Software Requirements, Specifications and Formal Methods

A/Prof. Lei Niu

UNIVERSITY OF WOLLONGONG AUSTRALIA

Introduction to Petri Net

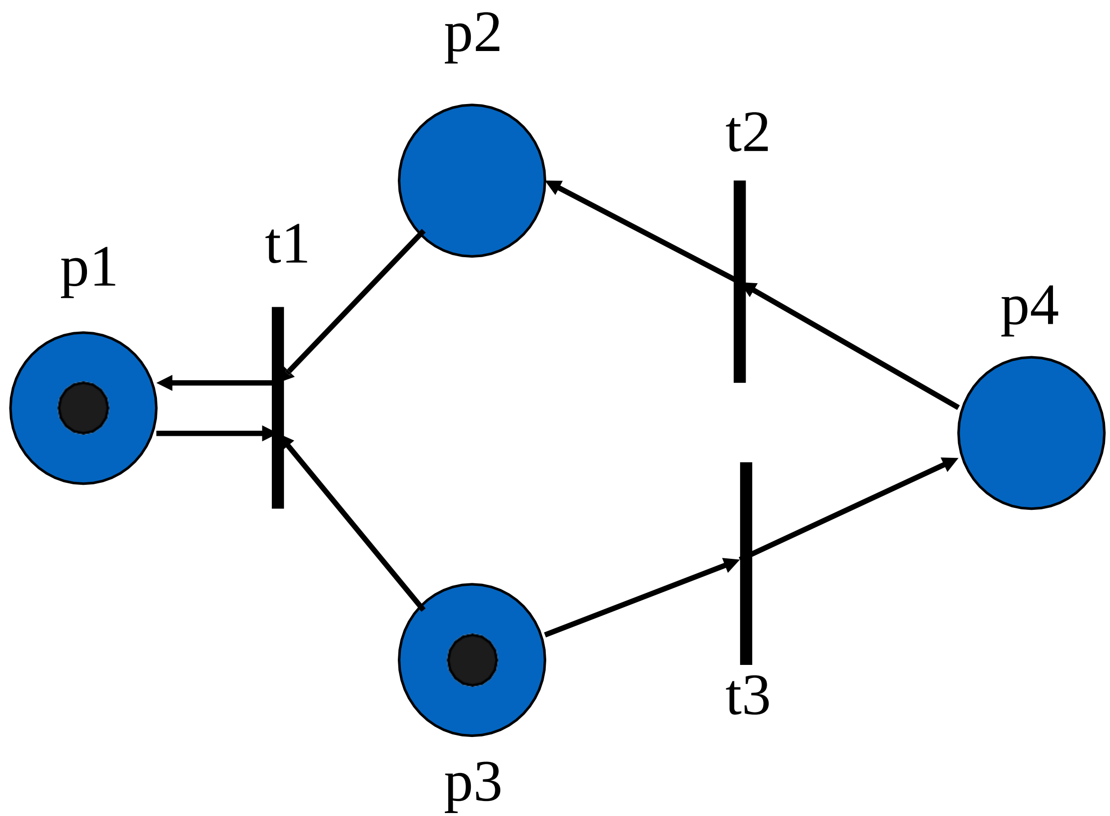
- Invented by Dr. Carl Adam Petri
- First introduced in Petri's dissertation: "Kommunikation mit Automaten" (1962)
- Petri Net is a tool for the study of system, and a mathematical representation of the system.

Why Petri Net (PN)?

- Petri Nets give a graph-theoretic representation of the structure and the dynamics of a discrete event system.
- They provide a mathematical framework for:
  - Analysis
  - Validation
  - Performance evaluation
- They focus on issues of
  - Concurrency
  - Asynchronous operations

UNIVERSITY OF WOLLONGONG AUSTRALIA

An example of PN



PN definition

A Petri Net is a bipartite directed graph, $G=(V,A)$, where $V=\{V_1, V_2, \ldots, V_s\}$ is a set of vertices and $A = \{ a1, a2, \ldots, a_r\}$ is a bag of directed arcs, $a_i = (v_j, v_k)$ with $v_j, v_k \in V$.

The set V are partitioned into two disjoint sets P *(circle nodes, i.e. places)* and T *(bar nodes, i.e. transitions)* such that $V = P \cup T$, $P \cap T = \varnothing$, and for each directed arc, $a_i \in A$, if $a_i = (v_j, v_k)$, then either $v_j \in P$ and $v_k \in T$ or $v_j \in T$ and $V_k \in P$.

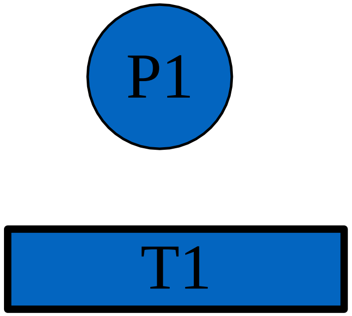
PN composition

A Petri Net is a bipartite directed graph

– **Bipartite**: two types of nodes

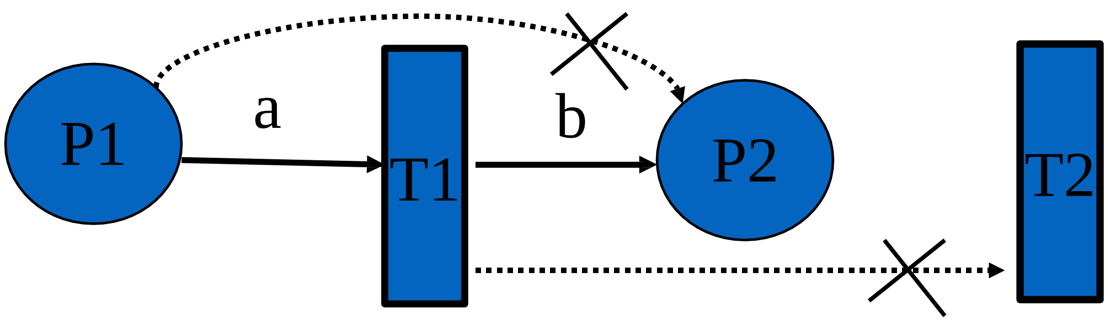
» *Circle nodes* denotes places

» *Bar nodes* denotes transitions



– **Directed**: the arcs that join two nodes are directed.

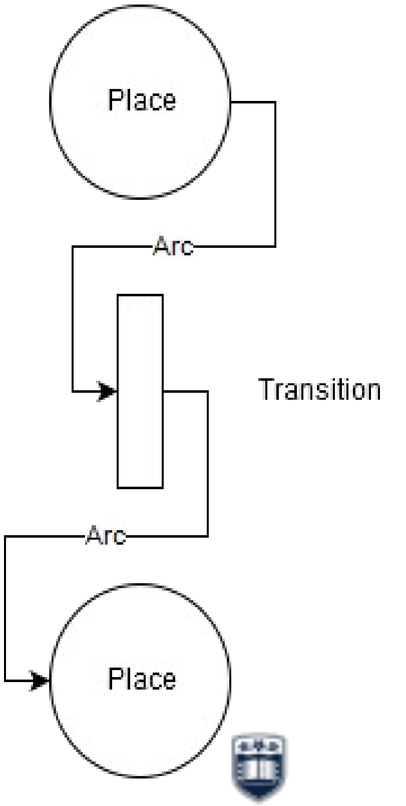
» *Arcs* connect places to transitions and transitions to places only



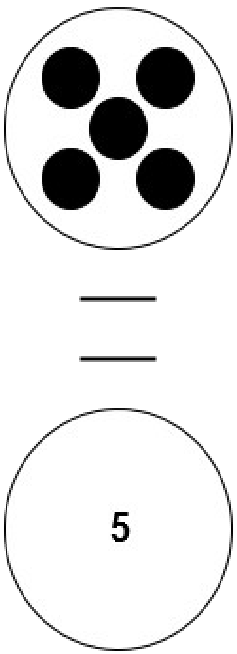
Place

A petri net is drawn as a directed, weighted, bipartite graph.

- A bipartite graph is a graph with two distinct sets of nodes such that there are **no edges between nodes of the same set**.
- In the case of petri nets, these two sets are defined as **places** and **transitions**.
- Typically, places are represented as circles and transitions as either bars or boxes.
- The edges between nodes are defined as **arcs**.



Marking and Token

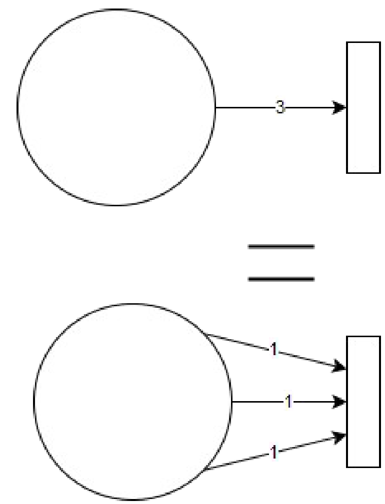


A petri net has **states**, designated as **markings**. Each marking corresponds to an assignment of some non-negative integer **k** to each place **p**.

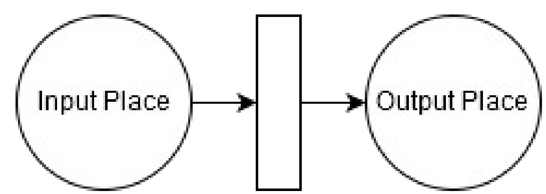
- This corresponds to the place being 'marked' with k **tokens**, which are represented on the graph typically as smaller black circles within each place.
- Normally, there is no limit to the number of tokens which may mark a given place.

Arc

- An arc, either from a place to a transition or from a transition to a place, has some weight **k**.
  - An arc with weight k is functionally the same as there being k arcs of weight 1.



- A place with an arc from itself to a transition is an **input place**, and a place with an arc from a transition to itself is an **output place**.

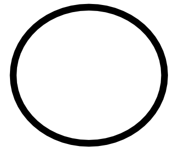


Role of a token

Tokens can play the following roles:

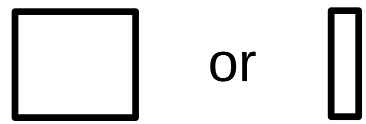
- a **physical object**, for example a product, a person;
- an **information object**, for example a message, a signal;
- a **collection of objects**, for example a truck with products, a warehouse with parts, or an address file;
- an **indicator of astate**, for example the indicator of the state in which a process is, or the state of an object;
- an **indicator of acondition**: the presence of a token indicates whether a certain condition is fulfilled.

Role of a place



- a type of **communication medium**, like a telephone line, a middleman, or a communication network;
- a **buffer**: for example, a depot, a queue or a post bin;
- a **geographical location**, like a place in a warehouse, office or hospital;
- a possible **state or state condition**: for example, the floor where an elevator is, or the condition that a specialist is available.

Role of a transition



- an **event**: for example, starting an operation, the death of a patient or the switching of a traffic light from red to green;
- a **transformation of an object**, like adapting a product, updating a database, or updating a document;
- a **transport of an object**: for example, transporting goods, or sending a file.

Petri Net Structure

- A Petri net is composed of four parts: a set of *places P,* a set of *transitions T*, an *input function I*, and an *output function O*.

- Input and output functions relate to transitions and places.
- Input function *I* is a mapping from a *transition t_j to a collection of places $I(t_j)$,* known as the input places of the transition.
- Output function *O* maps *a transition t_j to a collection of places $O(t_j)$,* known as the output places of the transition.

- The structure of a Petri net is defined by its places, transitions, input function, and output function.

Mathematical Definition

- A *Petri net structure*, C, is a four-tuple, $C=(P,T,I,O)$. $P=\{p_1,p_2,\ldots,p_n\}$ is a finite set of places, $n \geq 0$.
- $T=\{t_1,t_2,\ldots,t_m\}$ is a set of transitions, $m \geq 0$.
  - The set of places and the set of transitions are disjoint, i.e., $P \cap T = \emptyset$.
- $I: T \rightarrow P^{\infty}$ is the input function, a mapping from transitions to bags of places, and indicates the input places of a transition.
- $O: T \rightarrow P^{\infty}$ is the output function, a mapping from transitions to bags of places, and indicates the output places of a transition.

Example

A Petri net structure:

C = (P,T,I,O) *P={p1,p2,p3,p4,p5}* *T={t1,t2,t3,t4}*

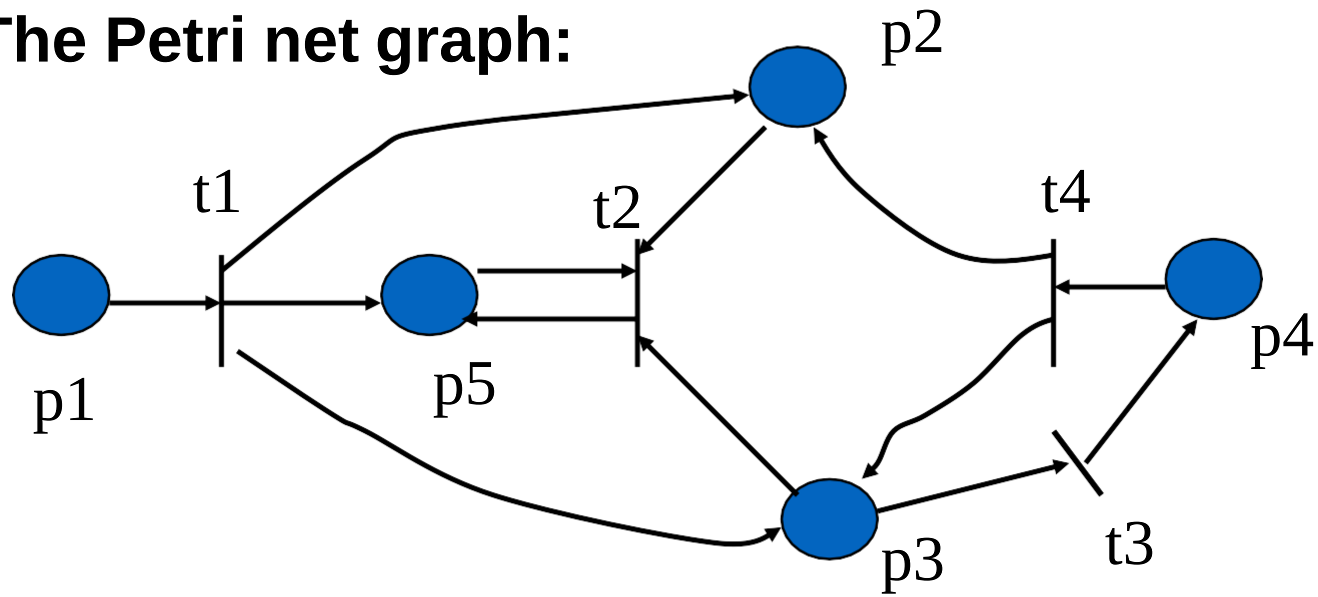
I(t1) = {p1} *O(t1) = {p2,p3,p5}*

I(t2) = {p2,p3,p5} *O(t2) = {p5)*

I(t3) = {p3} *O(t3) = {p4}*

I(t4) = {p4} *O(t4) = {p2,p3}*

The Petri net graph:



Marked PN

- A marked PN contains tokens
- Tokens are depicted graphically by dots (•) and reside in places
- A marking of a PN is a mapping that assigns a non-negative integer (the number of tokens) to each place of the net
- The marking characterizes the state of the Petri Net
- The initial marking is referred to as μ

Definition for marking μ

- Definition: *A marking* μ of a Petri net $C = (P,T,I,O)$ is a function from the set of places P to the non-negative integers N.
- $\mu: P \rightarrow N$

- The marking μ can also be defined as an n-vector, $\mu = (\mu_1, \mu_2, \ldots, \mu_n)$.
- The number of tokens in place p_i is μ_i, $i=1, \ldots, n$.
- The definitions of a marking as a function and as a vector are obviously related by $\mu(p_i) = \mu_i$.

Petri Net State Spaces

- The state of a Petri net is defined by its *marking*.
- The firing of a transition represents a change in the state of the Petri net.
- The state space of a Petri net with n places is the set of all markings.

Example of Petri Net Markings

A Petri net structure:

C = (P,T,I,O) *P={p1,p2,p3,p4,p5}* *T={t1,t2,t3,t4}*

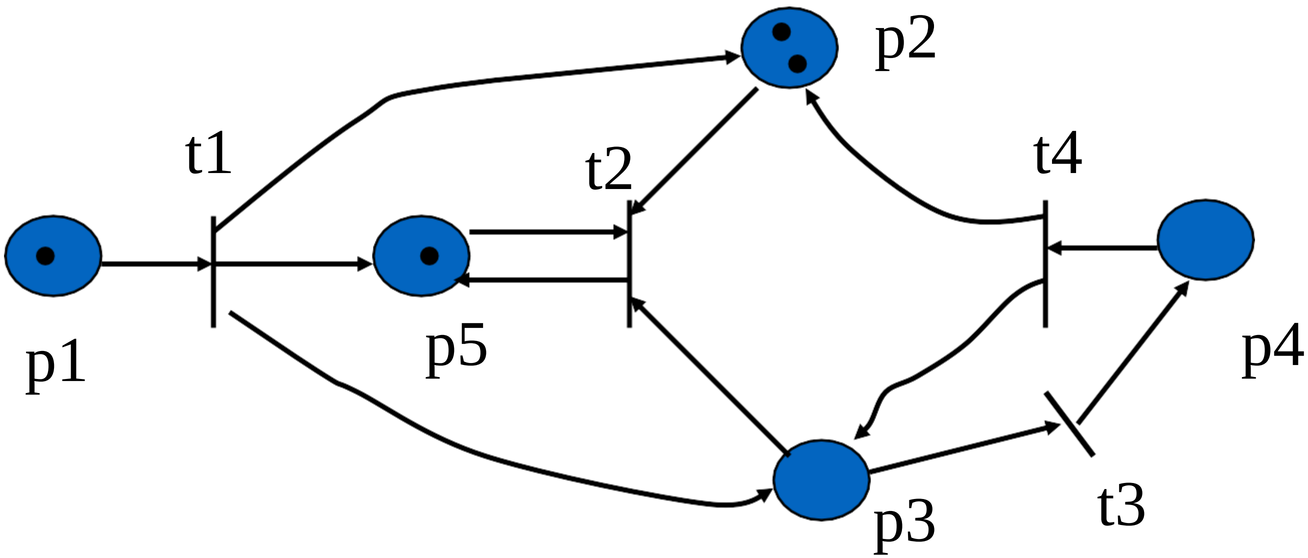
I(t1) = {p1} *O(t1) = {p2,p3,p5}*

I(t2) = {p2,p3,p5} *O(t2) = {p5)*

I(t3) = {p3} *O(t3) = {p4}*

I(t4) = {p4} *O(t4) = {p2,p3}*

The marking is μ=(1,2,0,0,1)

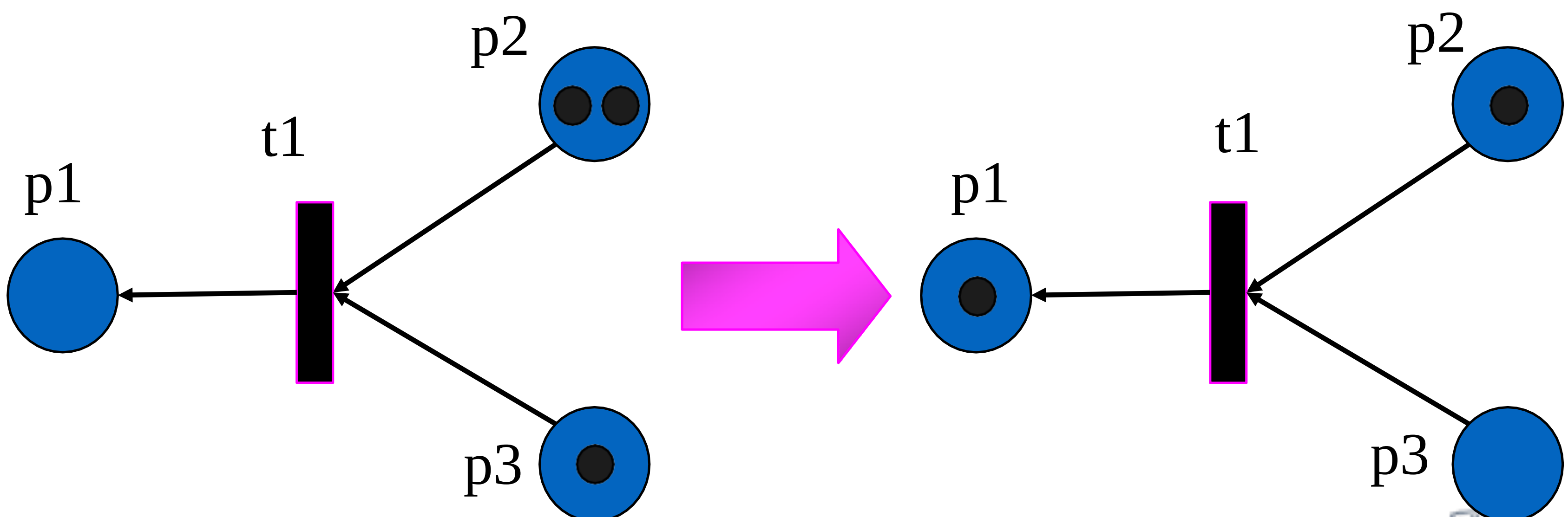


Execution Rules for Petri Nets

- A transition t is called *enabled* in a certain marking, if:
  - For every arc from an input place p to transition t, there exists a distinct token in the marking
- An enabled transition can be *fired* and result in a new marking
- Firing of a transition *t* in a marking is an atomic operation

Execution Rules for Petri Nets (cont.)

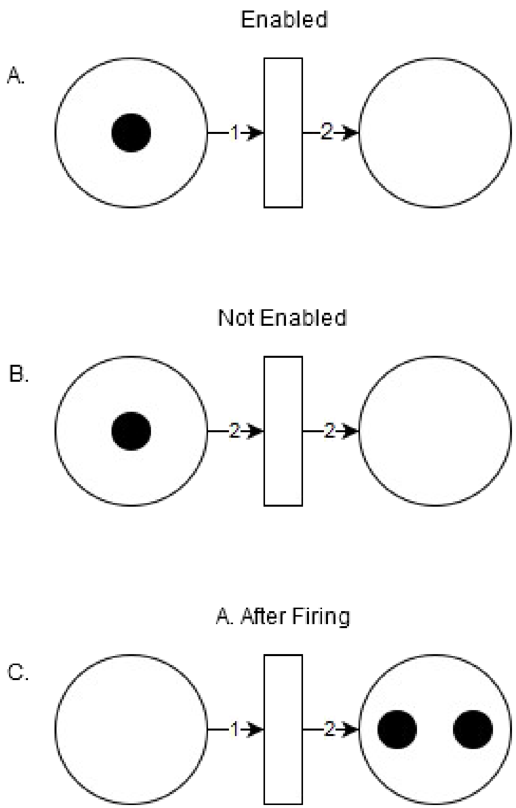
- Firing a transition results in two things:
    1. Subtracting one token from the marking of any input place p for every arc connecting place p to transition t, i.e. decrease token for all I(t)
    2. Adding one token to the marking of any output place p for every arc connecting transition t to place p, i.e. increment token for all O(t)



Enabled and Firing

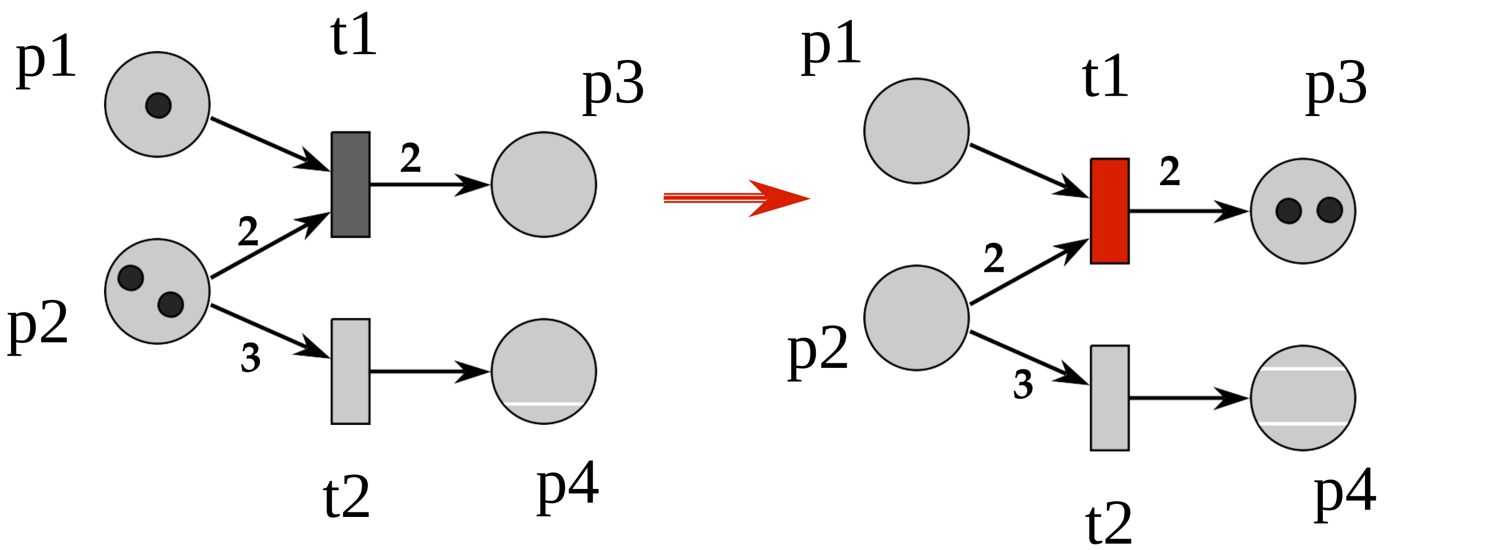
In Petri Nets, there is a **firing** rule, which has three parts.

1. A transition **t** is enabled if each input place **p** of **t** is marked with at least w((p,t)) tokens (where w((p,t)) is the weight of the arc from **p** to **t**).
2. An enabled transition **t** may or may not fire.
3. The firing of an enabled transition **t** removes w((p,t)) tokens from each input place **p** of **t** and adds w((t,p')) tokens to each output place **p'** of **t**.



UNIVERSITY OF WOLLONGONG AUSTRALIA

Enabled and Firing (cont.)



Definitions

A transition $tj \in T$ in a marked Petri net $C = (P,T,I,O)$ with marking μ is enabled if for all $pi \in P$,

$$\mu(pi) \quad \geq \quad \#(pi, tj)$$

number of token within place pi   *number of arcs from input place pi to transition tj*

A transition tj in a marked Petri net with marking μ may fire whenever it is enabled. Firing an enabled transition tj results in a new marking μ' defined by, for all $pi \in P$,

$$\mu'(pi) = \mu(pi) - \#(pi, tj) + \#(tj, pi)$$

new tokens in pi = original tokens in pi - tokens removed from pi + tokens added to pi

Non-determinism

- Even a transition is enable, the execution of transition is non-deterministic.

  1. Multiple transitions can be enabled at the same time, any one of them can be fired

  2. None are required to fire - they fire at will, between 0 times and infinity, or not at all

Example of Execution Rules for Petri Nets

Condition: for all $p_i \in P$, $\mu(p_i) \geq \#(p_i, t_j)$

μ=(1,0,0,2,1**),**

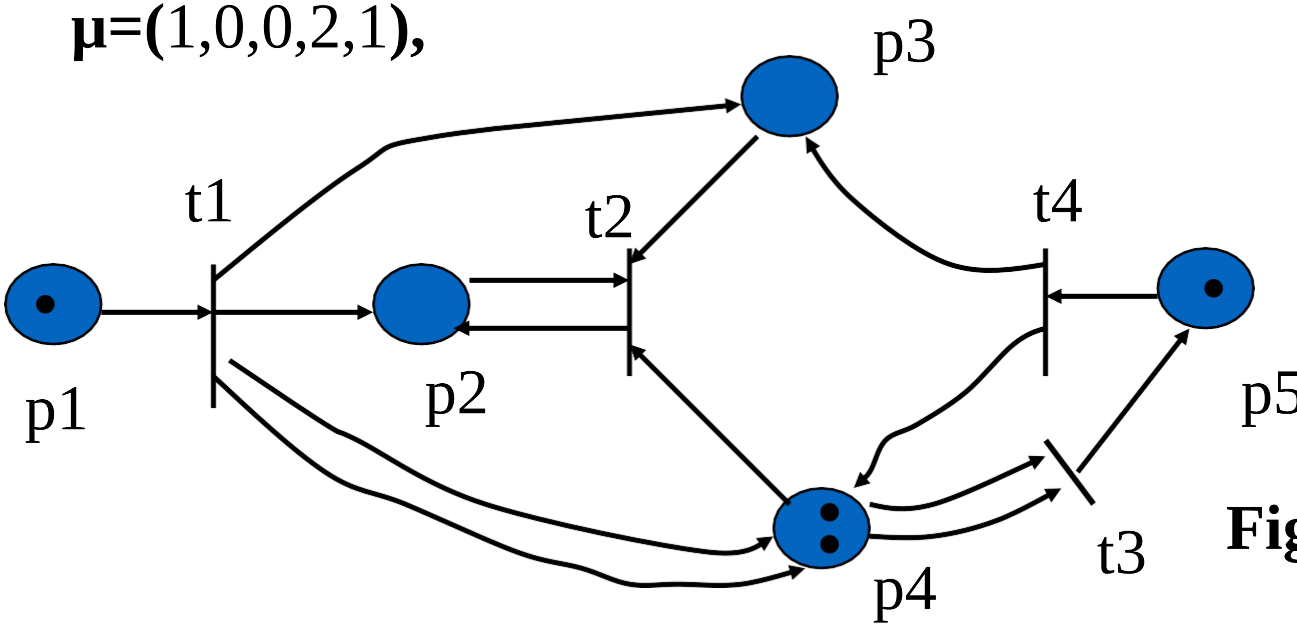


Figure 1

Transitions t1, t3, and t4 are enabled

Because the execution of PN is non-determinism, we assume that the fire order is t4, t1, t3

Example of Execution Rules for Petri Nets

The marking result from **<u>firing transition t4 </u>** in Figure 1

$\mu'(pi) = \mu(pi) - \#(pi, tj) + \#(pi, tj)$

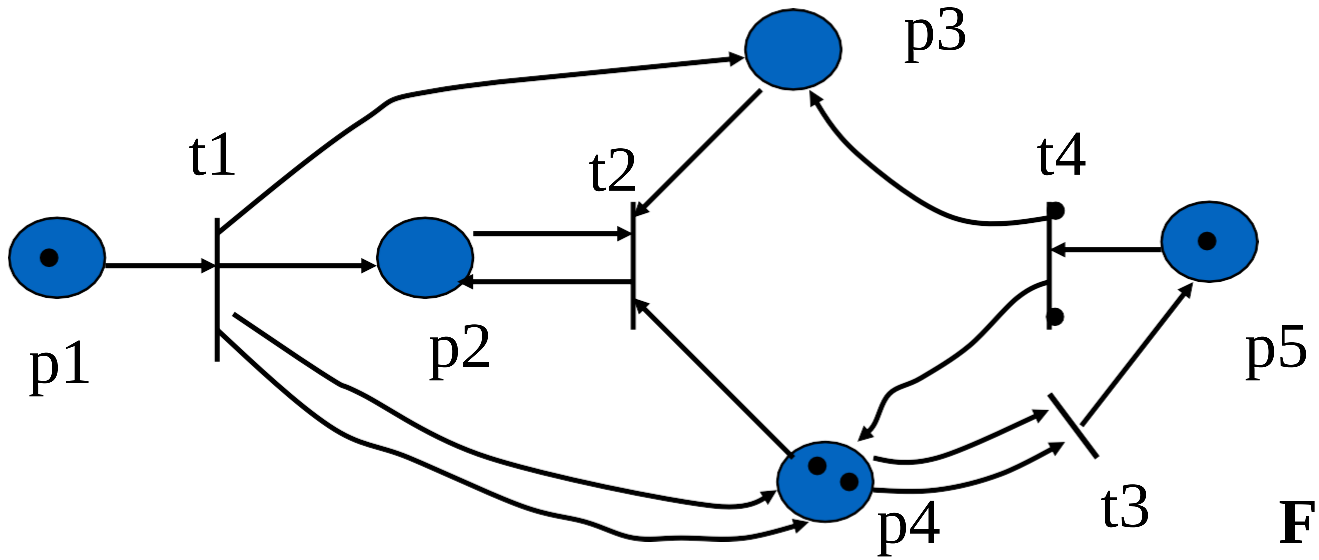


Figure 2

μ=(1,0,0,2,1**)** → **μ1=(**1,0,1,3,0**)** **(**firing transition t4**)**

Example of Execution Rules for Petri Nets (continuing)

The marking result from **firing transition t1** in Figure 2

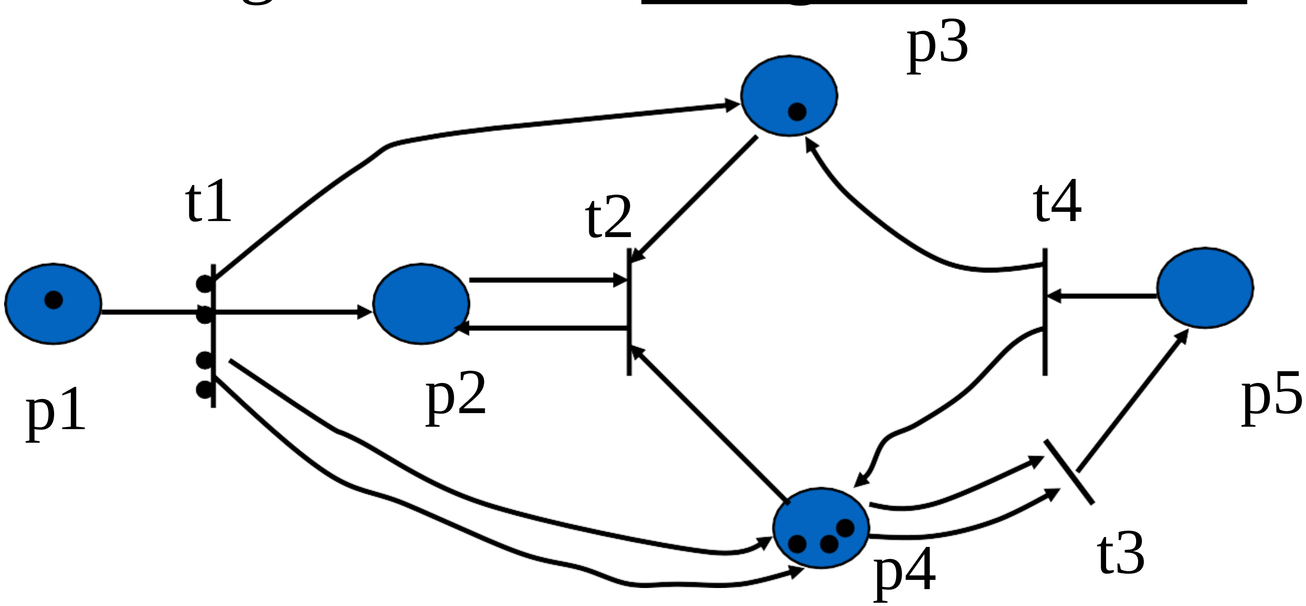


Figure 3

μ1=(1,0,1,3,0**)** → **μ2=(**0,1,2,5,0**)** **(**firing transition t1**)**

Example of Execution Rules for Petri Nets (continuing)

The marking result from **firing transition t3** in Figure 3

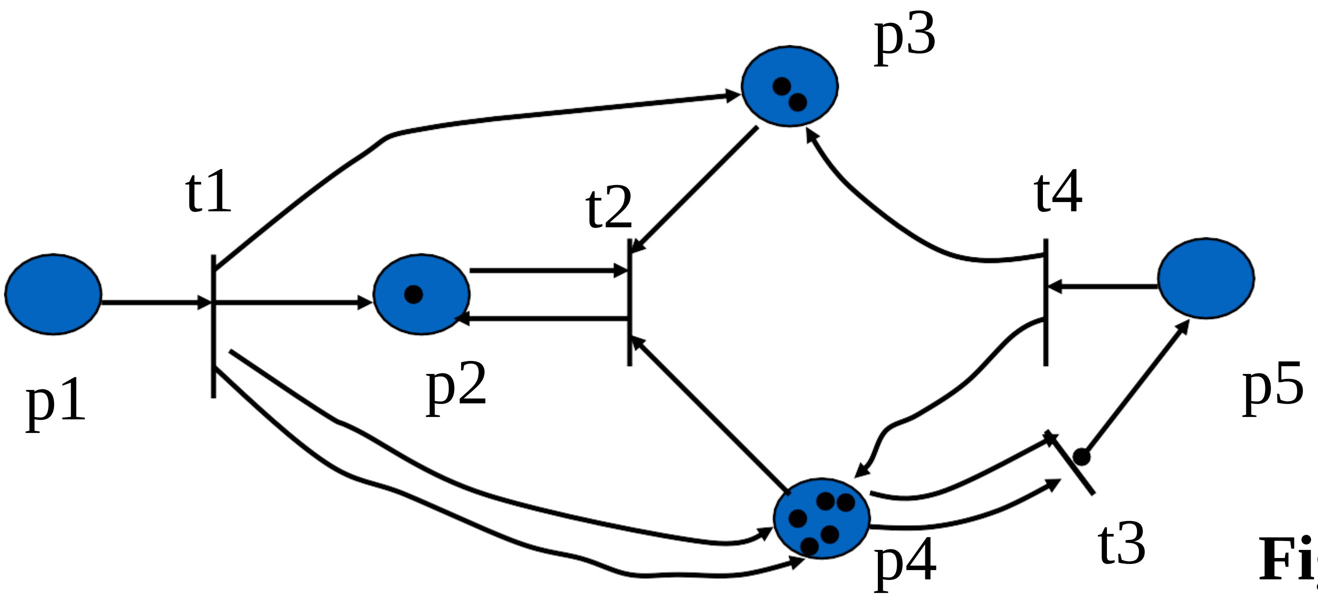
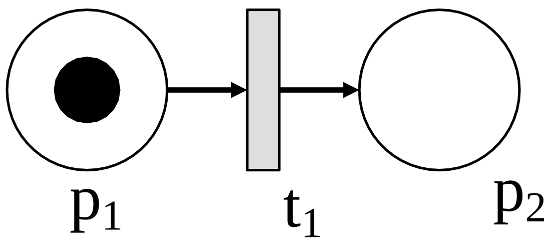


Figure 4

μ2=(0,1,2,5,0) → **μ3=(**0,1,2,3,1**)** **(**firing transition t3**)**

Examples

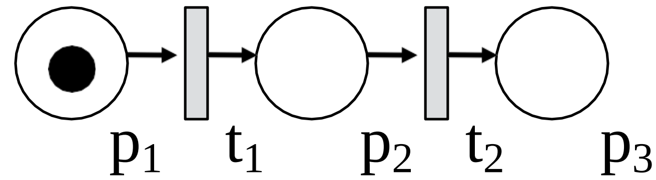
- Below is an example Petri net with two places and one transition.
- Transition node is ready to ***fire*** *if and only if there is at least one token at place p1*



State transition: **μ**=(1,0) → **μ1=(**0,1**)**

Examples (cont.)

$\mu=(1,0,0) \rightarrow \mu 1=(0,1,0) \rightarrow \mu 2=(0,0,1)$



- Sequential Execution

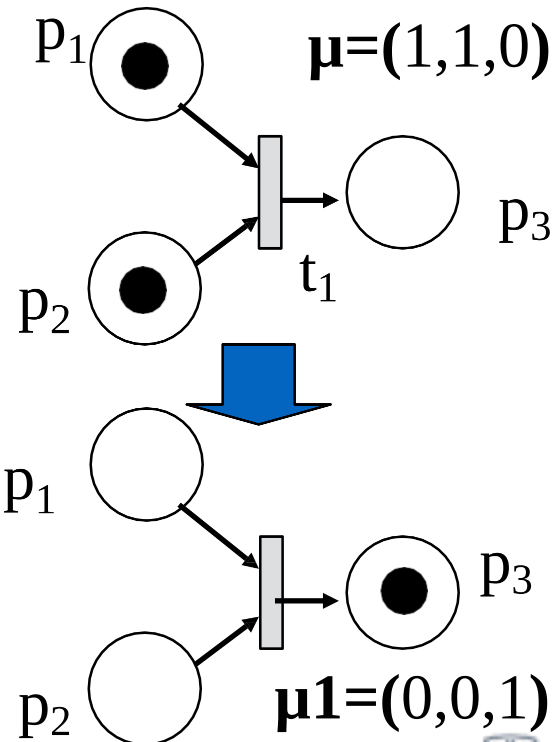
  Transition t_2 can fire only after the firing of t_1. This imposes the precedence of constrains "t_2 after t_1 ".

- Synchronization

  Transition t_1 will be enabled only when there are at least one token at each of its input places.

- Merging

  Happens when tokens from several places arrive for service at the same transition.

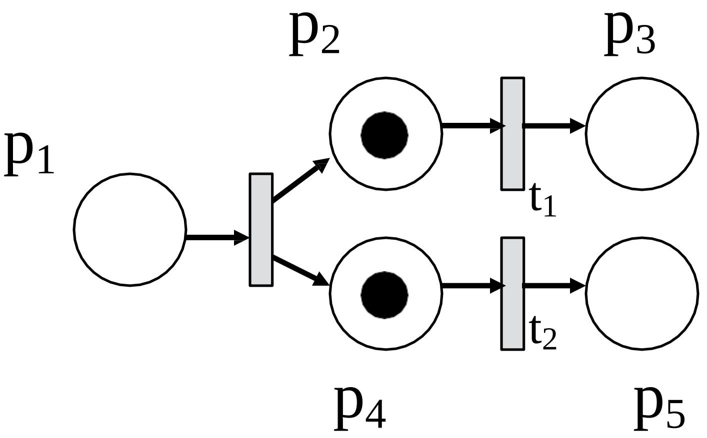


Examples (cont.)

- Concurrency

  t_1 and t_2 are concurrent.

  -with this property, Petri net is able to model systems of distributed control with multiple processes executing concurrently in time.



μ=(0,1,0,1,0**)**→ **μ1=(**0,0,1,0,1**)**

Petri Net Applications

- **performance evaluation**
- **communication protocols**
- distributed-software systems
- distributed-database systems
- concurrent and parallel programs
- industrial control systems
- discrete-events systems
- multiprocessor memory systems
- dataflow-computing systems
- fault-tolerant systems
- etc.

Conclusion

- Petri Nets
  - executable
  - concurrent, asynchronous, distributed, parallel, nondeterministic and/or stochastic systems
  - graphical tool
    - visual communication aid
  - mathematical tool
    - state equations, algebraic equations, etc
  - communication between theoreticians and practitioners