Mathematical Problems of Computer Science 38, 84–86, 2012.

Modeling FIFO Queues by Dynamic Petri Nets

Gohar R. Petrosyan

Armenian State Pedagogical University after Kh. Abovyan
E-mail: blueayez777@mail.ru

Petri Nets (PN) are a graphical tool for formal description of the flow of activities in complex systems. Compared to other more popular techniques of graphical system representation (for instance, block diagrams or logical trees), PN are particularly matched for representation of logical interactions among parts or activities in a system in a natural way. Typical situations that can be modeled by PN are: synchronization, sequentiality, concurrency and conflict [1], [2], [3], [5].

In computer science, a queue is a particular kind of abstract data type or collection in which the entities in the collection are kept in an order and the principal operations(or the only one) in the collection are (is) the addition of entities to the rear terminal position and removal of entities from the front terminal position. This makes the queue a First-In-First-Out (FIFO) data structure. In a FIFO data structure, the first element added to the queue will be the first one to be removed. This is equivalent to the requirement that once an element is added, all elements that were added before have to be removed before addition of the new element. A queue is an example of a linear data structure [5].

Definition. A Petri Net is $M = (C, \mu)$, where $C = (P, T, I, O)$ is the network structure, and μ is the network condition. The P-positions, T-transitions are finite sets; $I : T \to P\infty$, $O : T \to P\infty$ are the input and output functions, respectively, where $P\infty$ are all possible collections (repetitive elements) of P; $\mu : P \to N_0$ is the function of conditions, where $N_0 = \{0, 1, \ldots\}$ is the set of integers. We determine (in a known manner) the allowed transitions of Petri Nets and the transitions from one state to another, as well the set of reachable states.

The state of the network is represented by the following vector:

$$(\mu(P_1), \mu(P_2), ..., \mu(P_m)), \quad if \quad P = \{P_1, P_2, ..., P_m\},$$

μ is a function, which carries out the following mapping, $\mu : P \to N_0$, where N_0 is used to encode the number of tokens in the positions. Before starting its work the net must have an initial state, $(\mu_0(P_1), \mu_0(P_2), ..., \mu_0(P_m))$, where the number of tokens in their respective positions are shown [1, 4].

The complexity of the standard Petri Net modeling of the the FIFO queue is $4m+6(m-1)+2+2 = 10m-2$ (the number of places, transitions, arcs, for a network with m elements are calculated).

The difference between dynamic Petri Nets from the standard Petri Nets is that the structure of the first changes during its work, that is, positions or transitions can be added or removed from the net, besides, and , consequently, the input and output functions are changed [4].

Below is the mathematical definition of dynamic Petri Nets, where we have introduced idea of dynamic memory. This idea helps us to use the freed memory space.

First, we denote the sequence of all places as follows: $\Omega = (P_0, P_1, P_2, ..., P_k, ..., P_{2^k})$, and sequence of all transitions by the following

$= \{t_0, t_1, ..., t_k, ..., t_{2^k}\}$.

Let's define the following, $C = (P, T, I, O)$, as the current structure of the network, ere $P \subseteq \Omega$ is the finite set of positions at the given moment, and $T \subseteq \psi$ is the finite set of nsitions at the given moment, and $I : T \to P\infty$ and $O : T \to P\infty$, are the current input l output functions, respectively. $P\infty$ has the same meaning as in the previous cases.

Denote $\mu : \Omega \to N_0^{-1}$, as the function of condition, where N_0^{-1} is the set of integers (the mbers of tokens in the positions), including (-1), that encodes the unused positions.

Let $P = \{P_{i1}, P_{i2}, ..., P_{ir}\}$ is given, $(\mu(P_{i1}), \mu(P_{i2}), ..., \mu(P_{ir}))$ be the vector of the current te at the given moment.

Moreover, if $P = \{P_{i1}, P_{i2}, ..., P_{ir}\}$ is the number of positions in the network, then the nction μ performs the following mapping:

$$\mu : P \to N_o, \mu : (\Omega / P) \to \{-1\}.$$

o observe the processes in the network, you first need to know the laws of the structural nges.

Suppose we have an $M = (C, \mu)$ dynamic Petri Net, which currently has $C = (P, T, I, O)$ H is in the μ current condition.

Let's say that $t \in T$ transition is allowed, if $\forall P \in I(t)$, $\mu(P) \geq \sharp(P, I(t))$, where $\sharp(x, A)$ s the same meaning as in the standard Petri networks [1, 4].

Let $M = (C, \mu)$ be a dynamic network and the transition $t \in T$ is allowed to run.

In this case, as we have noted, not only the state of the network will change, but also structure. In contrast to the previous definition, we define the function δ [1, 4], which pends on the three arguments (C, μ, t), and returns back the new structure and state of network after the transition t as a new value. In fact we conclude that, in optimization se, the Petri Dynamic Nets are more convenient for modeling of systems of some type n the standard Petri Nets.

In fact, $\delta(C, \mu, t) = (C', \mu')$, where $C = (P', T', I', O')$ is a new structure.

is the sequence of the new positions,

is the sequence of the new transitions,

$T' \to (P')\infty$ is the new input function,

$: T' \to (P')\infty$ is the new output function.

$= (\mu'(P_0), \mu'(P_1), ..., \mu'(P_k), ..., \mu'(P_{2^k}))$ and $\mu' : (\Omega / P') \to \{-1\}$.

The initial structure of the network is denoted as C_0, and the initial condition is μ_0. The work generates (C, μ) pairs of sets, and this set is denoted by $R(C_0, \mu_0)$, where

$(C_0, \mu_0) \in R(C_0, \mu_0)$ is initial state,

if $(C', \mu') \in R(C_0, \mu_0)$ and $\exists t \in T'$, such that $\delta(C', \mu', t) = (C', \mu') \in R(C_0, \mu_0)$,

The other (C, μ) pairs in $R(C_0, \mu_0)$ do not belong to $R(C_0, \mu_0)$ and the latter can be inite.

References

[1] J. L. Peterson., *Petri net theory and the modeling of systems.* Prentice Hall, Englewood Cliffs, 1981.

[2] T. Murata, "Petri Nets: Properties, Analysis and Applications", *Proceedings of the IEEE*, vol. 77, no. 4, 1989.

[3] W. Reising, G. Rozenberg (eds). Lecture Notes on Petri Nets. Parts I and II // Lecture Notes in Computer Sciences. V. 1491 - 1492. Springer - Verlag, 1998.

[4] В. Е. Котов, Сети Петри. М.: Наука, 1984.

[5] Д. Кнут, Искусство программирования, Т1-Т3, 2008.