LECTURE 5

PETRI NETS

***THESE SLIDES ARE BASED ON LECTURE NOTES FROM:
DR. CHRIS LING (HTTP://WWW.CSSE.MONASH.EDU.AU/~SLING/)***

TOPICS

Petri nets

Examples:

- POS Terminal
- Vending Machine
- Restaurant
- Producer Consumer

Petri net structures

Petri net properties:

- Liveness
- Boundedness
- Reachability

OK, LET'S START...

INTRODUCTION

First introduced by Carl Adam Petri in 1962.

A diagrammatic tool to model concurrency and synchronization in systems

- They allow us to quickly simulate complex concurrent behavior (which is faster than prototyping!)

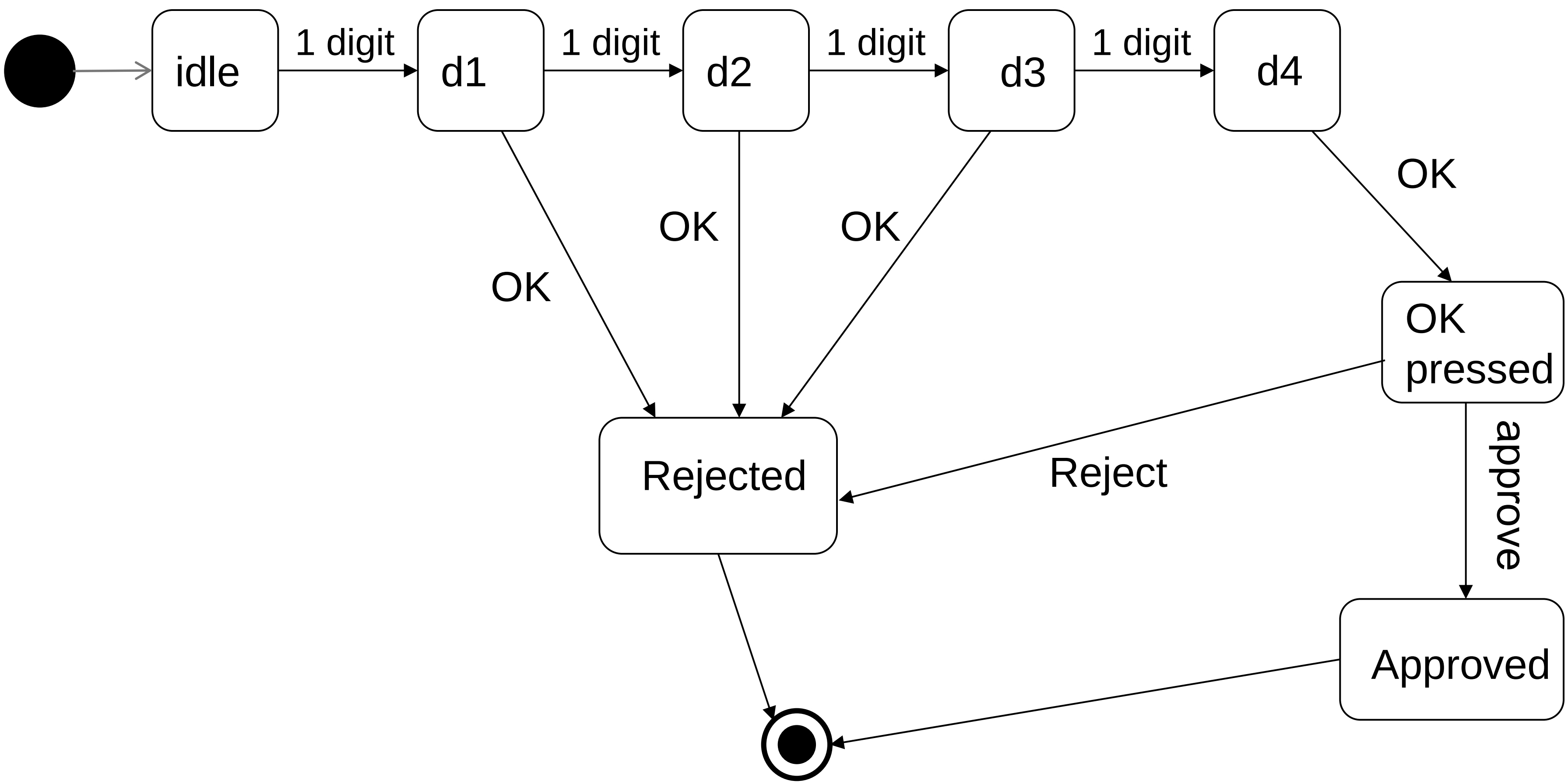
Fairly similar to UML State machines that we have seen so far

- Used as a visual communication aid to model the system behavior

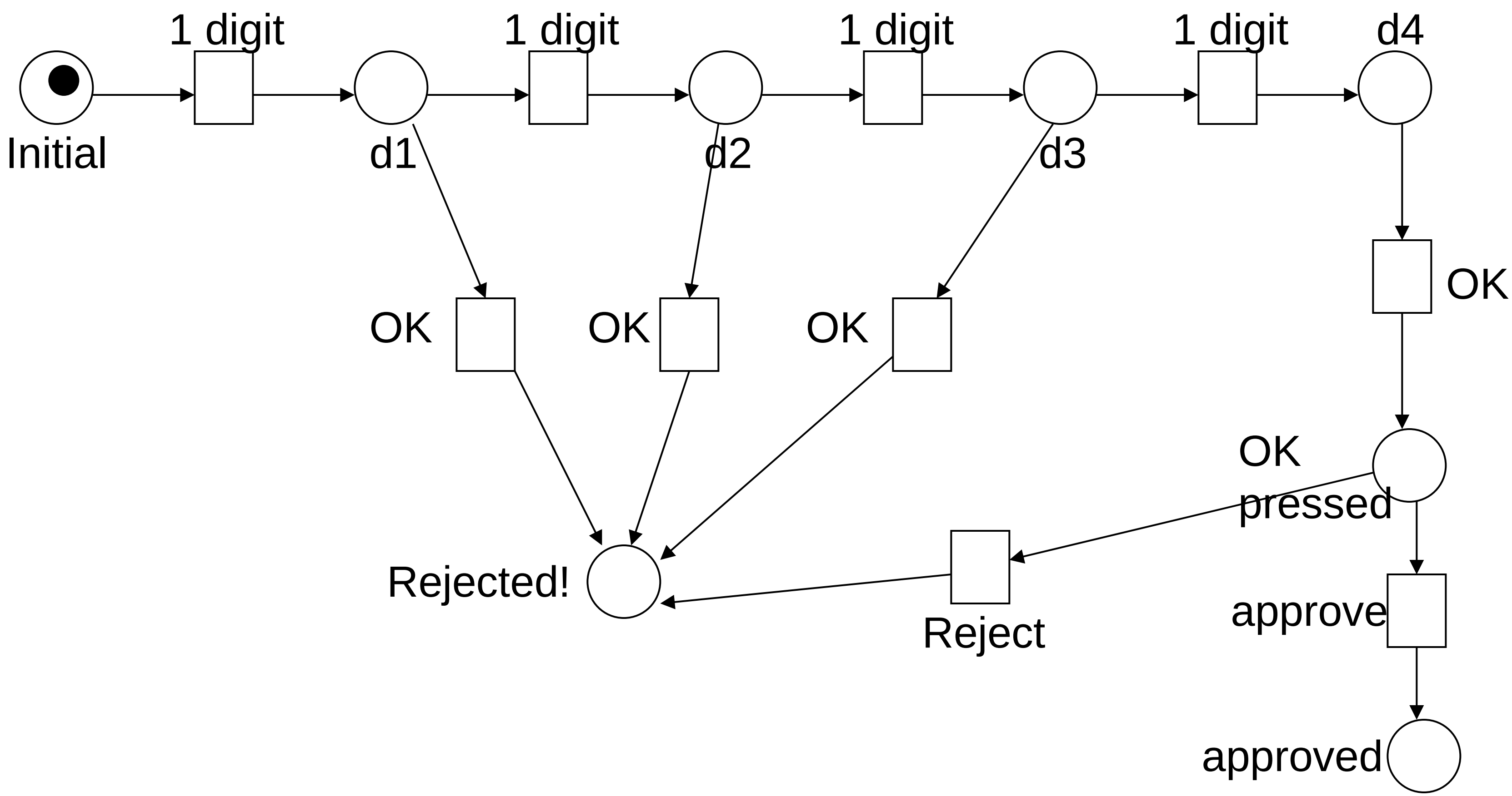
Based on strong mathematical foundation

EXAMPLE: POS TERMINAL (UML STATE MACHINE)

(POS= Point of Sale)



EXAMPLE: POS TERMINAL (PETRI NET)



POS TERMINAL

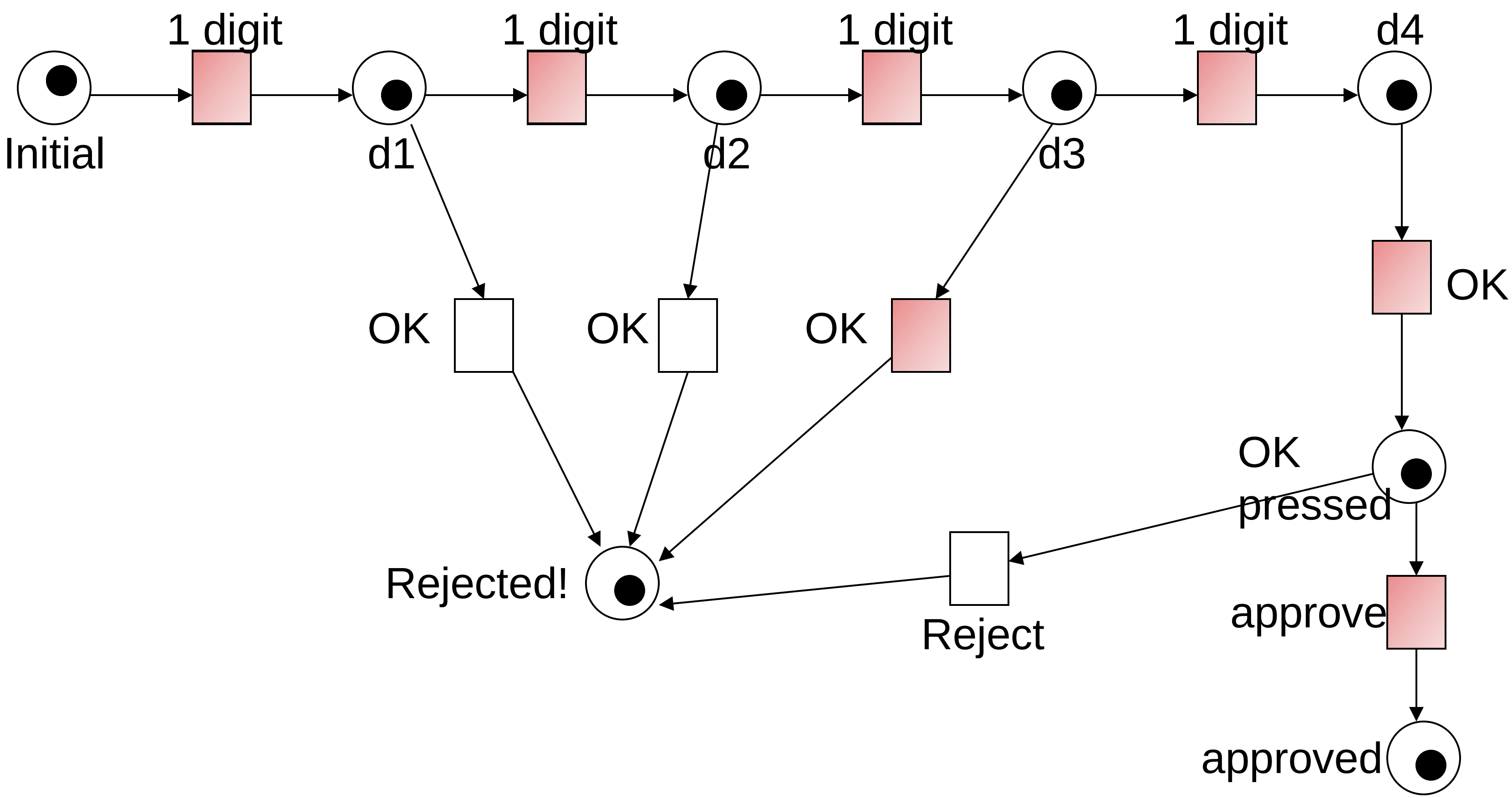
Scenario 1: Normal

- Enters all 4 digits and press OK.

Scenario 2: Exceptional

- Enters only 3 digits and press OK.

EXAMPLE: POS SYSTEM (TOKEN GAMES)



A PETRI NET COMPONENTS

uOttawa
L'Université canadienne
Canada's university

The terms are bit different than UML state machines

Petri nets consist of three types of components: *places* (circles), *transitions* (rectangles) and *arcs* (arrows):

- Places represent possible states of the system
- Transitions are events or actions which cause the change of state (be careful, transitions are no longer arrows here)
- Every arc simply connects a place with a transition or a transition with a place.

CHANGE OF STATE

A change of state is denoted by a movement of *token(s)* (black dots) from place(s) to place(s)

- Is caused by the *firing* of a transition.

The firing represents an occurrence of the event or an action taken

The firing is subject to the input conditions, denoted by token availability

CHANGE OF STATE

A transition is *firable* or *enabled* when there are sufficient tokens in its input places.

After firing, tokens will be transferred from the input places (old state) to the output places, denoting the new state

EXAMPLE: VENDING MACHINE

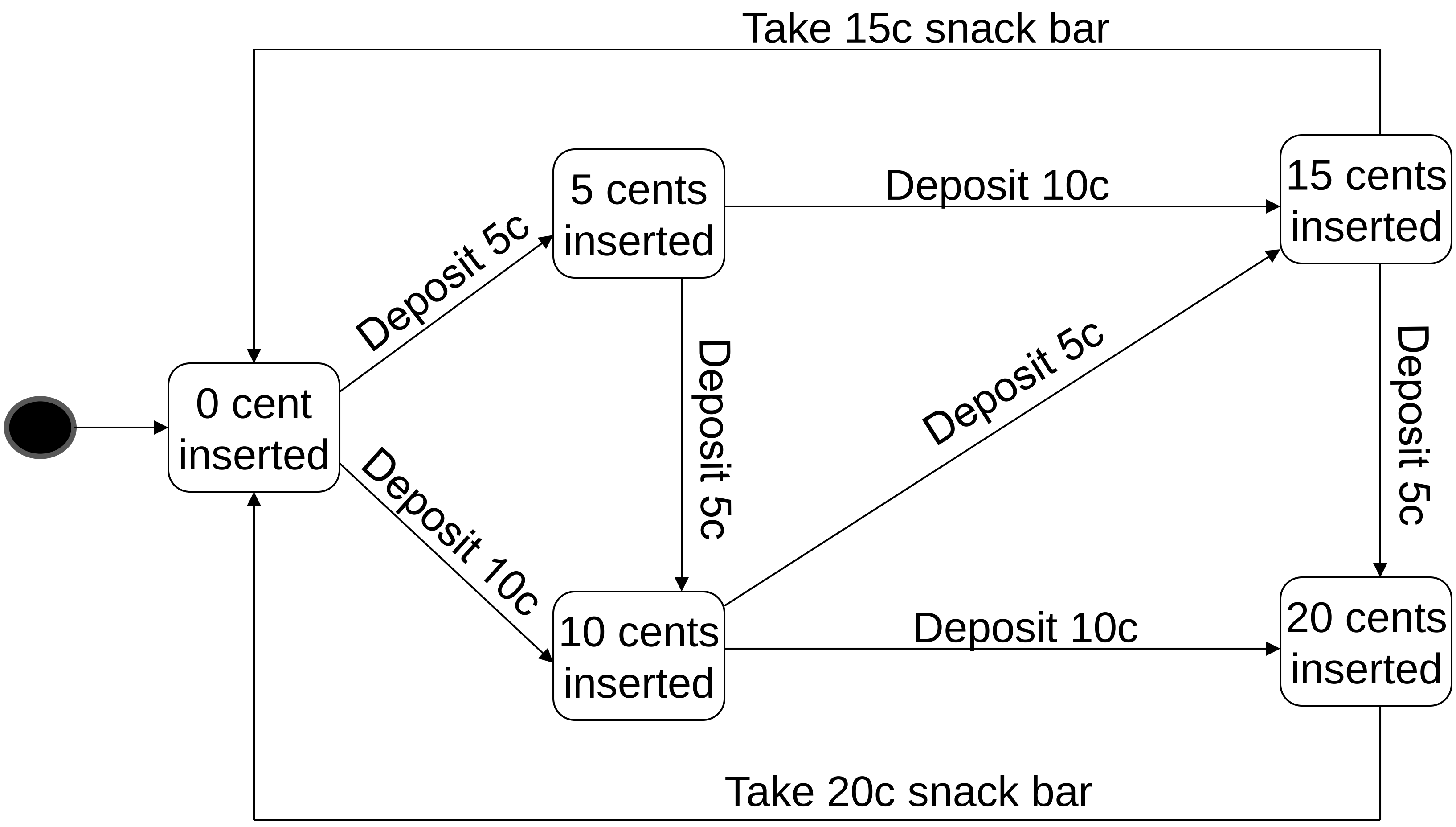
The machine dispenses two kinds of snack bars – 20c and 15c

Only two types of coins can be used

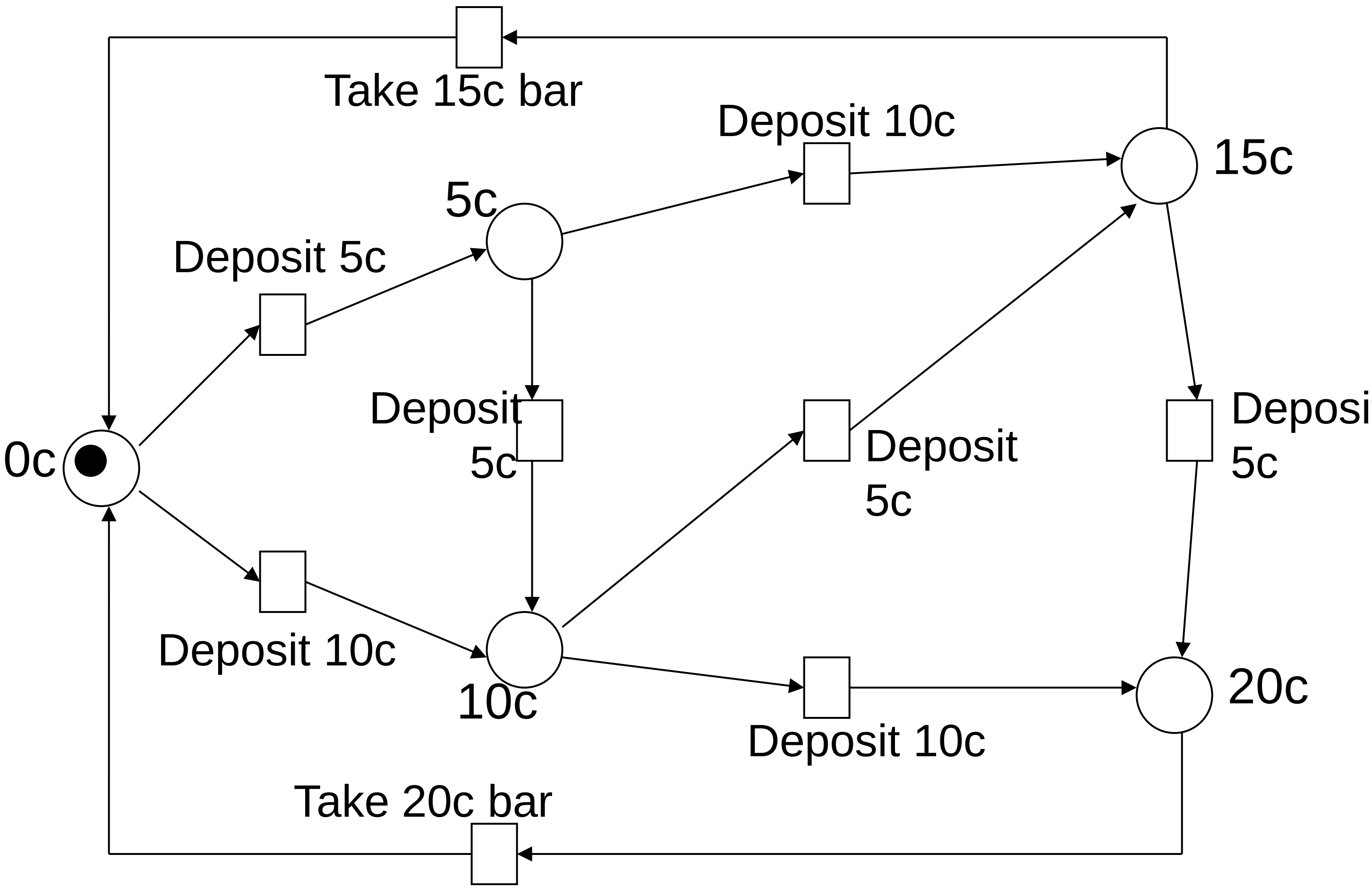
- 10c coins and 5c coins (ah the old days!!)

The machine does not return any change

EXAMPLE: VENDING MACHINE (UML STATE MACHINE)



EXAMPLE: VENDING MACHINE (A PETRI NET)



EXAMPLE: VENDING MACHINE (3 SCENARIOS)

Scenario 1:

- Deposit 5c, deposit 5c, deposit 5c, deposit 5c, take 20c snack bar.

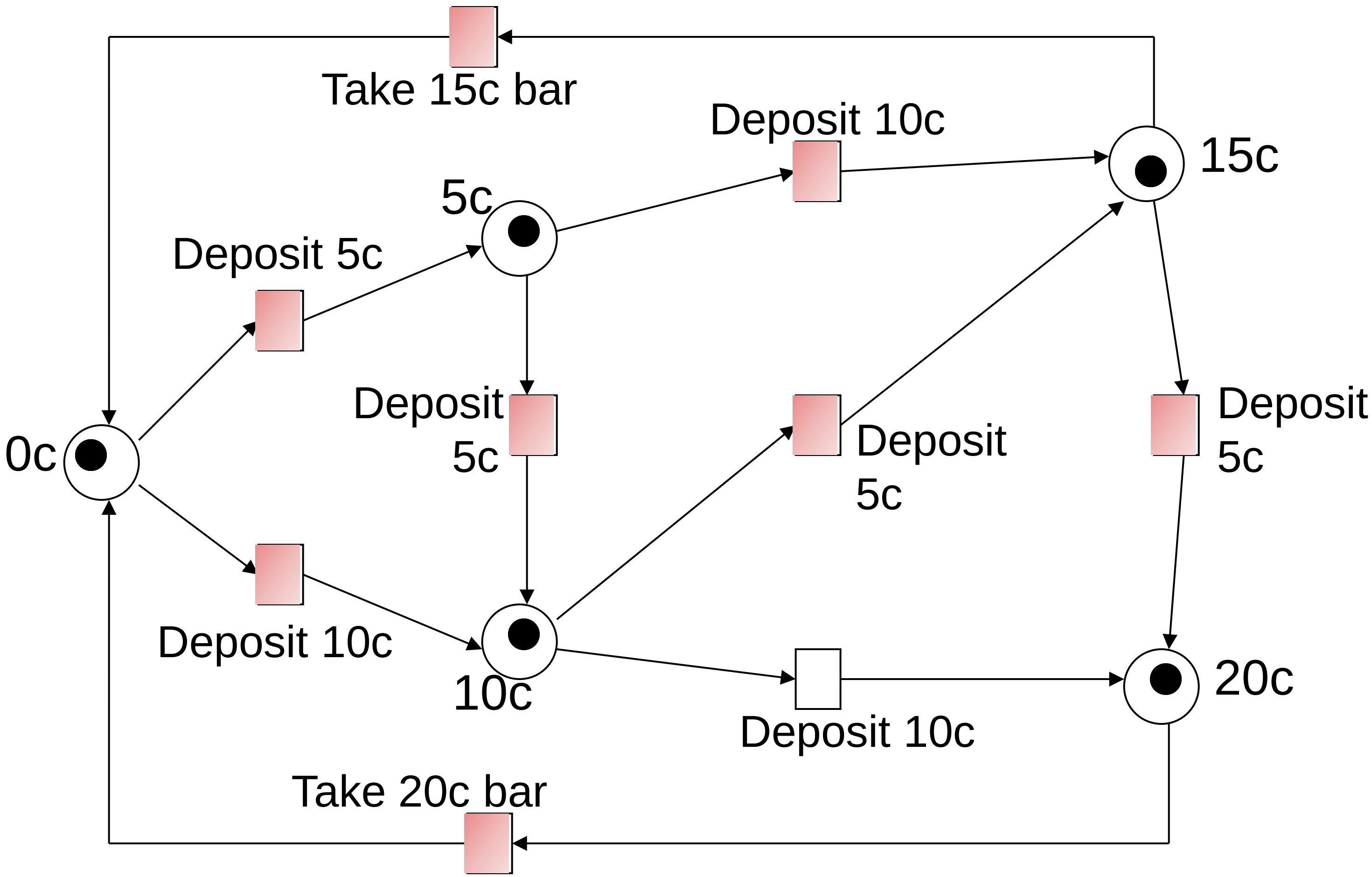
Scenario 2:

- Deposit 10c, deposit 5c, take 15c snack bar.

Scenario 3:

- Deposit 5c, deposit 10c, deposit 5c, take 20c snack bar.

EXAMPLE: VENDING MACHINE (TOKEN GAMES)



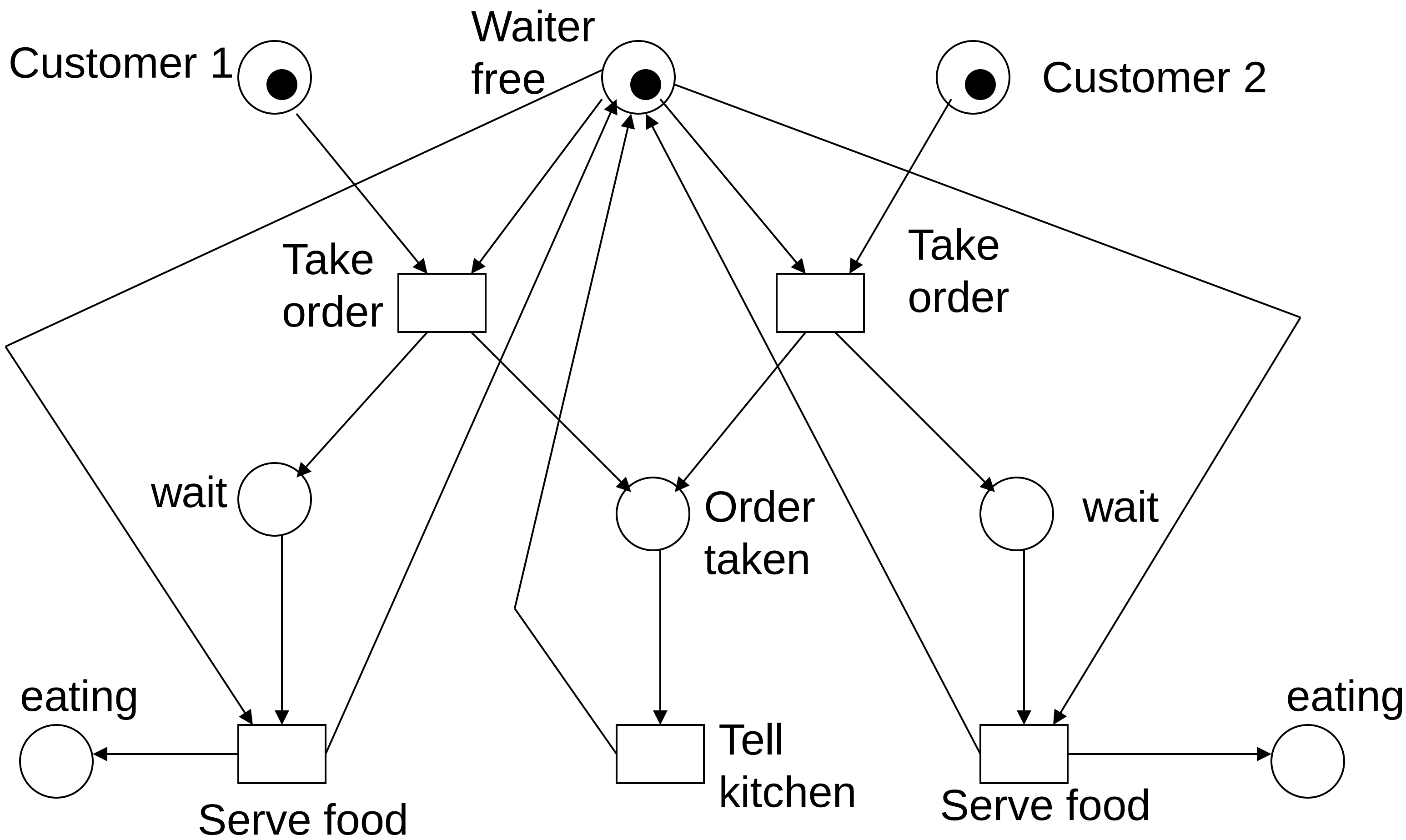
MULTIPLE LOCAL STATES

In the real world, events happen at the same time

A system may have many local states to form a global state.

There is a need to model concurrency and synchronization

EXAMPLE: IN A RESTAURANT (A PETRI NET)



EXAMPLE: IN A RESTAURANT (TWO SCENARIOS)

uOttawa
L'Université canadienne
Canada's university

Scenario 1:

Waiter

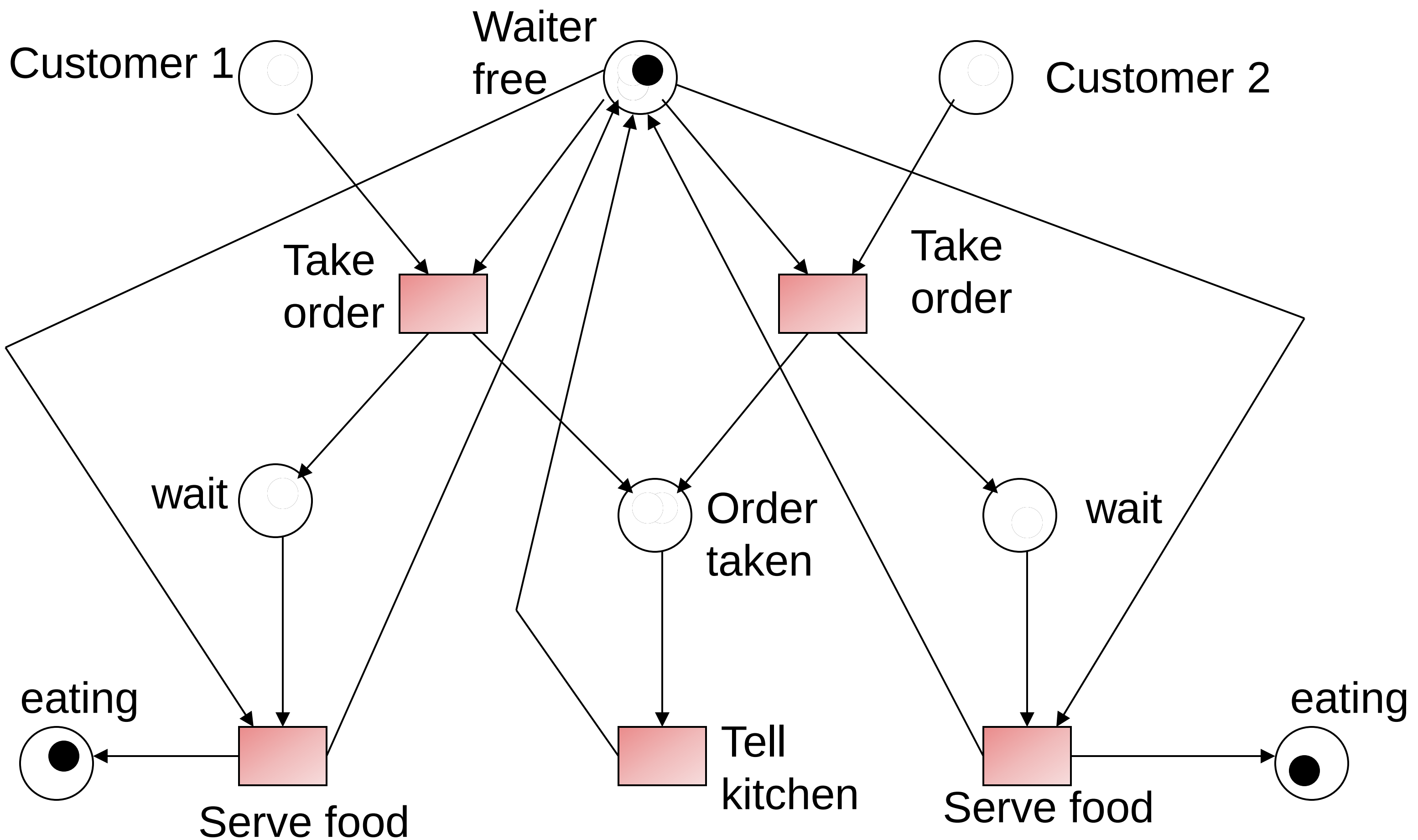
1. Takes order from customer 1
2. Serves customer 1
3. Takes order from customer 2
4. Serves customer 2

Scenario 2:

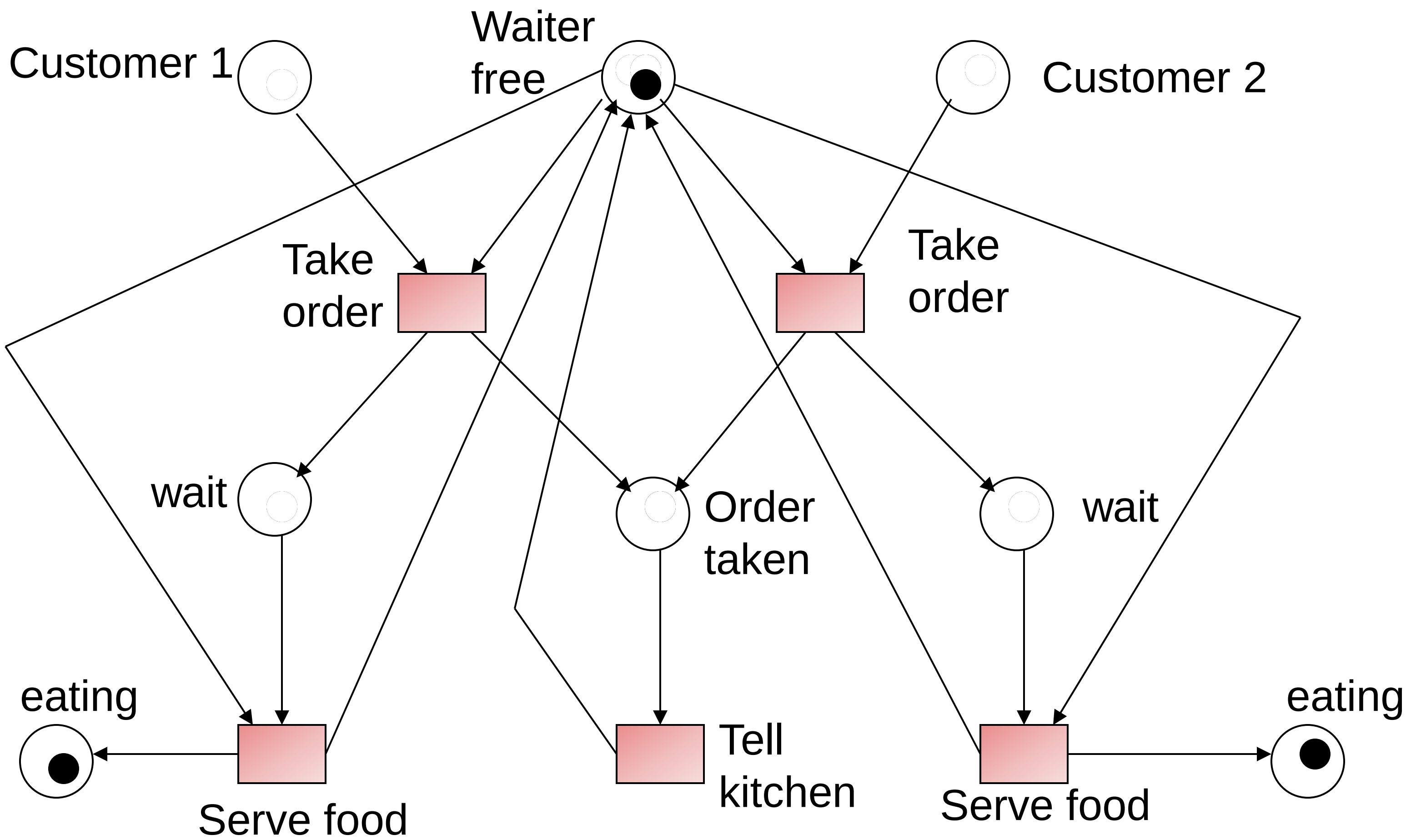
Waiter

1. Takes order from customer 1
2. Takes order from customer 2
3. Serves customer 2
4. Serves customer 1

EXAMPLE: IN A RESTAURANT (SCENARIO 2)

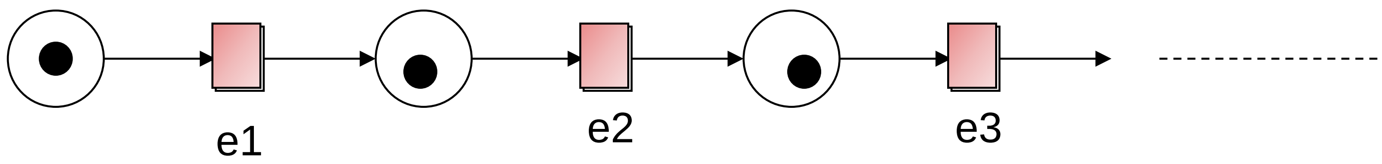


EXAMPLE: IN A RESTAURANT (SCENARIO 1)

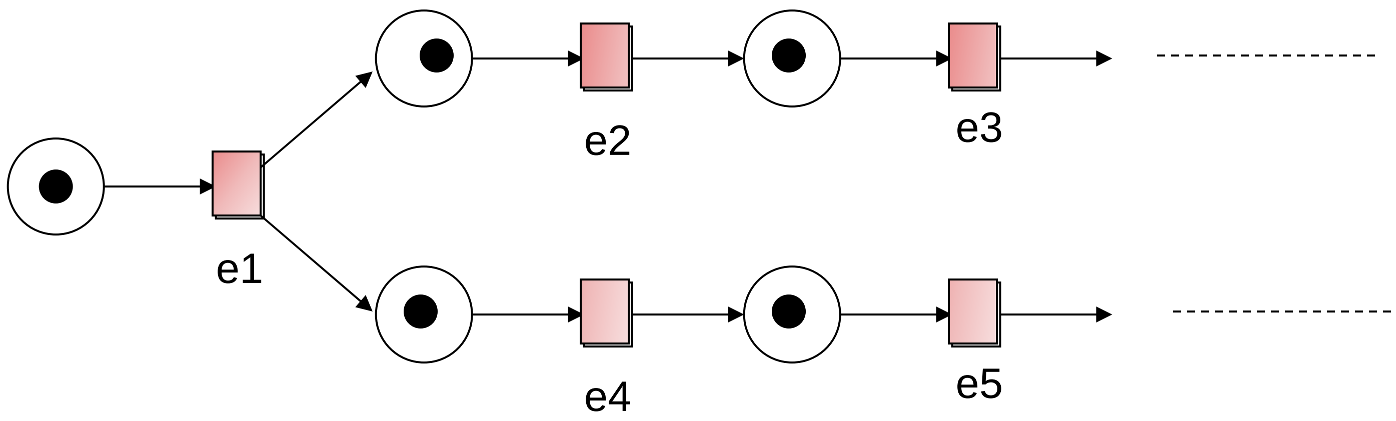


NET STRUCTURES

A sequence of events/actions:

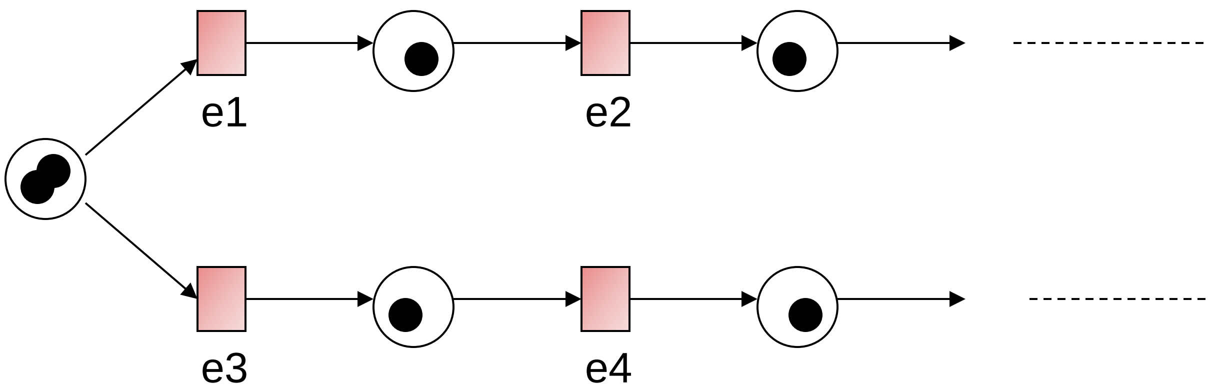


Concurrent executions:



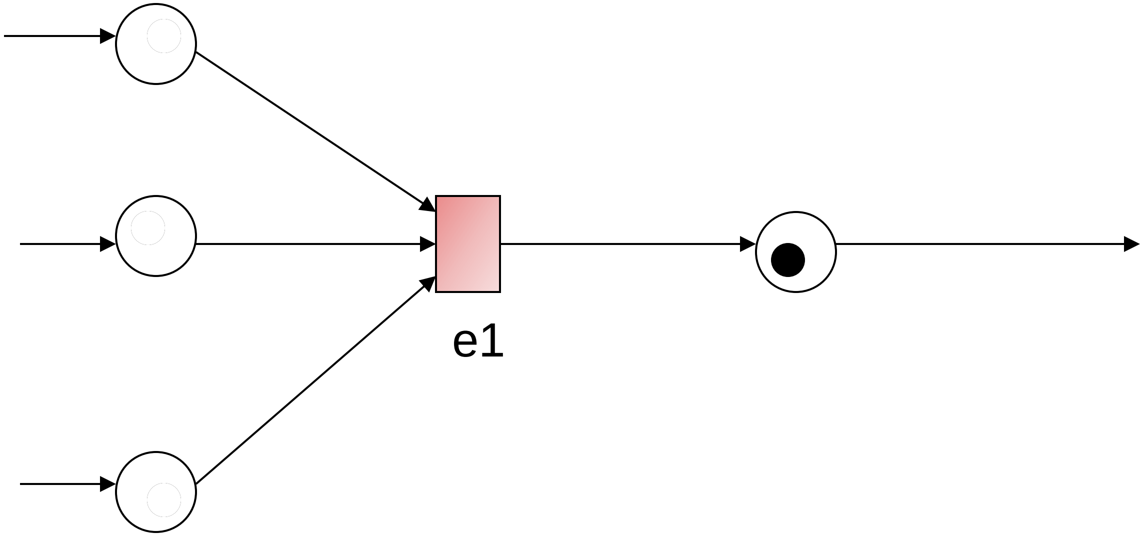
NET STRUCTURES

Non-deterministic events - conflict, choice or decision: A choice of either e1, e2 … or e3, e4 ...



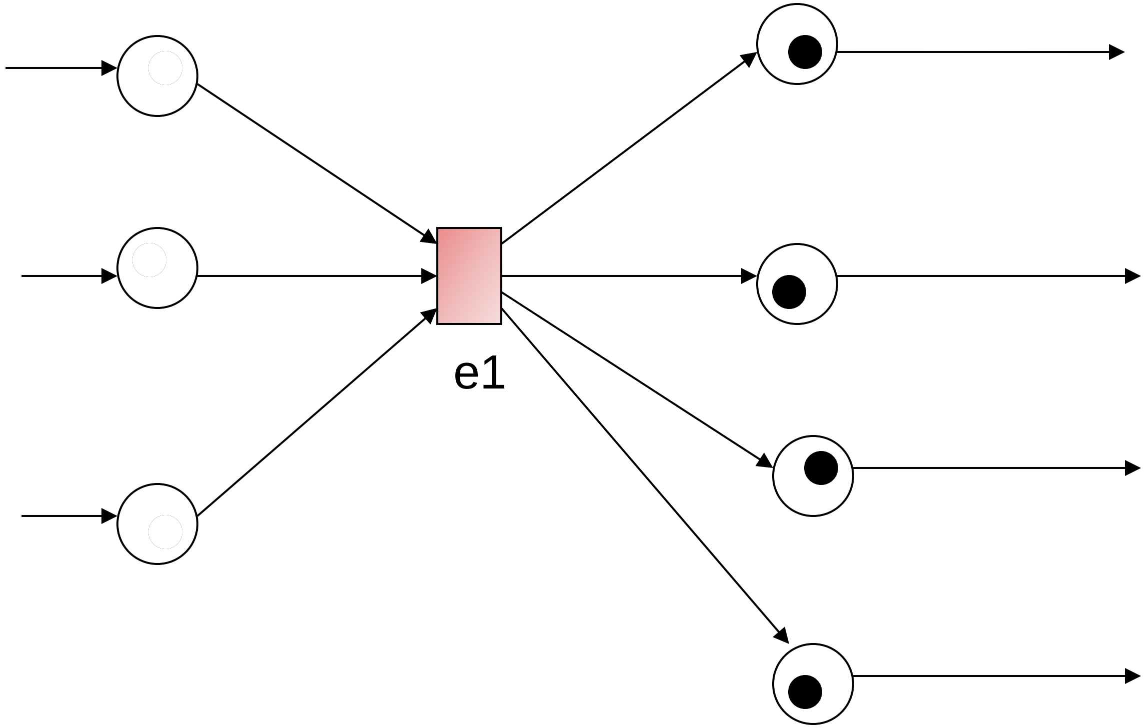
NET STRUCTURES

Synchronization



NET STRUCTURES

Synchronization and Concurrency



ANOTHER EXAMPLE

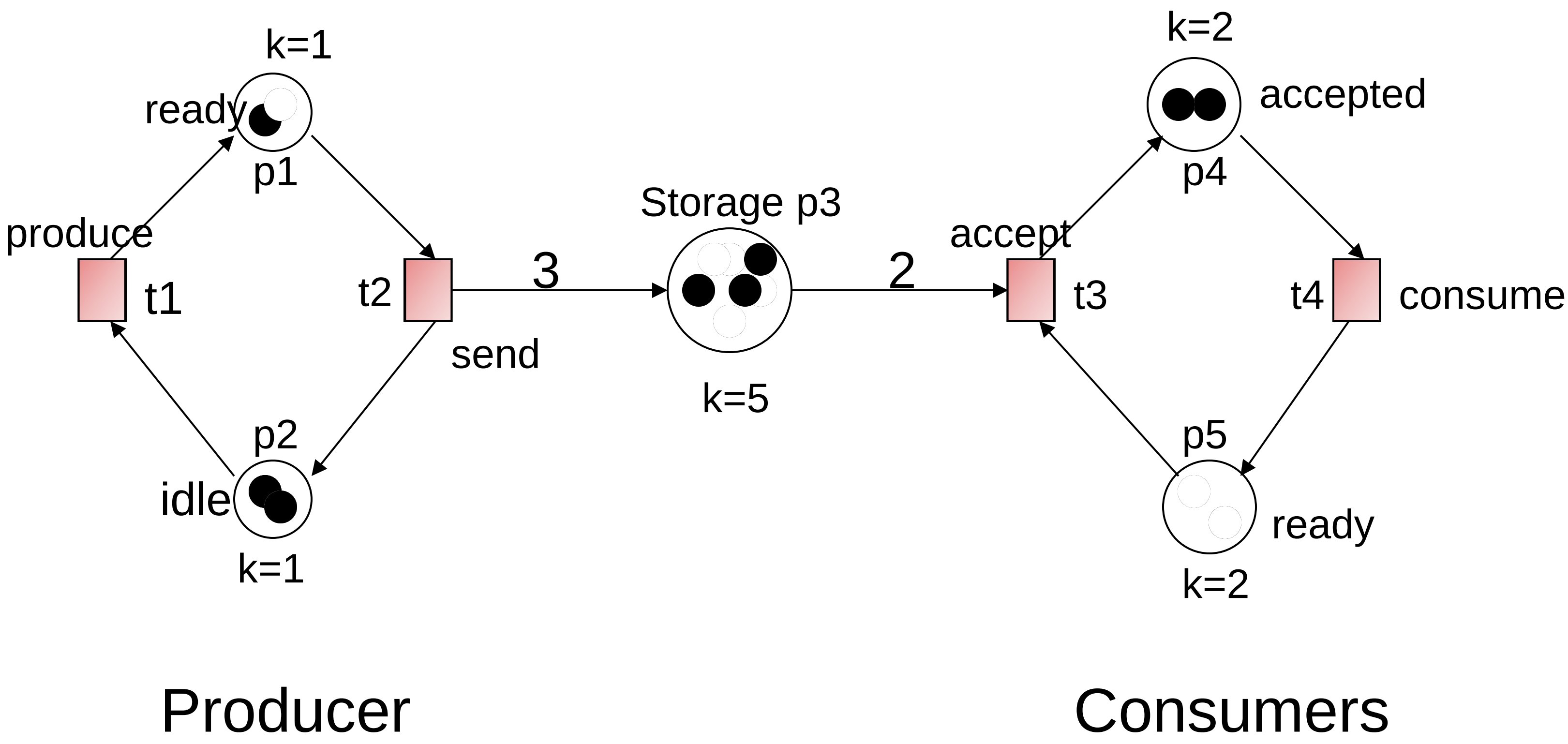
A producer-consumer system, consist of:

- One producer
- Two consumers
- One storage buffer

With the following conditions:

- The storage buffer may contain at most 5 items;
- The producer sends 3 items in each production;
- At most one consumer is able to access the storage buffer at one time;
- Each consumer removes two items when accessing the storage buffer

A PRODUCER-CONSUMER SYSTEM



A PRODUCER-CONSUMER EXAMPLE

uOttawa
L'Université canadienne
Canada's university

In this Petri net, every place has a capacity and every arc has a weight.

This allows multiple tokens to reside in a place to model more complex behavior.

BEHAVIORAL PROPERTIES

Reachability

- "Can we reach one particular state from another?"

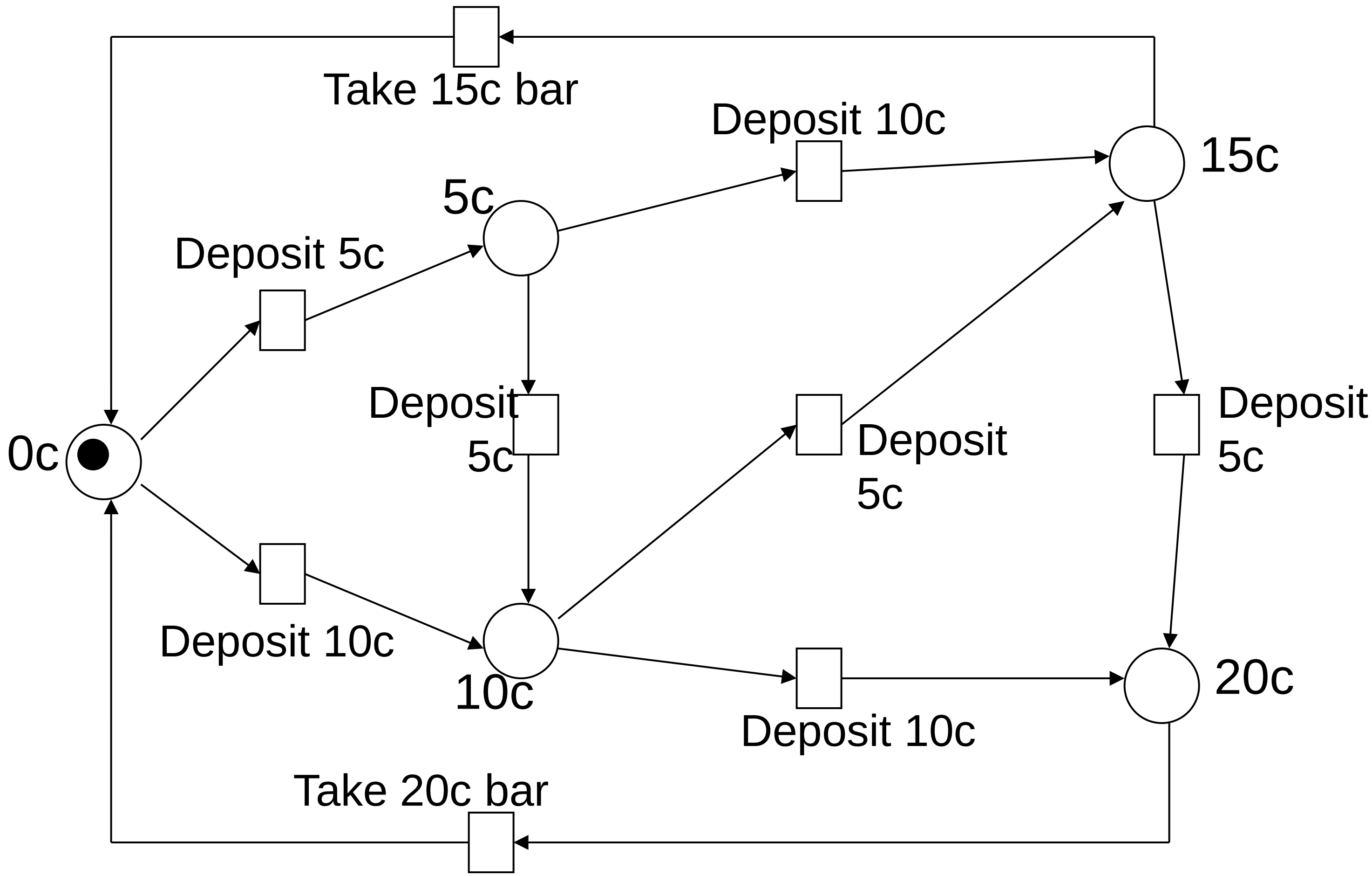
Boundedness

- "Can the number of tokens in a place increase infinitely?"

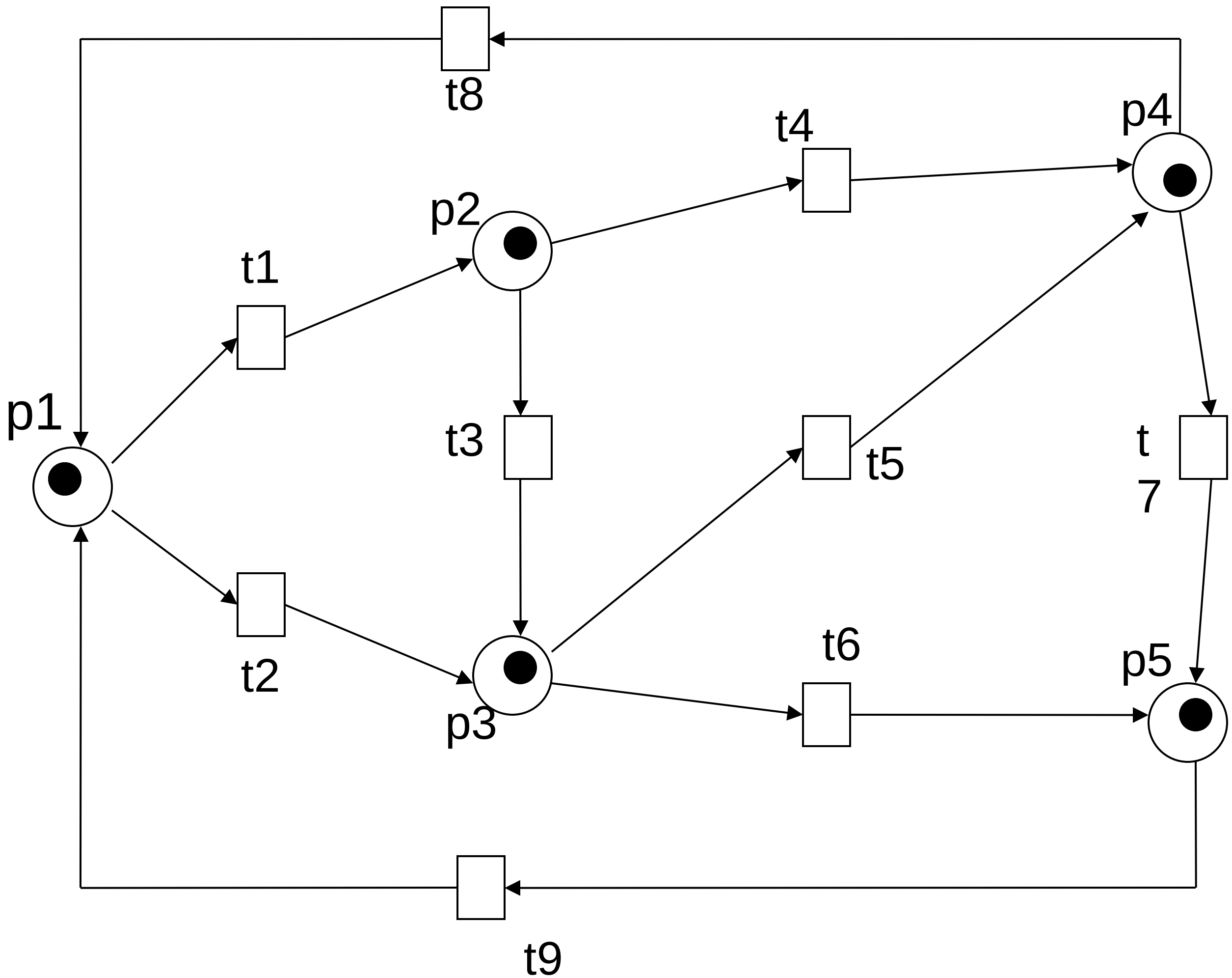
Liveness

- "Will the system die in a particular state?"

RECALLING THE VENDING MACHINE (TOKEN GAME)



A *MARKING* IS A STATE …



M0 = (1,0,0,0,0)

M1 = (0,1,0,0,0)

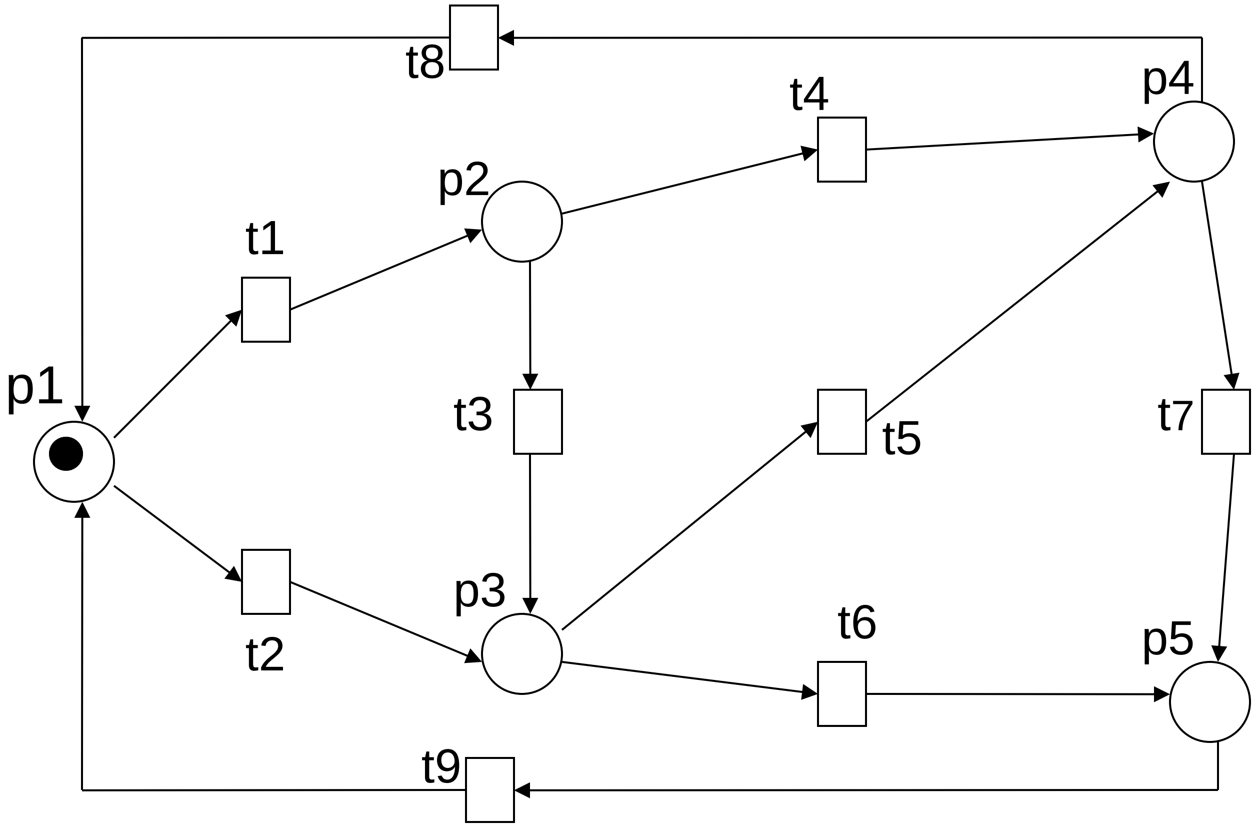
M2 = (0,0,1,0,0)

M3 = (0,0,0,1,0)

M4 = (0,0,0,0,1)

Initial marking:M0

REACHABILITY



M0 = (1,0,0,0,0)

M1 = (0,1,0,0,0)

M2 = (0,0,1,0,0)

M3 = (0,0,0,1,0)

M4 = (0,0,0,0,1)

Initial marking:M0

$$M0 \xrightarrow{t1} M1 \xrightarrow{t3} M2 \xrightarrow{t5} M3 \xrightarrow{t8} M0 \xrightarrow{t2} M2 \xrightarrow{t6} M4$$

REACHABILITY

A firing or occurrence sequence:

$$M0 \xrightarrow{t1} M1 \xrightarrow{t3} M2 \xrightarrow{t5} M3 \xrightarrow{t8} M0 \xrightarrow{t2} M2 \xrightarrow{t6} M4$$

"M2 is *reachable* from M1 and M4 is *reachable* from M0."

In fact, in the vending machine example, all markings are reachable from every marking.

BOUNDEDNESS

A Petri net is said to be *k-bounded* or simply *bounded* if the number of tokens in each place does not exceed a finite number *k* for any marking reachable from M0.

The Petri net for vending machine is 1-bounded.

LIVENESS

A Petri net with initial marking M0 is *live* if, no matter what marking has been reached from M0, it is possible to ultimately fire *any* transition by progressing through some further firing sequence.

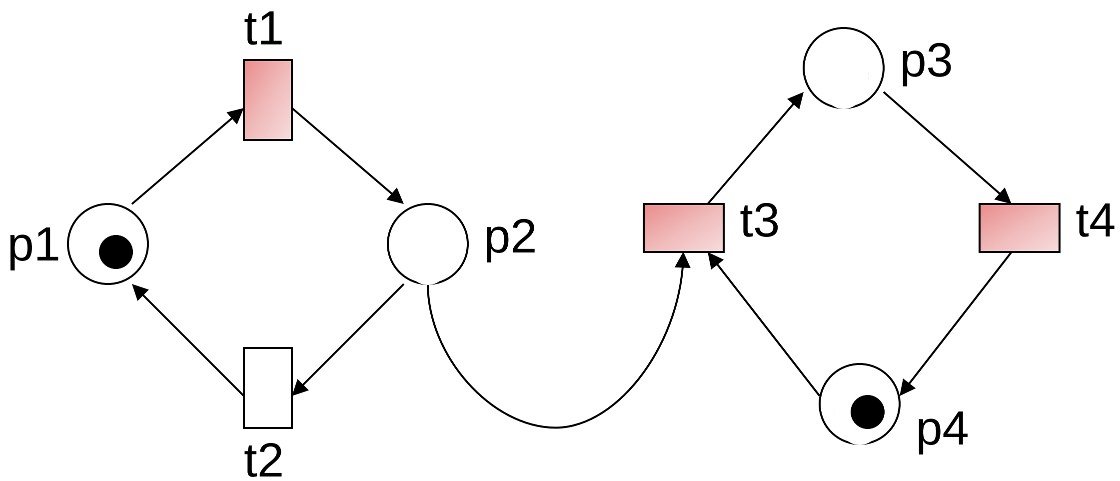
A live Petri net guarantees *deadlock-free* operation, no matter what firing sequence is chosen.

LIVENESS

The vending machine is live and the producer-consumer system is also live.

A transition is *dead* if it can never be fired in any firing sequence.

AN EXAMPLE



M0 = (1,0,0,1)

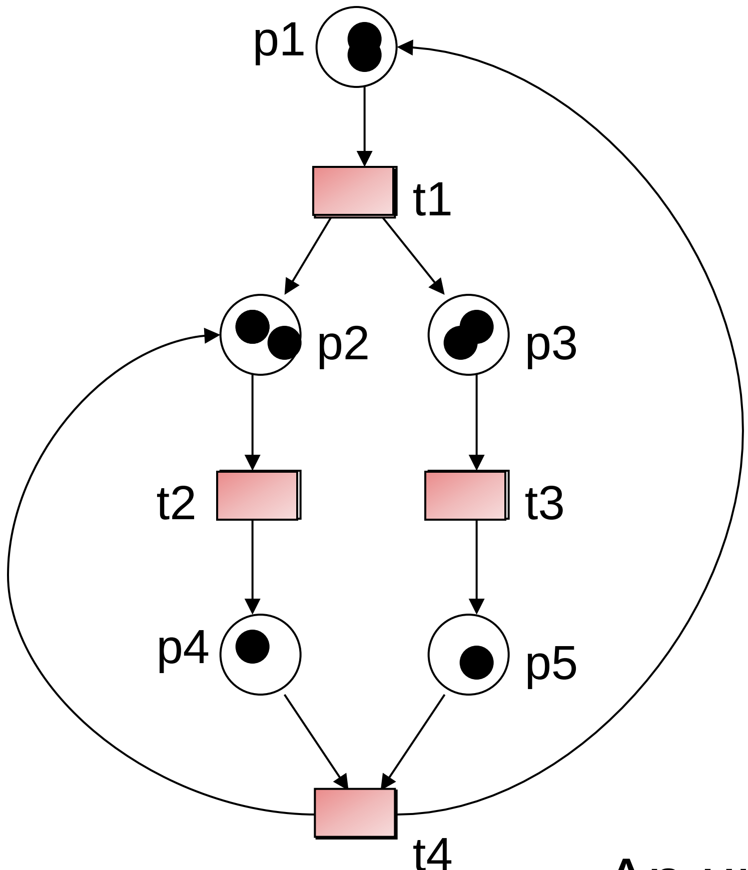
M1 = (0,1,0,1)

M2 = (0,0,1,0)

M3 = (0,0,0,1)

A bounded but non-live Petri net

ANOTHER EXAMPLE



M0 = (1, 0, 0, 0, 0)

M1 = (0, 1, 1, 0, 0)

M2 = (0, 0, 0, 1, 1)

M3 = (1, 1, 0, 0, 0)

M4 = (0, 2, 1, 0, 0)

An unbounded but live Petri net

THANK YOU!

QUESTIONS?