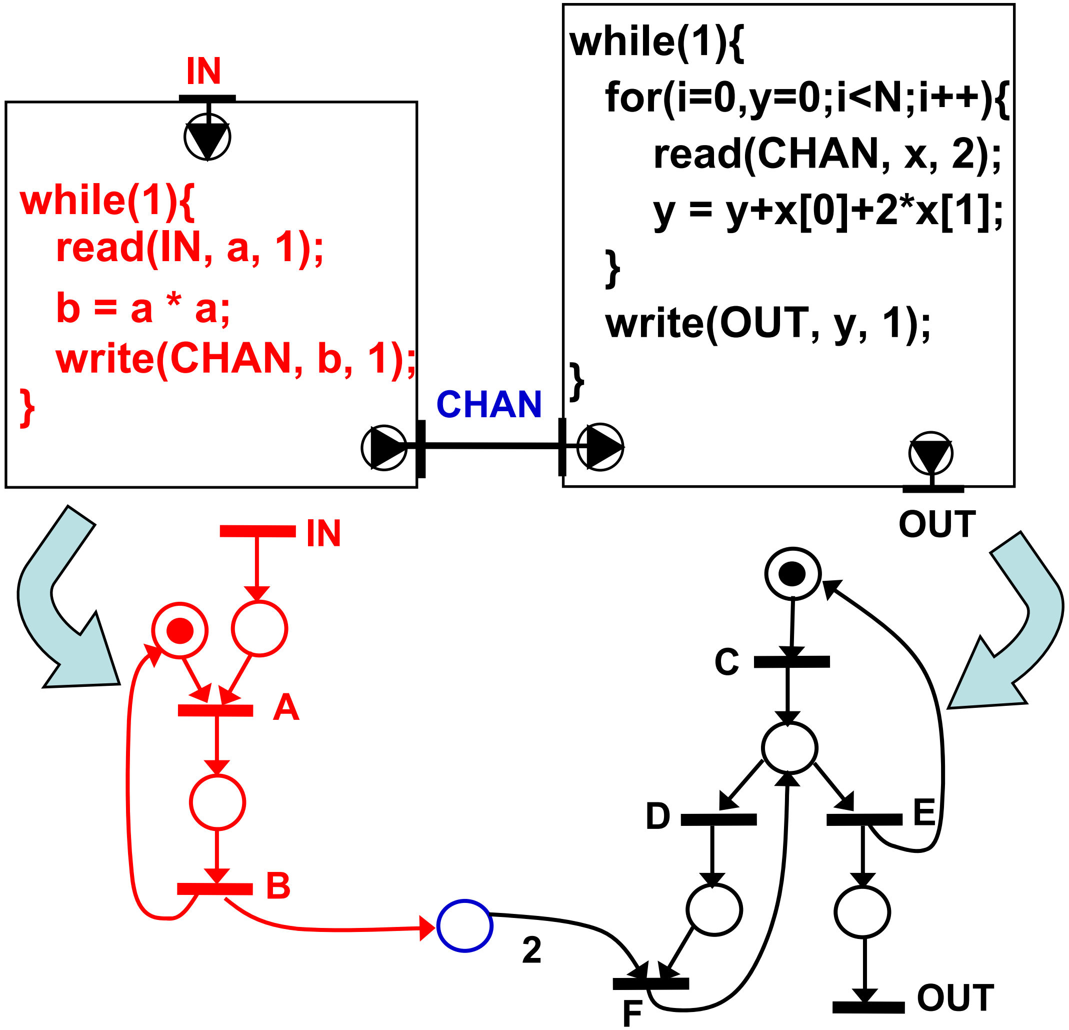
- Dynamic scheduling
  - Make all scheduling decisions at run-time
  - Context switch overhead
- Static scheduling [Lee 87]
  - Make all scheduling decisions at compile-time
  - Reduce context switch overhead
  - Restricted to specification without data-dependent controls (e.g. if-then-else)
- Quasi-static scheduling
  - Allow specification to have data-dependent controls
  - Perform static scheduling as much as possible
  - Leave data-dependent choices to be resolved at run-time

# Quasi-Static Scheduling [Cortadella et al 00]

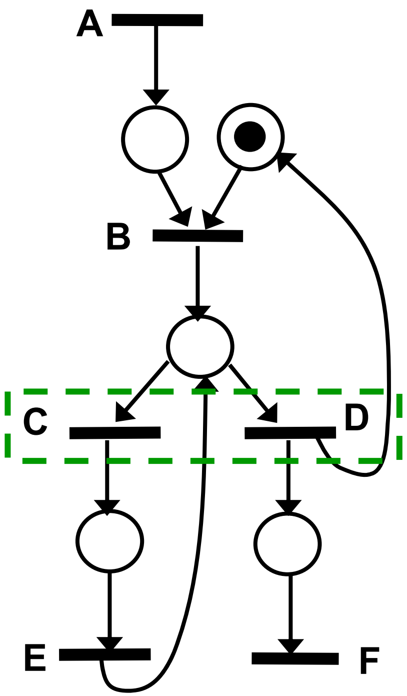
- Translate concurrent programs to a Petri net
- Find a quasi-static schedule for the Petri net
- Generate a sequential program from the schedule



# Concurrent Programs → Petri Net



# Petri Nets and Free Choice Sets



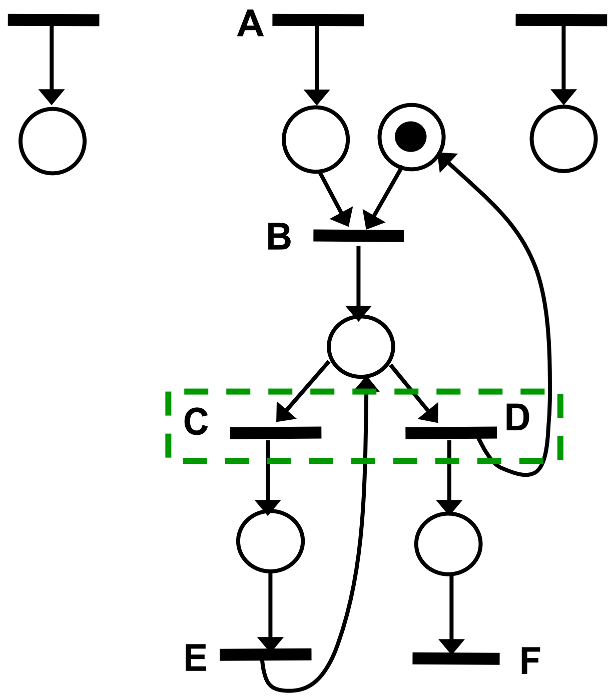

{C, D} is called a

**Free Choice Set (FCS).**

It represents a data-dependant branch (if-then-else, loop)

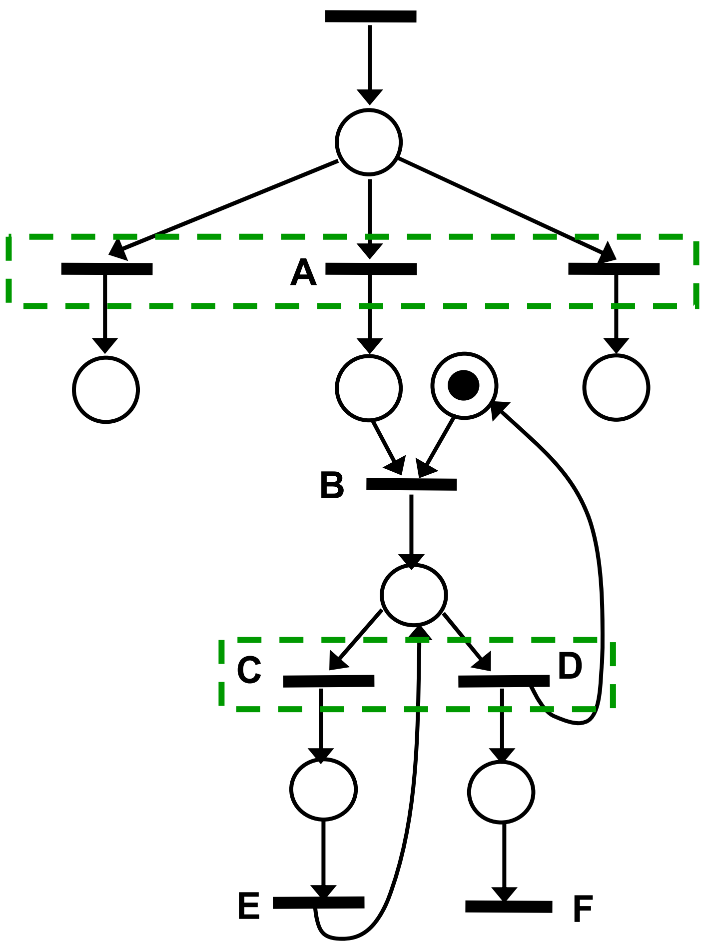
We assume that each Free Choice Set has exactly two elements

# Petri Nets and Free Choice Sets



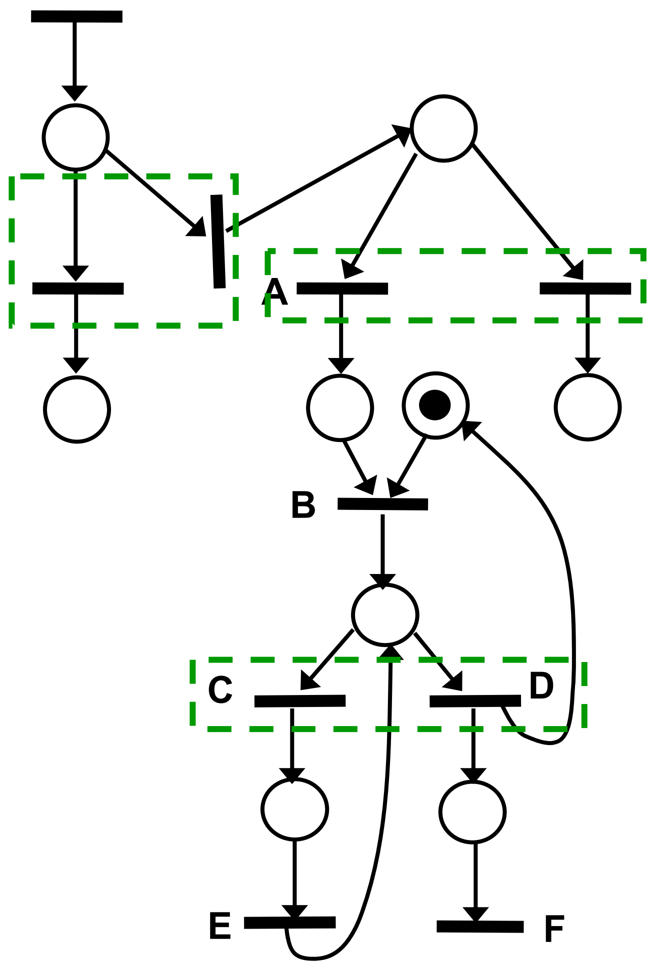

Several input transitions ?
(transitions with empty pre-set)

# Petri Nets and Free Choice Sets




**All input transitions (transitions with empty pre-set) generate a single Free Choice Set**

# Petri Nets and Free Choice Sets

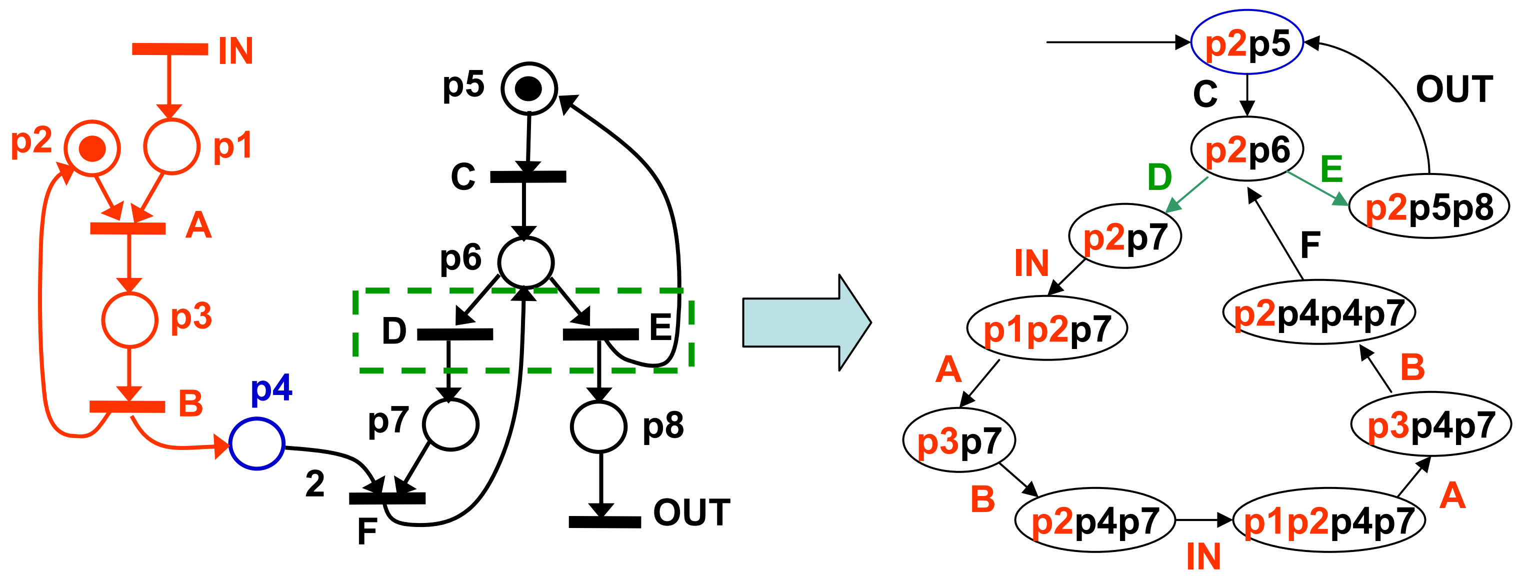



All input transitions
(transitions with empty pre-set)
generate a single Free Choice Set

We assume that each
Free Choice Set has exactly
two elements

# Schedule of a Petri net

- finite directed graph with a “root”
- **Vertices:** mapped to markings, root to initial marking
- **Edges:** transition occurrences, changing the marking
- Branching vertex: corresponds to a Free Choice Set
- strongly connected



# Schedulability

A Petri net is schedulable if it has a schedule

Question: Is a given Petri net schedulable?
Is a given Petri net not schedulable?

Solution 1: Try to construct a schedule
very time consuming

Solution 2: Employ necessary conditions for schedulability
which are based on the Petri net structure
and hence efficient to decide.

» Checking Cyclic Dependence of Transitions using Linear Programming

» Checking a Rank Condition using Linear Algebra

# Experiments

- Codecs
  - PVRG-JPEG encoder [Hung 93]
  - Motion-JPEG encoder [Lieverse 01]
  - Philips MPEG2 decoder [Wolf 99]
  - XviD MPEG4 encoder [Broekhof 04]

| | #P | #T | #Arc | #FCS | Rank | CDC | Scheduler |
|---|---|---|---|---|---|---|---|
| **JPEGenc1** | 26 | 27 | 64 | 6 | <0.01s | 0.19s | >24hr |
| **MJPEGenc** | 117 | 124 | 330 | 25 | <0.01s | 0.04s | >24hr |
| **MPEG2dec1** | 116 | 144 | 358 | 38 | <0.01s | 0.25s | >24hr |
| **MPEG4dec** | 72 | 72 | 184 | 15 | <0.01s | 0.16s | >24hr |

# Related work

Weakly bounded Message Sequence Charts

(Anca Muscholl, Blaise Genest, Dietrich Kuske)

# Open Questions

Decidability for the general case
(idea: exclude loops in a coverability graph

Algorithms

Further interpretations ($\rightarrow$ Monika Heiner)

Precise relation to weakly bounded MSCs

…