Issues in IFSs and GNs, Vol. 13, 2017, 61–71

GNDraw – Software Application for Creating Generalized Nets

Nikolay Ikonomov

Institute of Mathematics and Informatics
Bulgarian Academy of Sciences
8 Acad. G. Bonchev Str., 1113 Sofia, Bulgaria
e-mail: nikonomov@math.bas.bg

Abstract: This article describes a modern software application for creating Generalized Nets. It is written in the Java programming language.
Keywords and phrases: Generalized Net, Software, Java.
2000 Mathematics Subject Classification: 68Q85.

1 Introduction

The software application GNDraw draws Generalized Nets, by having as input the structure of the net. Creator of the concept of Generalized Nets is Krassimir Atanassov [1]. The initial software for drawing under DOS is due to Nikolay Nikolov, PhD student of Kr. Atanassov. This application uses the same formatting of the input data:

```
# Example 1
transitions
Z1 : l1 -> l2
Z2 : l2 -> l3
Z3 : l3 i[1] -> l4
Z4 : l4 -> l5
columns
Z1; Z2, Z3; Z4
adjustY
Z1@1; Z4@2
```

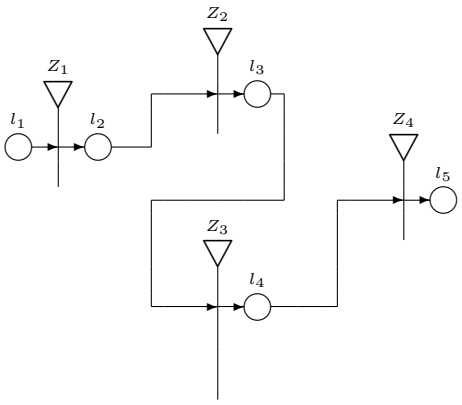

Figure 1: First example for a Generalized Net.

The net is shown on Figure 1. The transitions are denoted by triangle with a line down, each transition must have at least one input and one output place, they are denoted by a circle. The places and the transitions are connected by arcs. Tokens move through all these objects, and they take characteristics from the places. Tokens start their life at an input place, in the example l_1, and end their life at an output place, l_5. This is part of the functioning of the net, while this software application has its focus on the creation of the net.

2 Input for the Generalized Net

The structure of the Generalized Net has to be written between `transitions` and `columns`. The name of the transition is from the beginning of the row to a colon separator, then are the places separated by interval, while the input and output places are separated by `->`. Each transition is on its own row, there is no limit for the number of transitions or number of places for each transition.

The skeleton of the net is the first line after `columns`. Using semicolon as a separator between the transition names puts them in columns, while using comma puts them in rows. The example at Figure 1 has four transitions, which are ordered by `Z1; Z2, Z3; Z4`. Note that three columns are defined through `Z1; Z2; Z4`, and an additional row is defined by `Z3`, which is placed below `Z2`.

The manual adjustment of the transitions is the first line after `adjustY`. This does vertical adjustment of each defined transition, and it requires that the transition name already exists in the structure and the skeleton. Definitions must be separated by semicolon: the example at Figure 1 has `Z1@1; Z4@2`, which moves down Z_1 by two row, and Z_4 by four rows.

Phantom places can be defined by `i[1]`, which creates one phantom place, their count can be changed by the number in the brackets. The application creates unique values for each phantom place by using random floating point numbers. These places take space in the net, but are not shown: they are used for changing the layout of the net.

3 The path finding algorithm

The paths between the places are created by the path finding algorithm. The Generalized Net is saved internally as a matrix, therefore it uses the notation (row, column) for an element of the matrix. Let (a_i, a_j) be the coordinates of the starting place (where the path begins), and (b_i, b_j) – of the final place (where the path ends).

The places are ordered based on their priority: this is the sum of the absolute value of the difference between a_i and b_i, the absolute value of the difference between a_j and b_j, the absolute value of the distance to the closest free row (closest zero). The formula is:

$$|a_i - b_i| + |a_j - b_j| + |\text{closeZero}|.$$

This ordering connects the most inner input-output place first (which is the last place for the transition), then the one above it, and so on. This assures that the input-output places are connected without intersections in their paths.

The path finding algorithm searches for a free path (cells with value 0) and finds a number `indH`, which defines the horizontal path, a number `indV`, which defines the vertical path for the starting place, and a number `indVB` – the vertical path for the final place.

Find the horizontal path between a_j and b_j:

1. Check for a path at a_i, if the path is free, then assign $\texttt{indH} = a_i$.
2. Create a counter $cc = 1$. Check for free paths at $a_i - cc$ and $a_i + cc$.
3. Increase the value of cc, until a free path is found.
4. Assign its value: $\texttt{indH} = a_i - cc$ or $\texttt{indH} = a_i + cc$.

Find a secondary horizontal path between a_j and b_j:

1. Create a counter $cc = 1$. Check for a free path at $a_i + cc$.
2. If the difference between the horizontal path `indH` and the starting place a_i is bigger than the difference between the new path $a_i + cc$ and the final place b_i,
$$|\text{indH} - a_i| > |a_i + cc - b_i|,$$
then assign its value: `indH` $= a_i + cc$ (take the new path only if it is closer to b_i, than the old one to a_i).
3. Make the same check for $a_i - cc$.
4. Increase the value of cc, until a free path is found.

Find a vertical path between `indH` and a_i:

1. Check for a path at a_j, if the path is free, then assign: `indV` $= a_j$.
2. Create counter $cc = 1$. Check for free paths at $a_j - cc$ and $a_j + cc$.
3. Increase the value of cc, until a free path is found.
4. Assign its value: `indV` $= a_j - cc$ or `indV` $= a_j + cc$.

Find a vertical path between `indH` and b_i:

1. Check for a path at b_j, if the path is free, then assign: `indVB` $= b_j$.
2. Create counter $cc = 1$. Check for free paths at $b_j - cc$ and $b_j + cc$.
3. Increase the value of cc, until a free path is found.
4. Assign its value: `indVB` $= b_j - cc$ or `indVB` $= b_j + cc$.

Connect the vertical paths between a_i and `indH`, b_i and `indH`:

1. If the cell is empty (value 0), then save a vertical path, value 2.
2. If the cell has a horizontal path, then change to intersection, value 3.

Connect the horizontal paths between `indV` and `indVB`, also between a_j and `indV`, b_j and `indVB`:

1. If the cell is empty (value 0), then save a horizontal path, value 1.
2. If the cell has a vertical path, then change to intersection, value 3.

Set the angle type at (`indH`, `indV`), (`indH`, `indVB`), (a_i, `indV`), (b_i, `indVB`): upper-right with value 4, lower-right with value 5, upper-left with value 6, lower-left with value 7.

The internal matrix of the net is shown on Figure 2, the result of the path finding algorithm are the numeric values for each cell.

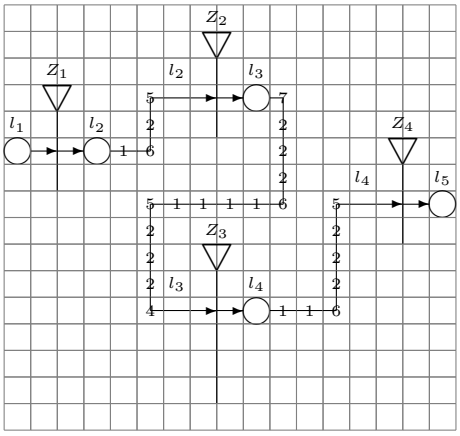

Figure 2: The path finding algorithm.

4 Phantom places

The phantom places can be used for changing the layout of the net, mostly for relocating the horizontal path. When `i[1]` is removed from the definition of transition Z_3 in Example 1, the horizontal path of l_3 moves from the top side to the bottom side of Z_3, it can be easily seen by running the application.

Consider this example:

```
# Example 2
transitions
Z1 : l1 l5 l6 -> l2
Z2 : l2 -> l3
Z3 : l3 i[1] -> l4
Z4 : l4 -> l5 l6
columns
Z1; Z2, Z3; Z4
adjustY
Z1@1; Z4@2
```

The net is displayed on Figure 3. Adding four phantom places `i[4]` at the end of the definition for transition Z_4 forces the horizontal path of l_6 to move from the bottom side to the top side of the net, as shown in Figure 4.

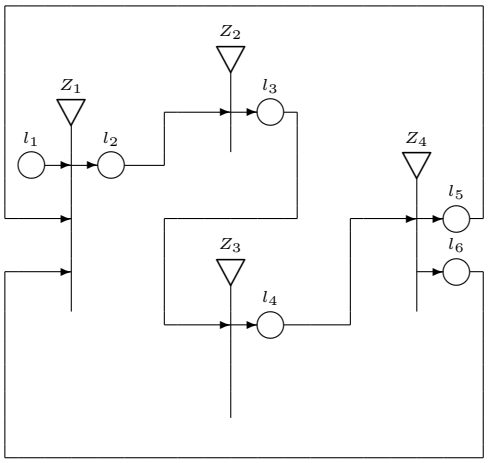

Figure 3: Second example for a Generalized Net.

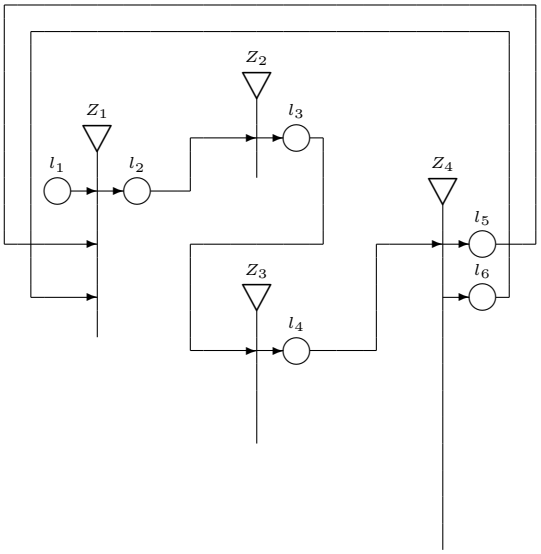

Figure 4: Changing the layout through phantom places.

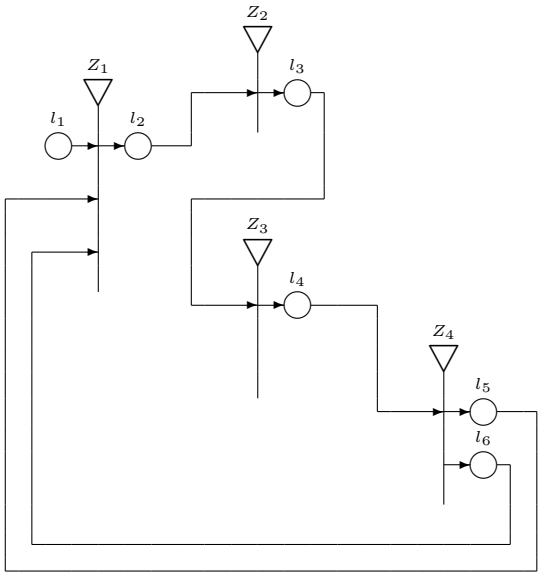

Figure 5: Changing the layout by moving a transition.

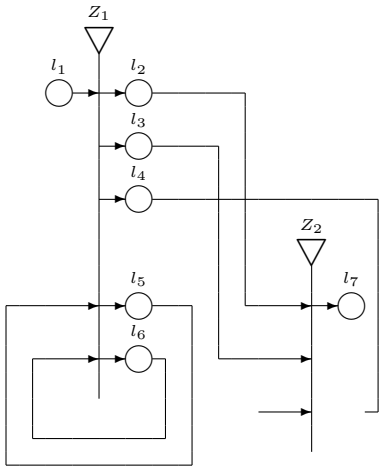

Figure 6: Third example for a Generalized Net.

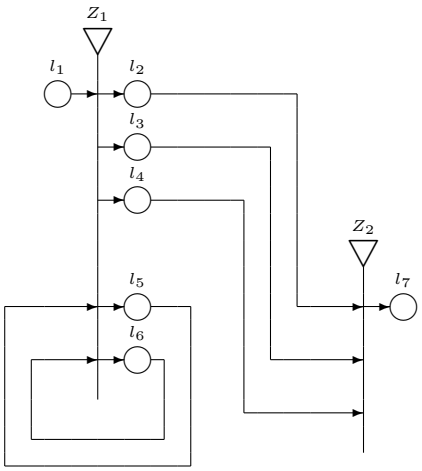

Figure 7: Changing the parameters for free rows and columns.

5 Moving a transition

Moving a transition changes the layout of the net. Changing the definition for manual adjustment `Z4@2` to `Z4@6` after `adjustY` in Example 2 (Figure 3), moves the horizontal path of l_5 from the top side to the bottom side of the net, shown on Figure 5.

One transition has be used as a reference point, it should not be moved. All other transitions are aligned with respect to it. In Example 2, the reference transition is Z_2.

The definition `Z1@1` adds two rows to the internal matrix above Z_1, while `Z4@6` adds 12 rows above Z_4. The formula is $\mathrm{val} * 2$.

6 Parameters for free rows and columns

Left of the button for drawing the Generalized Net, there is the possibility for changing two parameters: number of free rows top and bottom of the transition, number of free columns left and right of the transition. By default the values are 1 row and 2 columns, which means that there are two free rows between any two transitions (one row from each), and four free columns between any two transitions (two columns from each).

Consider this example:

```
# Example 3
transitions
Z1 : l1 i[3] l5 l6 -> l2 l3 l4 i[1] l5 l6
Z2 : l2 l3 l4 -> l7
columns
Z1; Z2
adjustY
Z2@4
```

The net is shown on Figure 6. Between Z_1 and Z_2 there are four free columns, by two from each transition. These columns are already occupied by the paths for l_2, l_3, l_5, l_6, and the path for l_4 cannot fit through there, since the parameters do not allow that. When the parameters are changed to 1 and 3, the path for l_4 is connected, as shown on Figure 7.

Summarizing, if there are unconnected paths, one of the following ways should be used for changing the layout of the net:

1. Changing the parameters for free rows and columns.
2. Moving a transition by adding `Z1@2` after `adjustY`.
3. Adding phantom places `i[2] i[3] i[5]` in the transition definition.

7 Installing the application

The application is written in the Java programming language, and requires installation of Java Runtime Environment [2]. After installation of the JRE, launch the application `GNDraw` from the file `GNDraw.jar` [3] (usually by double-clicking with the mouse). Launching on UNIX-based operating systems requires the execution of the command `java -jar GNDraw.jar` from a terminal.

8 User interface

The user interface consists of two panels: left panel for input of the structure of the Generalized Net, and right panel for drawing and exporting the net. Messages for the user are displayed in a small panel under the one for the input of the net. The user interface is shown on Figure 8.

Panel for input of the structure of the Generalized Net:

1. Open File: open an existing file.
2. Save File: save an already open file, or save as a new file.
3. Save Copy: save the input text as a new file under another name.

Right-clicking with the mouse on the panel shows a menu with text commands, the standard keyboard shortcuts for text manipulation are also available: Ctrl-X, Ctrl-C, Ctrl-V, Ctrl-Z, Ctrl-Y – cut, copy, paste, undo, redo. The text can be marked and then dragged to reorder it.

The parser for the Generalized Net (clicking the button with the same name) ignores blank lines and lines starting with `#`, they are considered comments, for example `#adjustY`.

Drawing the Generalized Net (right panel – top):

1. Rows Top/Bottom: parameter for free rows top and bottom of the transition.
2. Columns Left/Right: parameter for free columns left and right of the transition.
3. Generalized Net: parse the input text and draw the net.
4. View Data: show the internal tables for the net.

The internal tables consist of: skeleton (which saves the positions of the transitions), manual adjustment (integer value for each transition), node height (total rows that the transition occupies in the matrix), table 1 (the initial matrix for the net), table 2 (the matrix with paths), table 3 (with input/output place defined), table 4 (the final matrix).

Left-clicking and holding the mouse button on the net highlights the path with the connected place or the transition in red color.

Exporting the Generalized Net (right panel – bottom):

1. Change the view of the net: choose between normal view, grid view, or grid view and the paths (affects `PNG` and `TeX` export).
2. Choose the triangle for the transition: normal triangle by using LaTeX command `\bigtriangledown`, filled triangle with thin `\vector` or with thick `\vector` (affects `TeX` export).
3. Choose whether to show the text for transitions and places (affects `TeX` export).
4. Export: create an external file, where the net is saved as a `PNG` image or a `TeX` file.

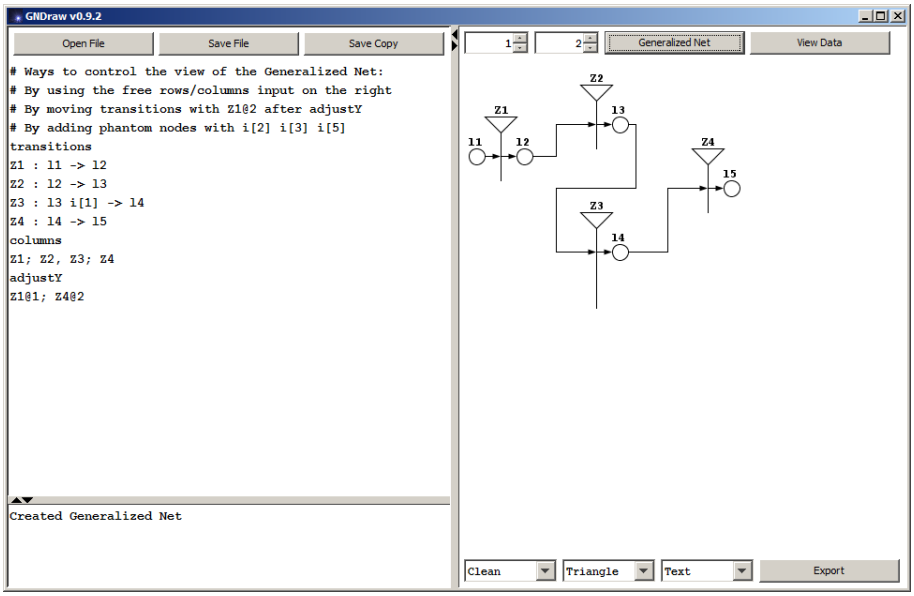

Figure 8: User interface.

9 Conclusion

The software application `GNDraw` is a complete solution for creating Generalized Nets: input of the net, layout of the net, drawing the net, exporting the net to different file formats. It is planned to extend the application with simulation of Generalized Nets.

References

[1] Atanassov, K., Theory of Generalized Nets (An algebraic aspect), *Advances in Modelling & Simulation*, AMSE Press, Vol. 1, 1984, No. 2, 27–33.

[2] `https://java.com/`

[3] `http://justmathbg.info/files/math/GNDraw092.zip`