13th Int. Workshop on Generalized Nets
London, 29 October 2012, 69–76

Verification of a token transfer algorithm in a .NET implementation of generalized nets

Polya Gocheva and Valeri Gochev

College of Telecommunications and Post
1 Acad. Stefan Mladenov Str., Sofia - 1700, Bulgaria
e-mails: {polya_gocheva, valeri_gochev}@hctp.acad.bg

Abstract: The present paper aims to consider verification of .NET implementation of generalized nets' functioning. Simple models are used in order influence of various generalized nets' components to be shown. Graphical structure of transitions and places including their priority and capacity is used. Number of tokens and token transfer are denoted in original manner according to authors' book.
Keywords: Generalized nets, Verification, .NET.
AMS Classification: 68Q85.

1 Introduction

The advances in generalized nets' theory contribute to creating of complex models as in [5, 6]. The related algorithms are complex, too, and many applications require formal verification. Recent research in this direction has been done for object-oriented and procedural programs, [7, 8, 9].

There are about 20 software products which handle generalized nets [3, 4]. The .NET implementation called GoGN 1.0 is developed by the authors. The source code of GoGN 1.0 is presented in [4]. It includes definitions related to an algorithm of generalized nets' functioning, but this algorithm is not fully considered. Examples on it based on simple generalized nets (actually almost all consisting of one transition only) are given below.

2 Short remarks on the .NET implementation GoGN 1.0

GoGN 1.0 represents a .NET application, which implements one generalized net using programming language C#, [4]. The developed software contains classes and structures, which implement all components of generalized nets in the context of object-oriented programming. The names of the upper classes are as follows: *Token*, *Place*, *Transition*. In case of whole generalized nets, the class *GN* is used. The last one is an abstract class, and some of its methods must be defined in inherited classes in order specific features of generalized nets to be

modeled. A source code of a token transfer algorithm is presented in class *GN*. In the present paper the considered classes are used, but new generalized nets are implemented.

A graphical user interface (GIU) is described in GoGN 1.0, too. The Windows form *FormGN* defines controls and event handlers. It includes drawing the generalized net. In all figures below such drawing is used. Priority and capacity of transitions and places, number of tokens and token transfer are denoted in original manner.

3 Token transfer algorithms in generalized nets

The algorithms in generalized nets are more complex than the corresponding ones in Petri nets [1, 2, 4, 7, 10]. A new term "abstract transition" is defined in the generalized nets' theory [1, 2, 4]. Such a transition consists of all active transitions of a generalized net in a given time-moment, and it includes components which are computed according to these (ordinary) transitions. For example, the input places in all active transitions are represented in an abstract transition as input places with unchanged capacity and priority. In [4] the implementation of an abstract transition is based on adding and removing of transitions.

Each token transfer in a generalized net (through an abstract transition respectively) depends (in this order) on:

- priority of input places;
- priority of output places;
- priority of transitions;
- priority of tokens.

The capacities of places and arcs, time components, transitions' conditions and other factors influence, too. Details on a classical token transfer algorithm are presented in next section.

4 Examples

Twelve examples are given below. Models concern important features of a well-known transfer algorithm [2, 4]. The first eight GNs (GN 01 – GN 08) have the graphical structure given on Figure 1. It consists of one transition only, and 1000 tokens enter the input place at the initial time moment. The time step of these models is equal to 1, and duration of their functioning is equal to 1000. All possible token transfers are performed. GoGN 1.0 denotes a great number of tokens in a given place by a big (filled on Figure 1) circle and a mark ("1000" in this case) over the place.

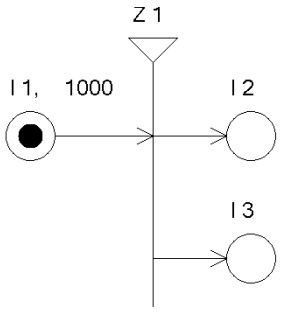
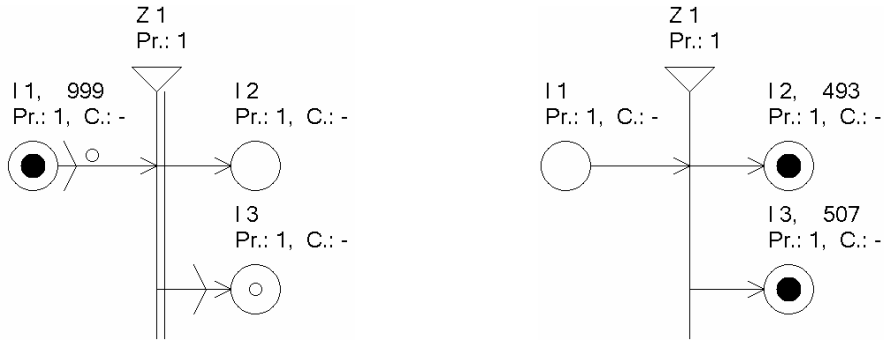


Figure 1: Graphical structure of generalized nets GN 01 – GN 08 at their initial time moment

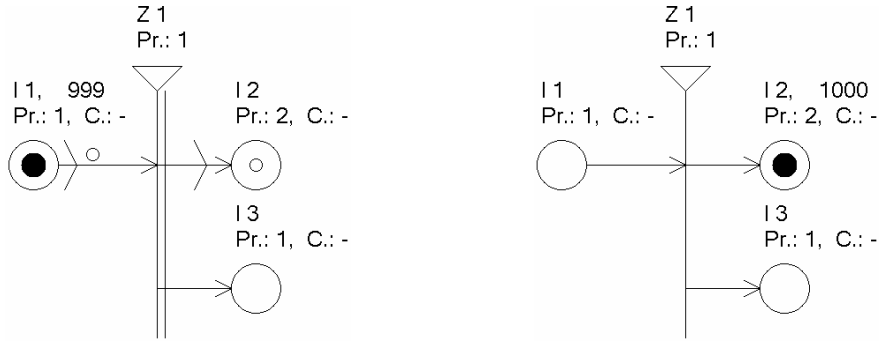
GN 01: Output places with equal priority

In this generalized net (Figure 2), the output places $l2$ and $l3$ have equal priority. Priority is shown over the transitions and the places next to mark "Pr.:"; in similar way capacity of places can be given next to mark "C.:" (in *GN 01* the capacity is infinity), [4]. An active transition and one token transfer are denoted on Figure 2 a). In the present paper, all active transitions are drawn with double lines, and all tokens transferring through these transitions are denoted by complementary arrows and tokens, [4]. An arrow and a token on a given transition's input arc and an arrow on a given transition's output arc (or two or more arrows on a different transition's output arcs) mark a transfer of a token to one or more output places. In case of the considered generalized net, a transfer from place $l1$ to place $l3$ is shown on Figure 2 a); the token which just transferred entering $l3$ is not filled. In *GN 01* tokens do not split and enter output places according to pseudo-random process because of an absence of predicates in the transition's condition. The last one contains value "true" only. At the end of the generalized net's functioning the tokens are divided into almost equal parts (Figure 2 b)).



a) b)

Figure 2: *GN 01:* Output places with equal priority:
a) the first token transfer; b) the end of the generalized net's functioning



a) b)

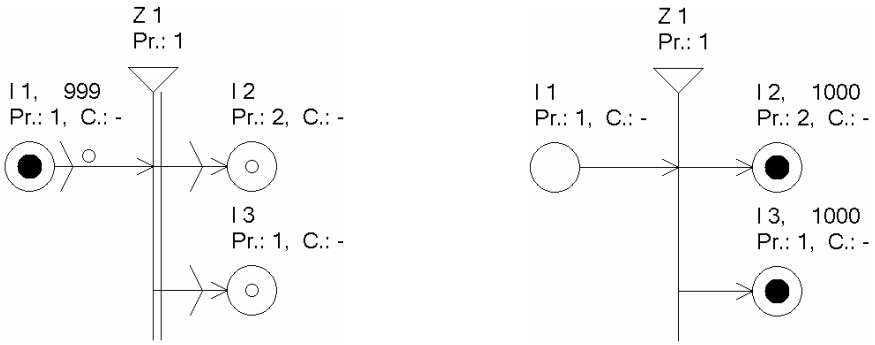
Figure 3: GN 02: Output places with different priority:
a) the first token transfer; b) the end of the generalized net's functioning

GN 02: Output places with different priority

Here the output places $l2$ and $l3$ have different priority (Figure 3). Because of bigger higher priority of $l2$, at each time moment a token enters this place.

GN 03: Splitting tokens

In this model (Figure 4), all tokens can split. Each token transferring through the transition splits on two tokens, which enter *l*2 and *l*3 simultaneously regardless of the priority of these places.

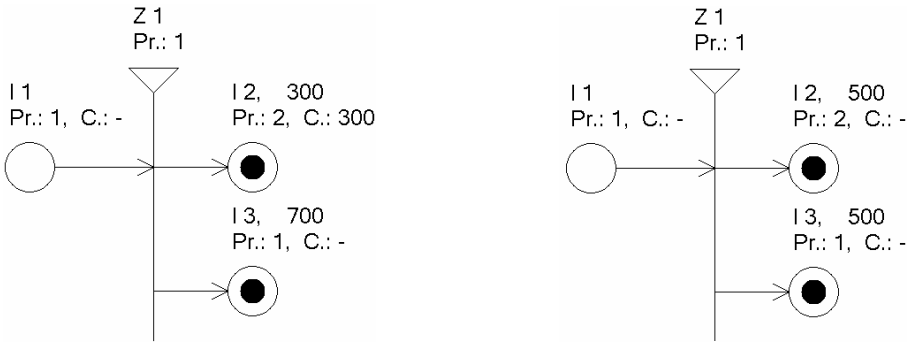


a) b)

Figure 4: GN 03: Splitting tokens:
a) the first token transfer; b) the end of the generalized net's functioning

GN 04: Place capacity

Here (Figure 5 a)), the place *l2* has capacity equal to 300, and it can hold 300 tokens maximum. Since the priority of *l*2 is bigger than the priority of *l*3, 300 tokens enter *l*2, and then the rest 700 enter *l*3.



a) b)

Figure 5: End of a generalized net's functioning:
a) GN 04: Place capacity; b) GN 05: A predicate concerning number of tokens

GN 05: A predicate concerning number of tokens

The number of tokens in output places can be caused by predicates. Let the condition of the transition given in Figure 5 b) be

$$\begin{array}{c|cc} & l2 & l3 \\ \hline l1 & w_{1,2} & true \end{array},$$

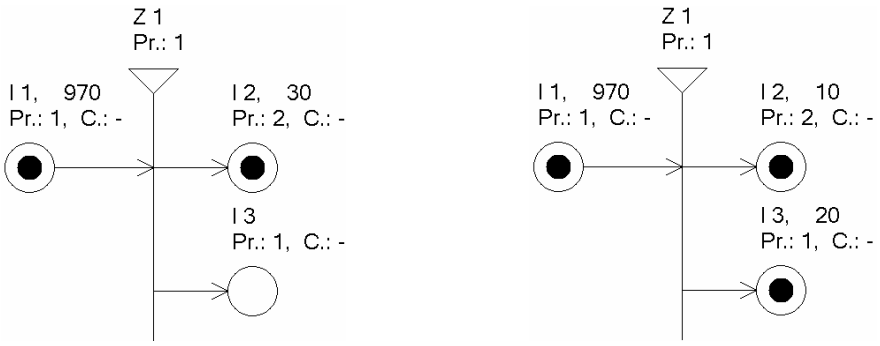
where the predicate in the index matrix has value: w_{12} = "The number of tokens in *l*3 is greater than the number of tokens in *l*2". In this case, the tokens are divided into two equal parts.

GN 06: Time components

In the upper models time components do not restrict the token transfer. However, let time components allow a single activation of the transition, and let the duration of this activation be equal to 30. It is obvious that 30 tokens maximum can be transferred (Figure 6 a)).

GN 07: Arc capacity

Let the generalized net *GN 07* be modified in order to limit the number of tokens transferring from *l*1 to *l*2 at one transition's activation. It can be done by adding arc capacity to the arc binding these places. In the model presented in Figure 6 b) this capacity is equal to 10.

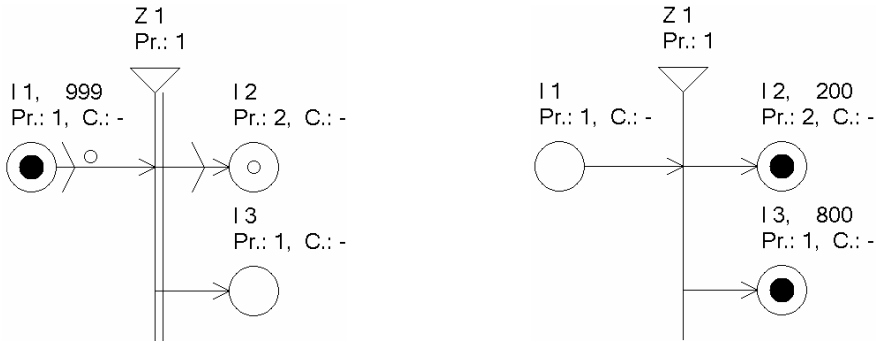


a) b)

Figure 6: End of a generalized net's functioning:
a) GN 06: Time components; b) GN 07: Arc capacity

GN 08: Tokens' characteristics

Token transfers can be conditioned by characteristics of tokens too. For example, let the initial characteristics of 200 tokens be the same, and let the initial characteristics of 800 tokens be others. Let, moreover, the predicates in the transition's condition allow token from the first group only to be able to transfer to *l2* (Figure 7). The example demonstrates a way of handling alternatives.



a) b)

Figure 7: GN 08: Tokens' characteristics:
a) the first token transfer; b) the end of the generalized nets' functioning

GN 09: Transition's type

This model has different graphical structure. The transition has two input places (Figure 8). Generally, the transfer of a given token can be forbidden because of the transition's type. The last one represents a Boolean expression indicating which input places must be non-empty in order token transfer to be able. In *GN 09* the transition's type has the form $\Delta = \wedge(l1, l4)$. That means that *l1* and *l4* must be non-empty in order any token to be transferred. On Figure 8 a) 1000 tokens in *l1* and 900 tokens in *l4* are shown. Tokens from *l1* and *l4* transfer for 900 time steps to *l2* and *l3*, respectively, according to the token's condition

	l2	*l3*
l1	*true*	*false*
l4	*false*	*true*

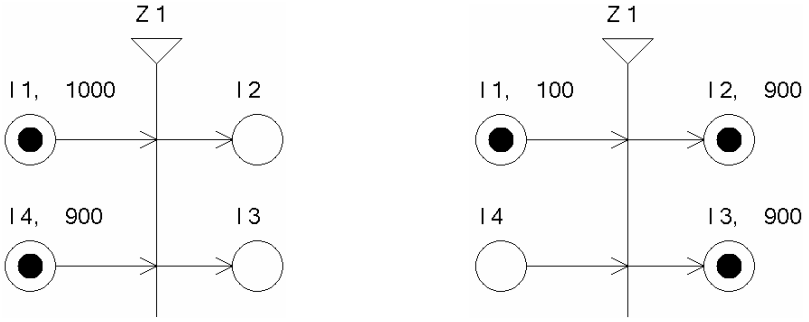


Figure 8: GN 09: Transition's type:
a) at the initial time moment; b) the end of the generalized nets' functioning

GN 10: Transitions with equal priority

This generalized net, presented in Figure 9 a), contains two transitions. Since the transitions as well as places have equal priority, abstract transitions are constructed in pseudo-random manner. The condition of *Z2* has the form

	l3
l2	$w_{2,3}$

,

where the predicate in the index matrix is: $w_{2,3}$ = "There is a single token in *l2*". The last predicate does not allow next transfer through *Z2* in case of two or more tokens in *l2*.

GN 11: Transitions with different priority

GN 11 (Figure 9 b)) is much the same as *GN 10*. The only difference is the priority of *Z2*. This priority orders the input places in abstract transitions synchronizing the token transfer.

GN 12: Priority of input places

GN 12 (Figure 9 c)) is much the same as *GN 10* too. In this case the only difference is the priority of *l2*. This priority also orders the input places in abstract transitions.

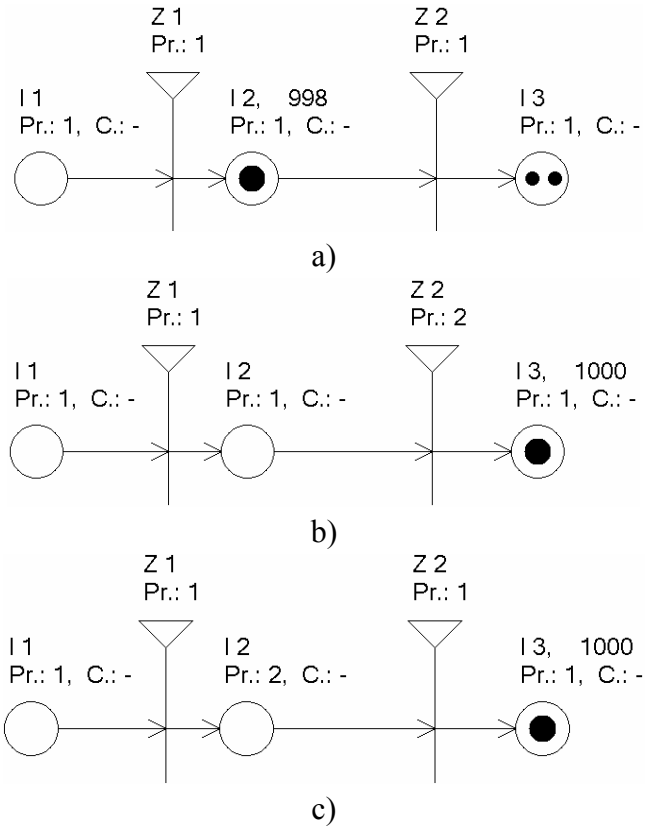


Figure 9: End of generalized nets' functioning: a) GN 10: Transitions with equal priority; b) GN 11: Transitions with different priority; c) GN 12: Priority of input places

5 Conclusion

The given examples verify the token transfer algorithm introduced in [4]. The results strictly conform to the theoretical notes in the present paper. The priority of transitions and places is taken into account in the considered .NET implementation GoGN 1.0. This priority is well defined in the source code of GoGN 1.0, and can be viewed in graphical objects. The influence of other components of generalized nets is shown too. For example, capacities of places and arcs limit the number of token transfers. Time components can be used in a proper way according to well-known restrictions.

References

[1] Atanassov, K., *Generalized Nets*. World Scientific, Singapore, 1991.

[2] Atanassov, K., *On Generalized Nets Theory*. 'Prof. Marin Drinov' Academic Publishing House, Sofia, 2007.

[3] Dimitrov, D., Software products implementing generalized nets. *Annual of Section "Informatics", Union of Scientists in Bulgaria*, Vol. 3, 2010, 37–50 (In Bulgarian).

[4] Gochev, V., P. Gocheva, *.NET implementation of generalized nets. Object-oriented approach.* College of Telecommunications and Post, Sofia, 2012 (In Bulgarian).

[5] Poryazov, S., Atanassov K. A Generalized Net Subscribers' Traffic Model of Communication Switching Systems. *Journal on Advances in Modelling&Analysis*, Vol. 37, 1997, No. 1–2, 27–35.

[6] Poryazov, S. A., Saranova E. T. Chapter 24: Some General Terminal and Network Teletraffic Equations in Virtual Circuit Switching Systems –In: *Modeling and Simulation Tools for Emerging Telecommunications Networks: Needs, Trends, Challenges, Solutions* (Nejat Ince, A., E. Topuz, Eds.), Springer Sciences + Business Media, LLC 2006, 471–505.

[7] Todorova, M., Construction of Correct Object-Orientated Programs via Building their Generalized Nets Models, Annual of "Informatics" Section, Union of Scientists in Bulgaria, Vol. 4, 2011, 1–28.

[8] Todorova, M., Model Checker of Object-Oriented Programs Based on Generalized Nets. IWIFSGN'2011, Warsaw, September 30, 2011, *Recent Developments in Fuzzy Sets, Intuitionistic Fuzzy Sets, Generalized Nets and Related Topics. Vol. II: Applications,* 309–319.

[9] Todorova, M., Verification of Procedural Programs via Building there Generalized Nets Models. *Proc. of the 41st Spring Conference of the Union of Bulgarian Mathematicians. Mathematics and Education in Mathematics.* 9–12 April 2012, Borovets, 259–265.

[10] `http://www.ifigenia.org`