ECE 09468/09568
Discrete Event Systems

Lecture 8: **Petri Nets (PNs) – Part V**

Dr. Ying (Gina) Tang
Department of Electrical and Computer Engineering
Rowan University

Timed Petri Nets

- The concept of time is not given in the original definition of PN.
- PN without time can describe the *logic* structure and *behavior* of the modeled system, but not its evolution over time.
- Timed Petri Nets (TPN) are introduced for *performance evaluation* of dynamic systems.

Interpretation of Execution Rules in TPN

- If a delay d is associated with a place, after a token arrives in the place, it has to wait for d units before it can be used to enable its output transition.
- If d is associated with a transition, it takes d units for the transition to finish firing.
- If d is associated with an arc, then a token flows through arc in d units.
- A timed Petri net can be converted to a timed transition or place or arc Petri net.

Timed Petri Nets

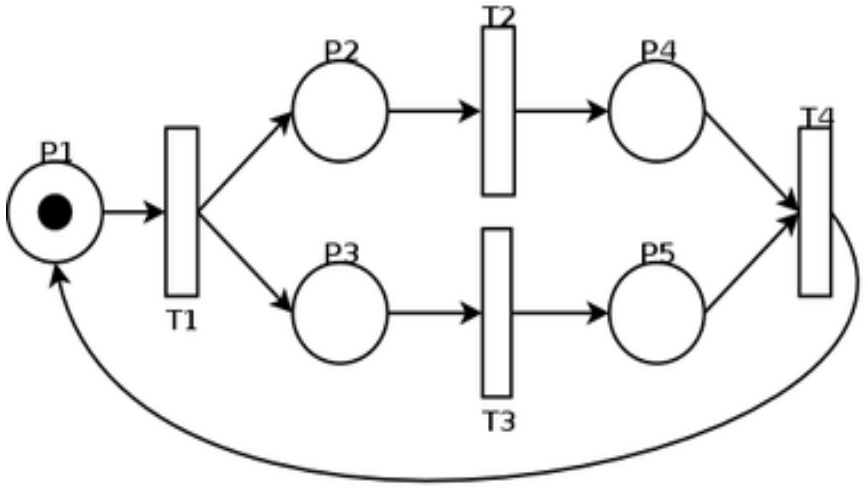
- Deterministic Timed PN (DTPN): Concurrent *choice-free* PN that associate *fixed time delays* to transitions, places, or arcs.
- Stochastic Petri Nets (SPN): When a firing time delay is characterized by a *random variable*, in particular, if these random variables are *exponentially distributed*, the resulting timed Petri nets are stochastic Petri nets.

DTPN – Timed Marked Graph

- Deterministic Timed PN: Concurrent choice-free PN that associate fixed time delays to transitions, places, or arcs.
- Choice-free PN is called **marked graphs** - each place has one incoming and one outgoing arc.
- A fundamental approach to studying DTPN is based on two concepts: total time delay in a loop and total number of tokens in a loop (token count).

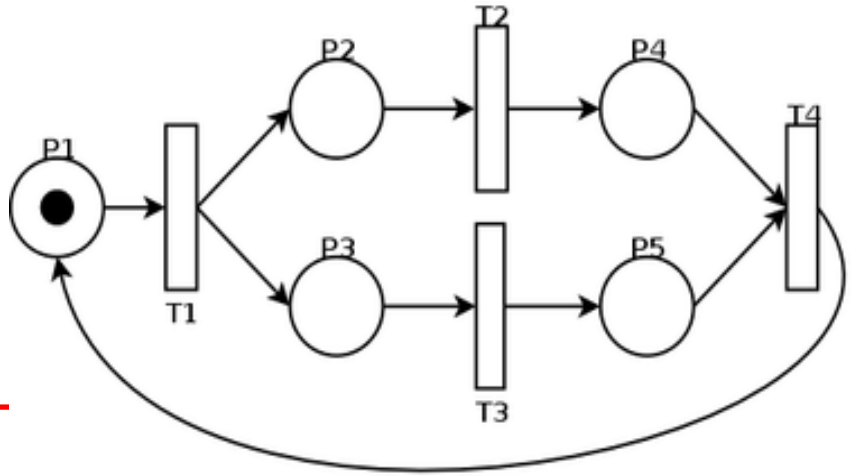
Marked Graphs & Properties

- The token count in an elementary loop of marked graph does not change as a result of transition firings.
- A marked graph is strongly connected if there is a path from any node to any other node.
- For a strongly connected marked graph, a firing sequence leads back to m_0 iff it fires every transition for an equal number of times.



Performance Evaluation

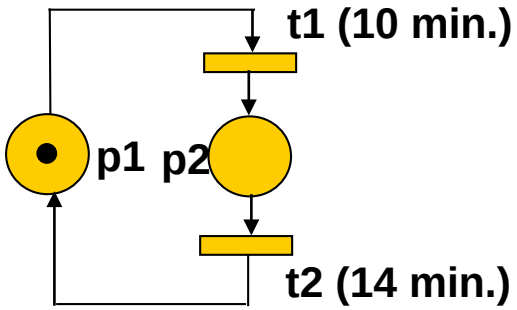
- Throughput is the inverse of the cycle time. The cycle time is defined as the largest cycle time in its marked graph representation. The cycle time of each cycle is the sum of the delays of all associated transitions (or places) divided by number of the tokens that can reside in the cycle.
- A cycle is one that contains no repeated nodes except the beginning and ending oness.



Cycle Time in Timed Marked Graph

Loop cycle time and its evaluation:

Transitions-have-delay-only case:



If $m_0=(1\ 0)^\tau$, the cycle time is: 24 min.

If $m_0=(2\ 0)^\tau$, the cycle time is: 12 min.

If $m_0=(1\ 1)^\tau$, the cycle time is: 12 min.

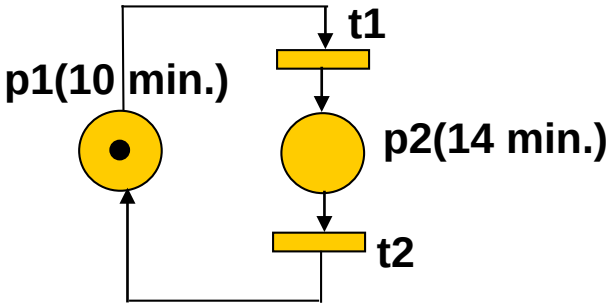
If $m_0=(3\ 0)^\tau$, the cycle time is: 8 min.

…

$\pi=[d(t1)+d(t2)]/[m_0(p1)+m_0(p2)]$
i.e., the cycle time is the sum of delays divided by the sum of token in a loop

Cycle Time in Timed Marked Graph

Loop cycle time and its evaluation:



Places-have-delay-only case:

If $m_0=(1\ 0)^\tau$, the cycle time is: 24 min.

If $m_0=(2\ 0)^\tau$, the cycle time is: 12 min.

If $m_0=(1\ 1)^\tau$, the cycle time is: 12 min.

If $m_0=(3\ 0)^\tau$, the cycle time is: 8 min.

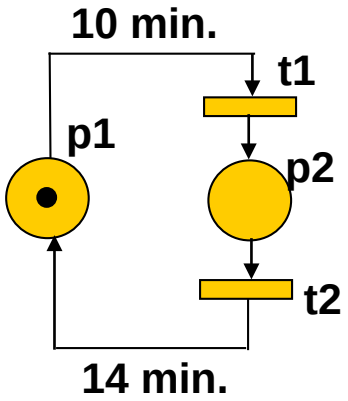
…

$\pi=[d(p1)+d(p2)]/[m_0(p1)+m_0(p2)]$
i.e., the cycle time is the sum of delays divided by the sum of token in a loop

Cycle Time in Timed Marked Graph

Loop cycle time and its evaluation:

Arcs-have-delay-only case:



If $m_0=(1\ 0)^\tau$, the cycle time is: 24 min.

If $m_0=(2\ 0)^\tau$, the cycle time is: 12 min.

If $m_0=(1\ 1)^\tau$, the cycle time is: 12 min.

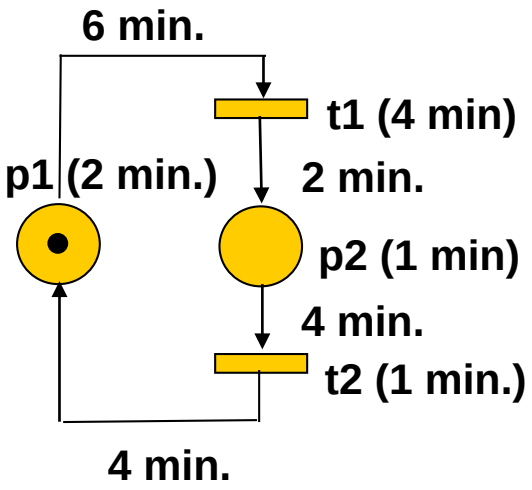
If $m_0=(3\ 0)^\tau$, the cycle time is: 8 min.

…

$\pi=[d(p1,t1)+d(t2,p1)]/[m_0(p1)+m_0(p2)]$
i.e., the cycle time is the sum of delays divided by the sum of token in a loop

Cycle Time in Timed Marked Graph

Loop cycle time and its evaluation:



General case:

If $m_0=(1\ 0)^\tau$,

Loop delay=2+6+4+2+1+4+1+4=24 min

Token count in the loop = 1

The cycle time is 24 Min.

If $m_0=(2\ 0)^\tau$, Token count =2 and thus the cycle time is: 12 min.

π=Loop delay/Token count in the loop

π = D/N

Cycle Time in Timed Marked Graph

- The cycle time (i.e., the time from any beginning state and back to itself) is found as:

$$\pi = \mathbf{Max}\{D_i/N_i\}$$

where D_i is the sum of all delays in the i-th loop and N_i is the total number of tokens in it.

- D_i is called loop delay and N_i called the token count of the i-th loop.
- If $D_j/N_j=\pi$, the j-th loop is called a **bottleneck**.

Computing Cycle Time

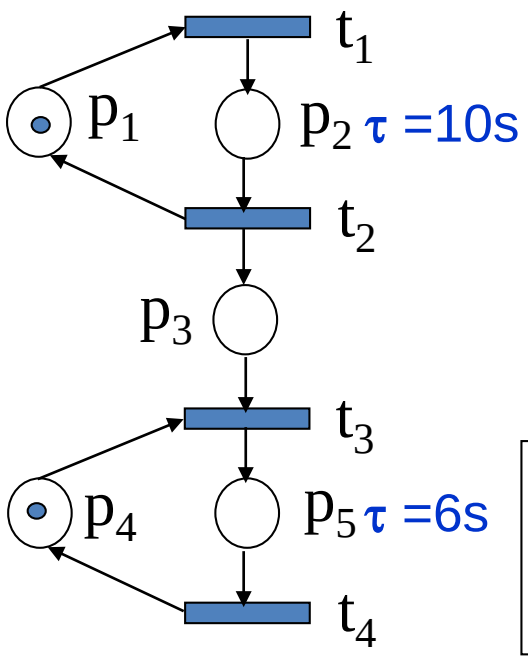
- Enumerate all loops
- Find loop delays, token count and cycle time for each loop.
- Apply to find the system cycle time

$$\pi = \text{Max}\{D_i/N_i\}$$

- Identify all bottleneck loops.

Example: Producer/Consumer

Loops	Total time delay	Token sum	Cycle time
*p*1*t*1 *p*2 *t*2 *p*1	10	1	**10**
*p*4 *t*3 *p*5 *t*4 *p*4	6	1	**6**

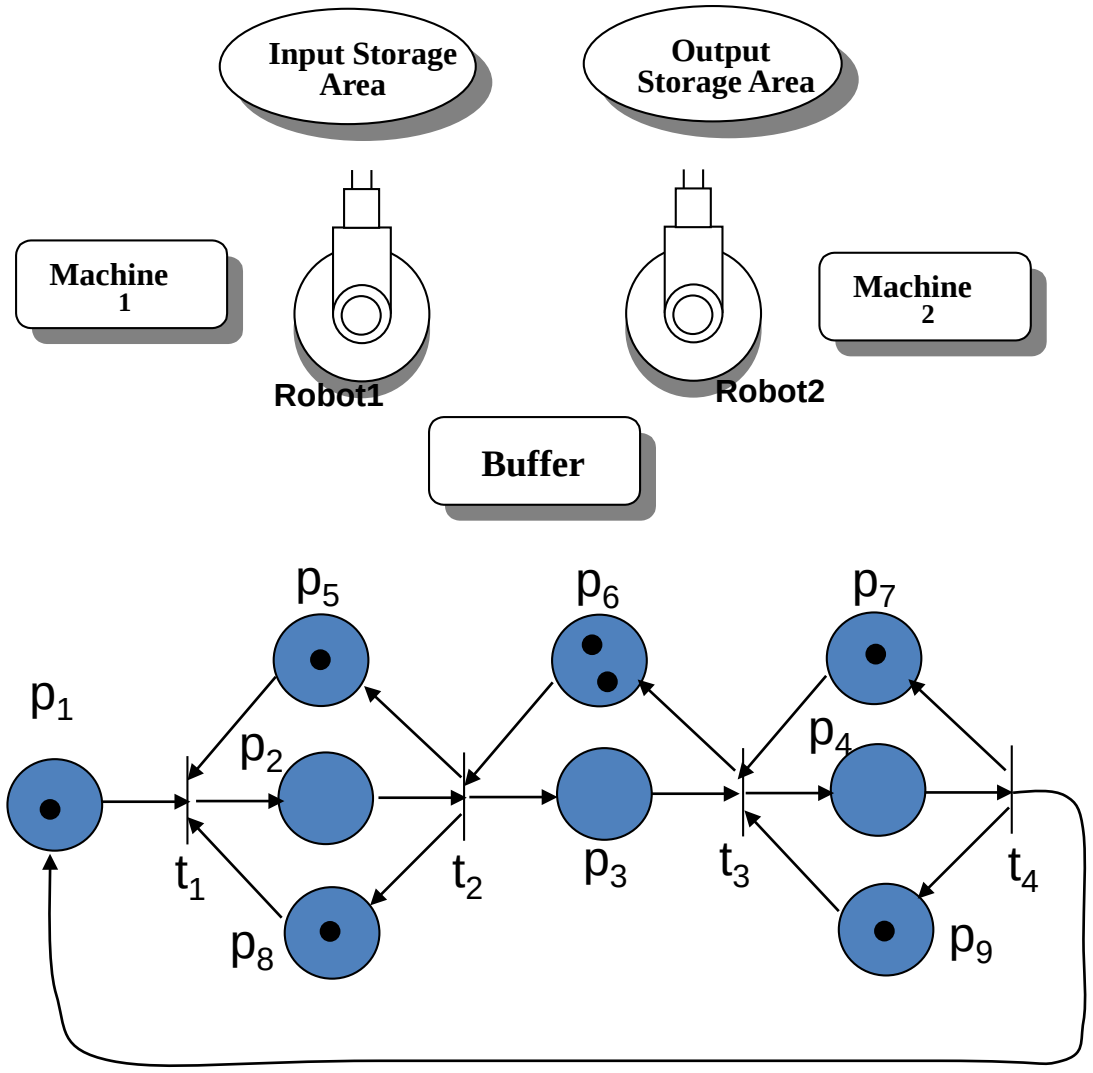


Producing process is slower than consuming process. We may want to add more producer to satisfy the consumer. If we add one more producer (two tokens in p1):

Loops	**Delay**	**Token sum**	**Cycle time**
***p*1*t*1 *p*2 *t*2 *p*1**	**10**	**2**	**5**
***p*4 *t*3 *p*5 *t*4 *p*4**	**6**	**1**	**6**

Will it help if more producers are added?

Example: Manufacturing cell



p1: Raw part available

p2: M1 processing a part (**10**)

p3: Intermediate parts ready

p4: M2 processing a part (**16**)

p5: M1 available

p6: buffer slots available

p7: M2 available

p8: R1 available

p9: R2 available

t1: R1 loading (**1**)

t2: R1 unloading (**1**)

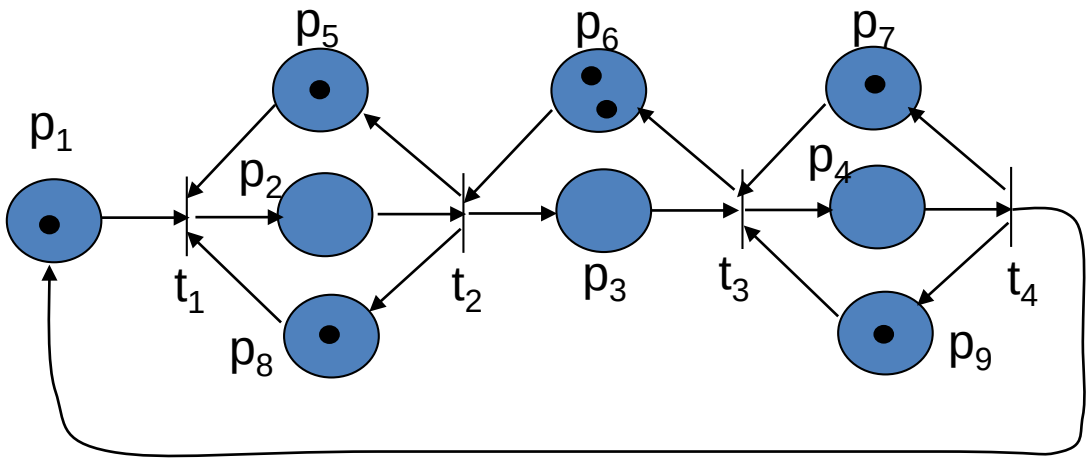
t3: R2 loading (**1**)

t4: R2 unloading (**1**)

(t4,p1): load a raw part to input storage area (**2**)

Example: Manufacturing cell

	T1	p2	t2	t3	p4	t4	(t4,p1)
Time delay	1	10	1	1	16	1	2



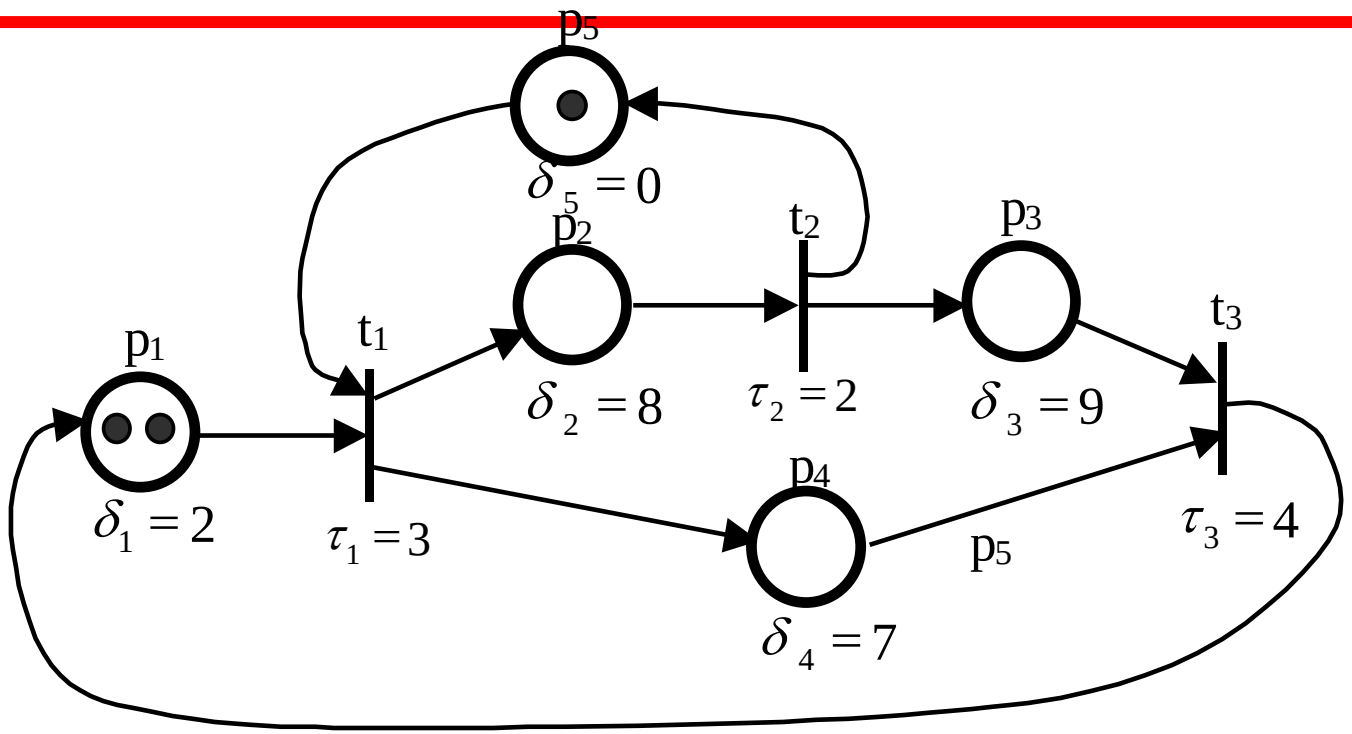
Loops	Di	Ni	Di/Ni
$p_5t_1p_2t_2p_5$	12	1	12
$p_6t_2p_3t_3p_6$	2	2	1
$p_7t_3p_4t_4p_7$	18	1	18
$p_1t_1p_2t_2p_3t_3p_4t_4p_1$	32	1	**32**
$p_8t_1p_2t_2p_8$	12	1	12
$p_9t_3p_4t_4p_9$	18	1	18

System cycle time:32

Adding one more token to p1 (load one more raw part to input storage area) will lower 32 to 16.

Will loading more raw parts to input area storage area shorten the cycle time?

Example



Loops	Total time delay	Token sum	Cycle time
*p*1*t*1 *p*2 *t*2 *p*3 *t*3 *p*1	28	2	14
*p*1 *t*1 *p*4 *t*3 *p*1	16	2	8
*p*5 *t*1 *p*2 *t*2 *p*5	13	1	13

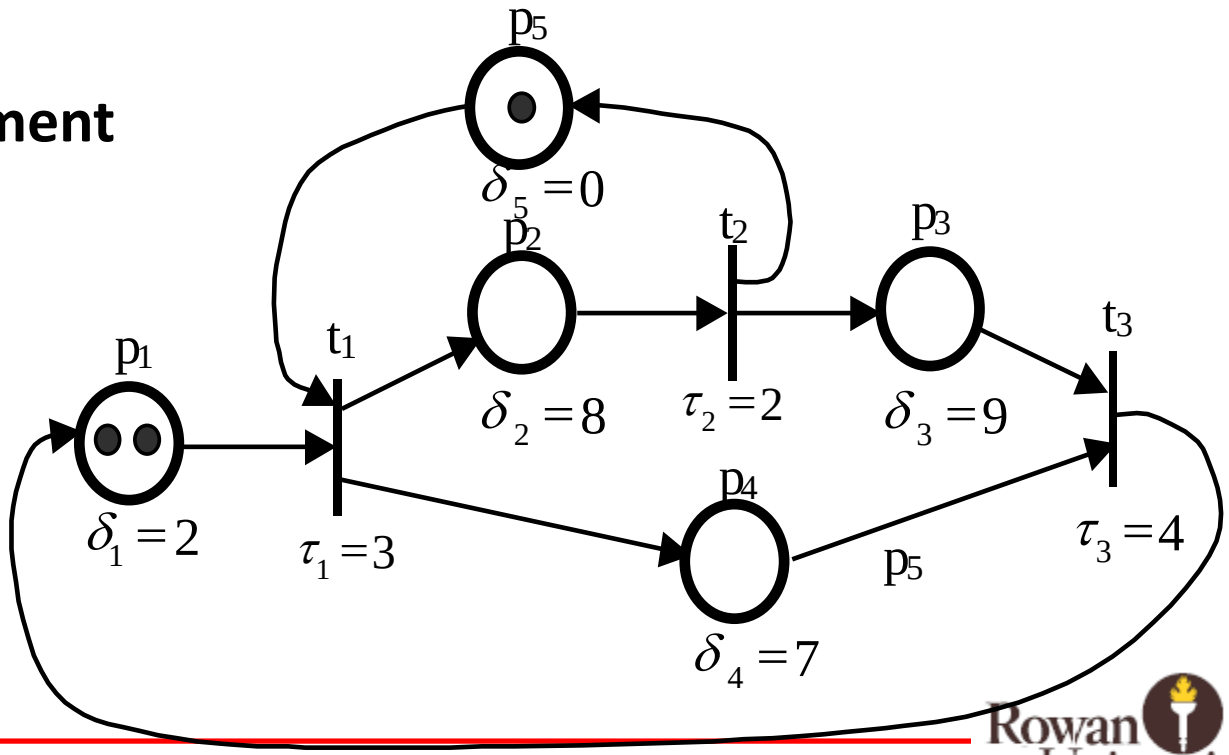
System cycle time is 14 time units.

Example (Continued)

Loops	Total time delay	Token sum	Cycle time
*p*1*t*1 *p*2 *t*2 *p*3 *t*3 *p*1	28	2	14
*p*1 *t*1 *p*4 *t*3 *p*1	16	2	8
*p*5 *t*1 *p*2 *t*2 *p*5	13	1	13

If you are given the opportunity to reduce one and only one activity's delay into half, which activity would you choose?

What is the improvement percentage?



Example (Continued)

Loops	Total time delay	Token sum	Cycle time
*p*1*t*1 *p*2 *t*2 *p*3 *t*3 *p*1	28	2	14
*p*1 *t*1 *p*4 *t*3 *p*1	16	2	8
*p*5 *t*1 *p*2 *t*2 *p*5	13	1	13

If you are given the opportunity to reduce one and only one activity's delay into half, which activity would you choose?

Solution: p2

What is the improvement percentage?

Solution:
(14-12)/14*100%