First International Workshop on GNs, Sofia, July 9, 2000, 26-28

An Algorithm for Transforming a Graph to a Generalized Net

Boyan Kolev – Technical University, Sofia, Bulgaria

Let a Generalized Net (GN, see [1]) $\check{E}$ is defined and for this GN is known the set of transitions A and for each transition $Z \in A$ are known the sets of incoming and outgoing places respectively L'_Z and L''_Z, i.e. we know the topological structure of the net $\check{E}$. Let the graph $G(V, E)$, where V is the set of nodes and E is the set of bows, is the result of transforming the GN $\check{E}$ by the operator Γ. The operator Γ transforms the GN $\check{E}$ to the graph G as to each place from $\check{E}$ it confronts an element from V and to each transition it confronts a set of bows. Each bow of this set connects a node corresponding to an incoming for the transition place with a node corresponding to an outgoing for the transition place. If we present a bow as an arranged couple of the numbers of the nodes it connects, we can claim that the subset $E_Z \subset E$, which corresponds to the transition $Z \in A$, corresponds to the cartesian product of the sets L'_Z and L''_Z.

$$\check{E} \xrightarrow{r} G,\ \bigcup_{Z\in A}(L'_Z \cup L''_Z) \xrightarrow{r} V, A \xrightarrow{r} E$$

and for each transition Z:

$$L'_Z \times L''_Z \xrightarrow{r} E_Z$$

For example the net on Fig.1 is transformed by the operator Γ to the graph on Fig.2.

Now we will define an operator Γ^1, which has the opposite action than the operator Γ and from the certain graph G restores the topological structure of the net $\check{E}$. We will also present an algorithm, which realizes the action of Γ^1. In fact this task is reduced to dividing the set E into non-intersecting subsets E_Z, and then determining the set A and for each $Z \in A$ determining the sets L'_Z and L''_Z. The algorithm also has to verify that the graph G is correctly defined, i.e. if on each step we determine that a certain bow (k, l) from E belongs to a certain subset E_Z, then we have to check if the following conditions are observed:

- the set of nodes connected with k by an outgoing from k bow coincides with the set L''_Z
- the set of nodes connected with l by an incoming to l bow coincides with the set L'_Z

Thus we can assure that eventually the following conditions will be observed:

$$\bigcup_{Z\in A} E_Z = E \bigcap_{Z\in A} E_Z = \varnothing,\ \ E_Z = L'_Z \times L''_Z\ \ \forall Z \in A$$

i.e. the set E will be divided into non-intersecting subsets E_Z and each of them will correspond to a transition Z of the GN $\check{E}$ and will coincide with the cartesian product of the sets of incoming and outgoing places for the transition Z.

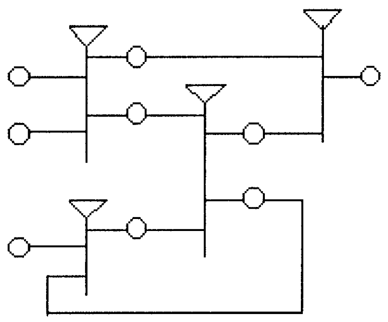
Fig. 1

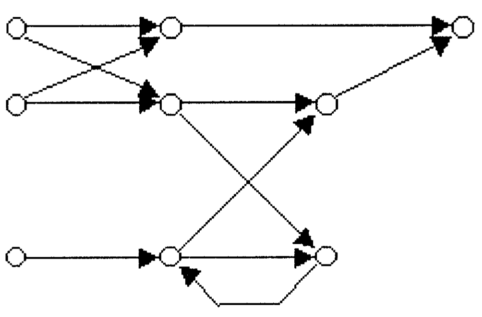
Fig. 2

We will present two variants for the algorithm – an iterative and a recursive. Each of the both variants subsequently separates the subsets E_Z and determines each element Z from the set A. With the first variant (iterative) while determining a transition Z the sets L'_Z and L''_Z are being determined in the beginning and afterwards the verification for correctness is being made. With the second variant (recursive) on each step an element is being added to one of the sets L'_Z or L''_Z and is being made a verification if this element is correctly added to the set.

Variant I:

```
1. If doesn't exist not visited (not marked) bow $(k, l) \in E$, then go to 9 else:
2. A consecutive number $Z$ for new transition is taken and the bow $(k, l)$ is marked with it, i.e. we add
   this bow to $E_Z$
3. To the set $L'_Z$ are added $k$ and all nodes connected with $l$ by an incoming to $l$ bow
4. To the set $L''_Z$ are added $l$ and all nodes connected with $k$ by an outgoing from $k$ bow
5. For each $i \in L'_Z \backslash k$:
   5.1. For each outgoing from $i$ bow $(i, j) \in E \backslash (i, l)$:
        5.1.1. If $j \notin L''_Z \backslash l$, then message "A bow is missing: $k \rightarrow j$" is printed and go to 9, else:
        5.1.2. $(i, j)$ is marked with $Z$
   5.2. If $\exists\, m \in L_Z'' \backslash X_i$, where $X_i$ is the set of nodes connected with $i$ by an outgoing from $i$ bow,
        then message "A bow is missing: $i \rightarrow m$" is printed and go to 9
6. $\forall n \in L''_Z \backslash l$:
   6.1. If exists an incoming to $n$ bow $(p, n) \in E$, which is not marked, then message "A bow is
        missing: $p \rightarrow l$" is printed, go to 9
```

```
7. $Z$ is added to $A$, go to 1
8. Message "The generalized net is determined successfully" is printed
9. End
```

Variant II:

It is realized with two mutually recursive procedures *DoLeft* and *DoRight.*

```
1. If doesn't exist not visited (not marked) bow $(k, l) \in E$, then go to 6 else:
2. A consecutive number $Z$ for new transition is taken
3. DoLeft($k$)
4. $Z$ is added to $A$, go to 1
5. Message "The generalized net is determined successfully" is printed
6. End
```

```
DoLeft($k$):
1. $k$ is added to $L'_Z$
2. $\forall j \in L''_Z \setminus X_k$, where $X_k$ is the set of all nodes connected with $k$ by outgoing from $k$ marked bow:
    2.1. If $(k, j) \notin E$, then message "A bow is missing: $k \rightarrow j$", end
    2.2. The bow $(k, j)$ is marked with $Z$
3. For each outgoing from $k$ bow $(k, n)$ that observes the condition $n \notin L''_Z$:
    3.1. $(k, n)$ is marked with $Z$
    3.2. DoRight($n$)
```

```
DoRight($l$):
1. $l$ is added to $L''_Z$
2. $\forall i \in L'_Z \setminus Y_l$, where $Y_l$ is the set of all nodes connected with $l$ by incoming to $l$ marked bow:
    2.1. If $(i, l) \notin E$, then message "A bow is missing: $i \rightarrow l$", end
    2.2. The bow $(i, l)$ is marked with $Z$
3. For each incoming to $l$ bow $(m, l)$ that observes the condition $m \notin L'_Z$:
    3.1. $(m, l)$ is marked with $Z$
    3.2. DoLeft($m$)
```

References:

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