PIGRA– A Tool for Pixelated Graph Representations

Thomas Bläsius, Fabian Klute, Benjamin Niedermann, and Martin Nöllenburg

Karlsruhe Institute of Technology (KIT)

At GD 2013 Biedl et al. presented a simple and versatile formulation of grid-based graph representation problems as integer linear programs (ILPs) and corresponding SAT-instances [1]. In a grid-based representation each vertex and each edge of a graph is represented by a set of grid cells, which we also call *pixels*. Biedl et al. described a general ILP model, where each object (vertex or edge) corresponds to a set of variables that determine which pixels represent the object. They introduced constraints that restrict the shapes of objects (e.g., requiring the pixels of a vertex to form a 2D box) and how the representations of different objects can intersect. In this way, one can solve a variety of NP-hard graph problems, including pathwidth, bandwidth, optimum st-orientation, area-minimal (bar-k) visibility representation, boxicity-k graphs and others. For example, in a grid-based drawing of a visibility representation, each vertex is represented by a horizontal box of height 1 and each edge is represented by a vertical box of width 1. Moreover, two boxes overlap if and only if they represent a vertex and an incident edge.

Biedl et al. [1] implemented and evaluated the ILP-models for the above problems. The experiments showed that their models successfully solve NP-hard problems within few minutes on small to medium-size graphs. They further provided their C++ implementation as the framework GDSAT[1], which can be freely downloaded and adapted.

With GDSAT it requires little effort to solve problems that are already modeled within GDSAT for a given graph. However, adapting the models to solve different grid-based graph drawing problems requires deeper insights and adaptions. Moreover, there is no graphical output of the result. Hence, it is not an easy-to-use tool for tasks like running initial experiments to explore new grid-based graph drawing problems.

To make the framework of Biedl et al. [1] more widely accessible and useful to the community, we developed the GUI-based tool PIGRA (**pi**xelated **gra**phs) that allows to easily combine pre-defined general constraints in order to model the above mentioned problems as well as other grid-based layout problems. Additionally, in case the pre-defined constraints are not sufficient, the user may adapt existing or define new constraints using the simple, mathematically-oriented language PGL (**p**ixelated **g**raphs **l**anguage), which we have introduced for this purpose.

In PIGRA the typical workflow consists of the following three steps; see Fig. 1.

1. The user defines the problem as a generic ILP-formulation in PIGRA, using combinations of pre-defined and custom constraints formulated in PGL.
2. PIGRA instantiates the ILP-formulation for a user-specified graph and solves it with GUROBI[2] (an automatic conversion to an equivalent SAT-instance is planned).
3. The resulting grid-based representation is graphically displayed and can be interactively explored.

[1] i11www.iti.kit.edu/gdsat – The name GDSAT comes from the fact that solving an equivalent SAT-instance performs better than solving the ILP itself.

[2] www.gurobi.com

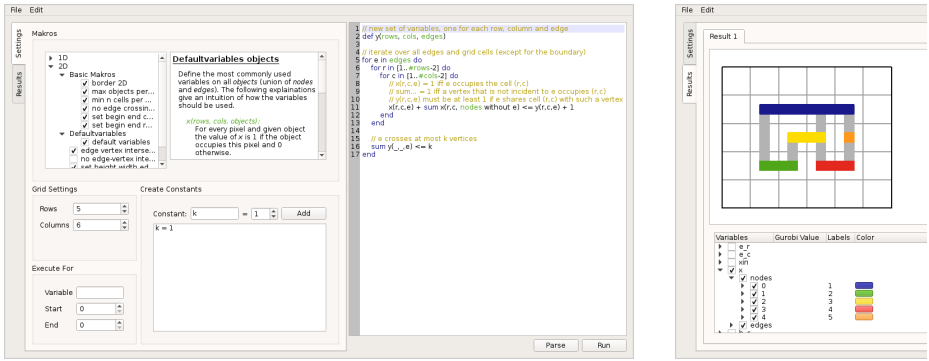

Fig. 1. Screenshots of a typical workflow in PIGRA using the example of bar-k visibility representations. Model formulation using a combination of pre-defined constraints and custom constraints expressed in PGL (left). Graphical output of the ILP solution for a small example graph (right).

PIGRA provides a variety of pre-defined ILP-constraints that are useful to formulate many custom grid-based graph drawing problems and can simply be selected by ticking the corresponding check-boxes. Among others, the following constraints are available: **(a)** vertices or edges are represented by boxes, **(b)** boxes have a certain height or width, **(c)** certain types of boxes may not overlap, **(d)** boxes of edges overlap exactly the boxes of their incident vertices. Additional constraints on shapes and intersections can be specified directly in PIGRA using PGL. For example, vertices could be modeled as the union of two boxes with non-empty intersection, including the special case of L-shapes. Or a constraint could be added so that each vertex box may only intersect a single incident edge, which models selecting a set of matching edges. In summary, the main features of PIGRA are:

- A macro system with pre-defined ILP-constraints for grid-based graph layouts.
- The simple, mathematically-oriented language PGL providing the capability to formulate ILP constraints with low overhead.
- A simple editor for writing constraints in PGL. Since all pre-defined constraints are also written in PGL, the user may adapt those constraints.
- A well-structured graphical user interface for presenting the result of the ILP as grid-based graph drawing.
- Support of GML-format for loading graphs (we use OGDF[3] to parse GML-files).
- Implemented in C++ and soon available for download[4] under the GPL.

References

1. Biedl, T.C., Bläsius, T., Niedermann, B., Nöllenburg, M., Prutkin, R., Rutter, I.: Using ILP/SAT to determine pathwidth, visibility representations, and other grid-based graph drawings. In: Wismath, S.K., Wolff, A. (eds.) Graph Drawing. LNCS, vol. 8242, pp. 460–471. Springer (2013)

[3] www.ogdf.net

[4] i11www.iti.kit.edu/pigra