Project 1

CIS 4930/CDA 5636 (Embedded Systems): Fall 2012

Due Date: Oct 03, 2012 11:55 PM **(EDGE:** Oct 06, 2012 11:55 PM**)**

You are not allowed to take or give help in completing this project. Please submit only one source file (.c, .cpp or .java) that contains everything. On top of that source file (as comment), please provide the command to compile your file (e.g., /* gcc myfile.c –o PNsim */) in the Linux environment. Please also include the sentence in quote on top of the source file: "/* **On my honor, I have neither given nor received unauthorized aid on this assignment */**". The TA will compile it in thunder.cise.ufl.edu. Please make sure you compile and test your framework in a Linux environment only.

For this project you will create a simulator for a **Petri net**. Your simulator should be capable of constructing a Petri net based on the input specification, and generate the step-by-step simulation of the Petri net. Your Petri net simulator (with executable name as **PNsim**) should provide the following option to users:

```
./PNsim   inputfilename  outputfilename
```

Where,

* Inputfilename – The input filename of Petri-net specification (e.g., input.txt).
* Outputfilename – The file name for printing the simulation output (e.g., output.txt).

You are strongly recommended to test your submission. Download your submission from eLearning, use a LINUX machine (e.g., thunder.cise.ufl.edu or some other Linux machine in your place), and test it with the following commands.

```
> gcc  PNsim.c  -o PNsim   (or,  javac PNsim.java )
> ./PNsim   input.txt   output.txt  ( or,   java  PNsim   input.txt   output.txt  )
```

Please notice that the TA will use "diff –w –B" (ignores blanks etc.) to grade your simulation output with the provided output file. Any difference will be regarded as incorrect output. Please make sure your output exactly follows the sample output. You can check difference using the following command line in linux:

```
> diff –w  -B your_output.txt   sample_output.txt
```

Petri net Specification (Input file format)

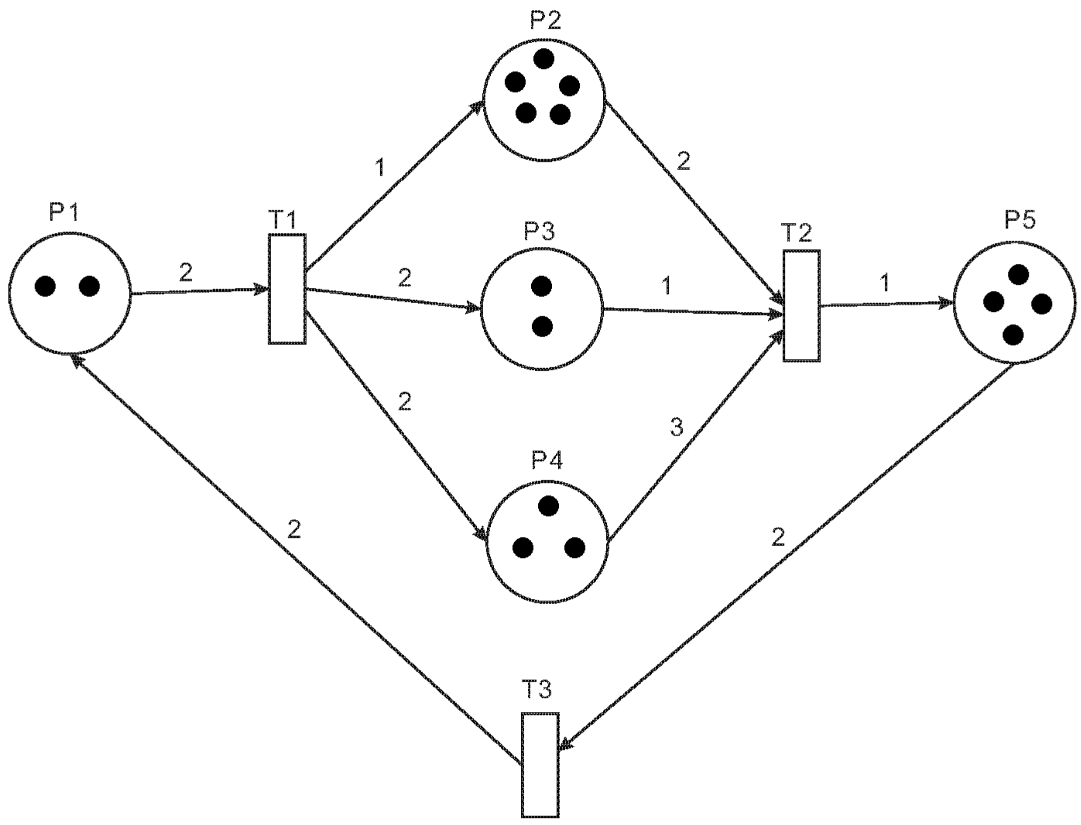
In this project, we assume that the execution of the Petri net is synchronized by a single step counter. Initially the counter is 0 (initial state), although in real Petri net, there is no synchronization scheme. The counter value indicates a **simulation step**. In each step, all the enabled transitions (the number of tokens satisfies the condition) are fired. The updated token numbers are available only at the next step. Please note that all enabled (valid) transitions are fired once and only once in every step.

The input file format is as follows. Please see the example below (input.txt) if you have any problem in understanding this format. The input file consists of four lines (no spaces, tabs or blanks). The first line indicates the maximum number of *simulation steps* to execute. The second line describes all the *places* i.e., the name of the *places* and their initial states (the number of tokens in each circle). The third line indicates the *transitions*. Each *transition* has three fields. The first field is the *transition* name. The second and third fields contain the names of the incoming and outgoing *arcs*, respectively. The fourth line describes various *arcs* as a pair of *arc* name and associated value (the number of tokens consumed or produced when traversing that *arc*).

```
SimulationSteps:[number]
Places:(PlaceName,numTokens), …
Transitions:(TransitionName,<incomingArcName, …>,<outgoingArcName, …>), …
Arcs:(ArcName,ArcValue), …
```

The following is an example specification (**input.txt**) of the Petri net shown in Figure 1 that has five places (P1, P2, P3, P4 and P5), three transitions (T1, T2 and T3) and ten arcs. Here, (P1, 2) means the place P1 has initially 2 tokens. Similarly, (T1, <P1-T1>, <T1-P2, T1-P3, T1-P4>) means the transition T1 has one incoming arc (P1-T1) and three outgoing arcs (T1-P2, T1-P3, T1-P4). Likewise, (P1-T1, 2) means that the incoming arc P1-T1 (starts at the place P1 and ends at transition T1) consumes (needs) two tokens to enable the transition T1. Similarly, (T1-P2, 1) means the outgoing arc T1-P2 (starts at the transition T1 and ends at place P2) produces one token when the transition T1 is enabled.

```
TargetStep:[10]
Places:(P1,2),(P2,5),(P3,2),(P4,3),(P5,4)
Transitions:(T1,<P1-T1>,<T1-P2,T1-P3,T1-P4>),(T2,<P2-T2,P3-T2,P4-T2>,<T2-P5>),(T3,<P5-T3>,<T3-P1>)
Arcs:(P1-T1,2),(T1-P2,1),(T1-P3,2),(T1-P4,2),(P2-T2,2),(P3-T2,1),(P4-T2,3),(T2-P5,1),(P5-T3,2),(T3-P1,2)
```



Output file format

The output file should be formatted as follows:

Step 0:(PlaceName, numTokens), …
……………………………………..
< continues until the required simulation step is reached or there are no enabled transitions >

The following should be the content of the simulation output (**output.txt**), when the above example input file (**input.txt**) is used.

```
Step 0:(P1,2),(P2,5),(P3,2),(P4,3),(P5,4)

Step 1:(P1,2),(P2,4),(P3,3),(P4,2),(P5,3)

Step 2:(P1,2),(P2,5),(P3,5),(P4,4),(P5,1)

Step 3:(P1,0),(P2,4),(P3,6),(P4,3),(P5,2)

Step 4:(P1,2),(P2,2),(P3,5),(P4,0),(P5,1)

Step 5:(P1,0),(P2,3),(P3,7),(P4,2),(P5,1)
```

Every line shows the simulation step number and the Petri net state (places and the number of tokens in each place) at the beginning of that step. There are two ways simulation ends:

- When the "simulation step" is reached, the simulation ends. In other words, print the (PlaceName, #ofTokens) information up to (including) the simulation step.
- The simulation also ends when there are no enabled transitions in the system. In the above example, although the simulation step is 10, there are no enabled transitions after step 5. Therefore, the values of the places are printed up to step 5.

The assignments webpage has the same sample input and output files. Correct handling of the sample input (with possible changes of number of tokens and arc values) will be used to determine 60% of credit awarded. The remaining 40% will be determined from other input test cases that you will not have access prior to grading. The other test cases can have different number of places, transitions and connectivity as well as different number of tokens and arc values. It is recommended that you construct your own sample input files with which to further test your simulator. In our new test cases, the number of each of the items (places, transitions, arcs) would be less than 15. You can also assume that the number of tokens in each place will be less than 10. Similarly, the possible arc values will be also less than 10.