CS-417
COMPUTER SYSTEMS MODELING

Spring Semester 2020

Batch: 2016-17

(LECTURE # 27)

FAKHRA AFTAB
LECTURER
DEPARTMENT OF COMPUTER & INFORMATION SYSTEMS ENGINEERING
NED UNIVERSITY OF ENGINEERING & TECHNOLOGY

Recap of Lecture # 26

Discrete & Continuous Events Simulation

Simulation Algorithm

Generation of Random Numbers

Chapter # 8

PETRI NET-BASED PERFORMANCE MODELING

Prepared by: Ms. Fakhra Aftab (Lecturer, CISD, NEDUET)

INTRODUCTION

- Petri nets (PNs) provide a *graphical tool* as well as a *notational method* for the formal specification of systems.

- The systems PNs model tend to include more than simply an arrival rate and a service rate.

- Two important aspects in modeling the performance of computer systems (1) contention for resources and (2) synchronization between various concurrent activities.

- The former aspect can be adequately represented by Queuing Network Modeling, but the later cannot.

- Petri nets have long been used for modeling synchronization behavior of concurrent systems but not as completely as simulations.
- They act more like simulations in that they allow the modeler to examine single entities within the system, as well as their movement and effect on the state of entire system.
- Petri nets were first introduced in 1966 to describe concurrent systems.
- This initial introduction was followed by continual improvements for example,
  - the addition of timing to transitions,
  - priority to transitions,
  - types to tokens,
  - and colors depicting conditions on places and tokens.

GRAPHICAL REPRESENTATION

- Petri nets are usually represented graphically according to the following conventions:
  - *Places* are represented by *circles*,
  - *transitions* by *bars*,
  - *input function* by arcs directed from places to transitions,
  - *output function* by arcs directed from transitions to places, and
  - *markings* by small filled circles called *tokens*.

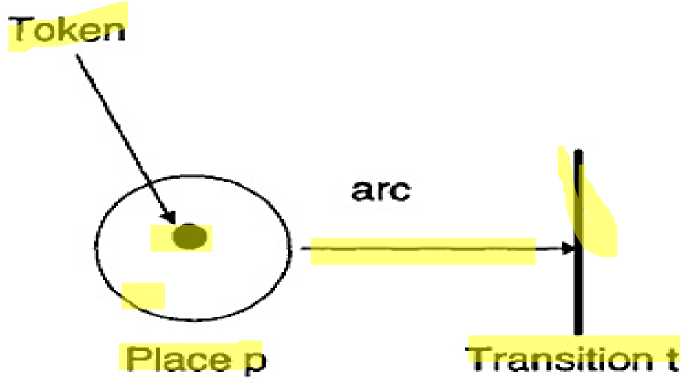

Fig 1: Basic Petri net components

Example

Fig 2 illustrates a simple Petri net with only one place and one transition.

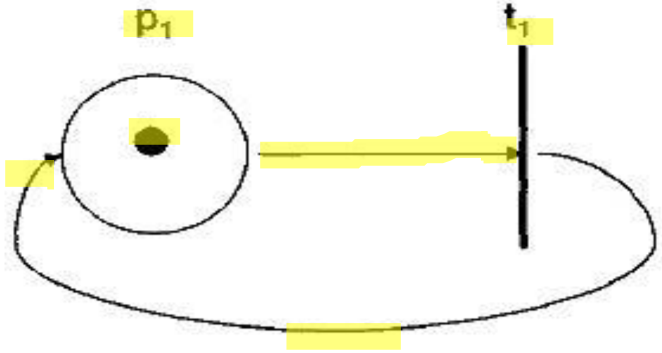

Fig 2: Example perpetual motion Petri net

- The former arc is input arc, while the later is an output arc.
- The placement of a token represents the active marking of the Petri net state.
- The Petri net shown in Fig 2 represents a net that will continue to cycle forever.

REPRESENTATION USING SET NOTATION

Using this notation, we can describe a PN as a 5-tuple, M= (P, T, I, O, MP)

- where P is the set of places, P = $\{P_1, P_2, \dots, P_n\}$
- T is the set of transitions, T= $\{t_1, t_2, \dots, t_m\}$
- $I \subset P \times T$ is the input function,
- $O \subset T \times P$ is the output function, and
- $MP \subset P \times Z^+$ is a marking of the net,
- Here, Z^+ denotes the set of all nonnegative integers,
- I represents a bag of sets of input functions for all transitions,
- I = $\{I_{t1}, I_{t2}, \dots, I_{tm}\}$, mapping places to transitions,

- O represents a bag of sets of output functions for all transitions,
- O = $\{O_{t1}, O_{t2}, \ldots, O_{tm}\}$, mapping transitions to places; and
- *MP* represents the marking of places with tokens.
- The initial marking is referred to as MP_o.
- MP_o is represented as an ordered tuple of magnitude n, where n represents the number of places in our Petri net.
- Each place will have either no tokens or some integer number of tokens.

For example, the Petri net graph depicted in Fig 3 can be described using the previous notation as:

- $M = (P, T, I, O, MP)$
- $P = \{p1, p2, p3, p4, p5\}$
- $T = \{t1, t2, t3, t4\}$
- $I(t1) = \{p1\}$
- $I(t2) = \{p2, p3, p5\}$
- $I(t3) = \{p_3\}$
- $I(t4) = \{p_4\}$
- $O(t_1) = \{p_2, p_3, p_5\}$
- $O(t_2) = \{p_5\}$
- $O(t_3) = \{p_4\}$
- $O(t_4) = \{p_2, p_3\}$

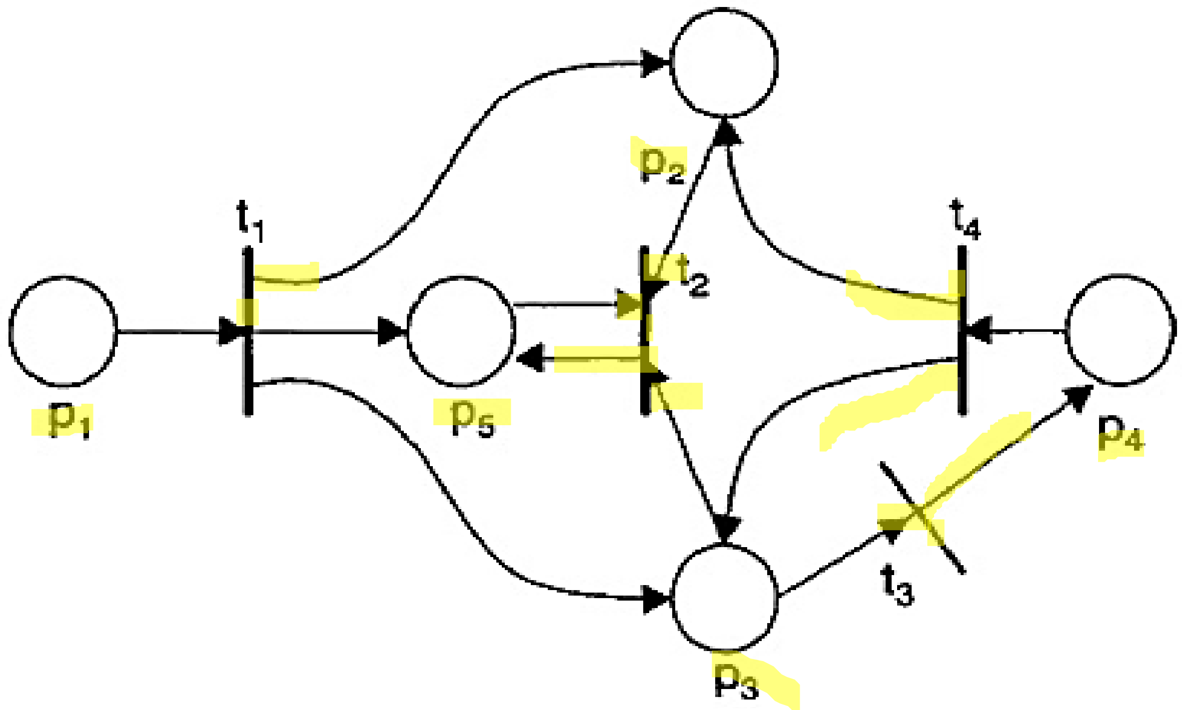

Fig 3: Petri net example

Dynamic Behaviour of Petri-Nets

- The dynamic behavior of a Petri net is described by the sequence of transition firings.
- The firing rules are as follows: Let *t* be a transition with incoming arcs from places ip_1 ,...,ip_K and outgoing arcs to places op_1,...:op_L for some K,L > 1.
- Then *t* can fire if and only if each of the K input places contains at least one token.
- As a result of firing, one token will be removed from each of the K input places, and one token will be added to each one of L output places.

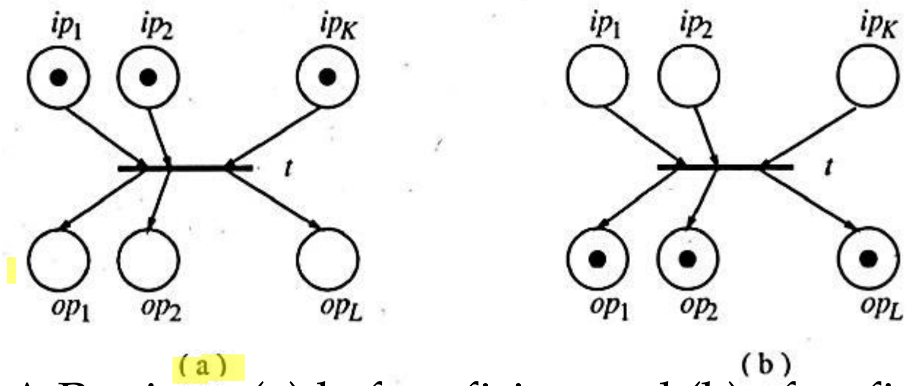

Fig 4: A Petri net (a) before firing and (b) after firing

DUAL OF A PETRI-NET

- The *dual* of a Petri net: transitions changed to places and places changed to transitions.

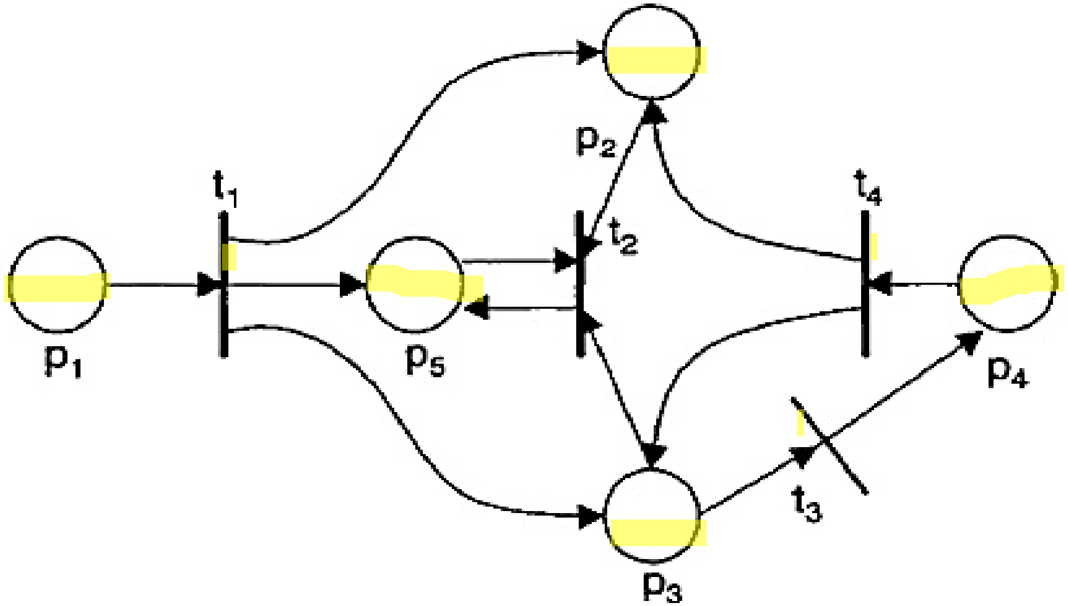

Fig 3: Petri Net Example

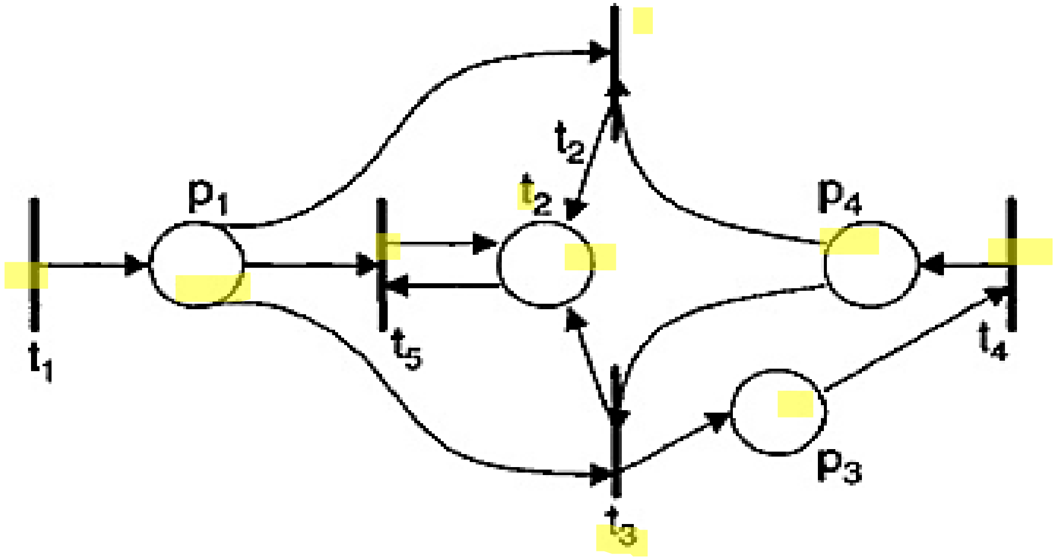

Fig 5: Dual of Petri Net from Fig 3