An Invitation to newGRAPH

Dragan Stevanović
Department of Mathematics, Faculty of Science and Mathematics,
Višegradska 33, 18000 Niš, Serbia and Montenegro
dragance@pmf.ni.ac.yu

Vladimir Brankov
Mathematical Institute of Serbian Academy of Science and Arts,
Knez Miša Jovanova 35, 11000 Belgrade, Serbia and Montenegro
brankov@beotel.yu

Abstract

We present a concept of newGRAPH, a new version of GRAPH, one of the first computer programs for supporting research in graph theory by helping to pose, verify and disprove conjectures. newGRAPH is aimed to further improve research process in graph theory by making it more productive.

1 Introduction

Since their invention, computers were used in research to perform lengthy computations. However, this asks a researcher to have a knowledge of programming. This situation usually represents a waste of resources and it led to appearance of programs, more comfortable to the user, which already have the knowledge how to perform standard tasks in a given field.

We are interested in programs aimed to improving a research process in graph theory. There are a few such programs and certainly most widely known are Graffiti [7, 8], AutoGraphiX [1] and GRAPH [2, 3, 4, 5, 6, 9]. These three programs have very different ways of helping a graph theorist.

The goal of Graffiti, written by Siemion Fajtlowicz, is to find conjectures instead of a researcher. It asks for a set of graph invariants and then proposes a conjecture relating given invariants, which is then refined whenever the user supplies a counterexample. It has been reported that Graffiti was successfully used in education—students learn graph theory concepts and proof techniques by searching for counterexamples. Fajtlowicz maintains a file “Written on the Wall” of conjectures posed by Graffiti [8]. However, Graffiti is not publicly available, so that we cannot really consider it a program which could improve anyone’s research efforts, but its author’s or his students’.

AutoGraphiX, or in short AGX, written by Pierre Hansen and Gilles Caporossi, uses an optimization technique, a Variable Neighborhood Search metaheuristic [1], to find extremal graphs with respect to a function of graph invariants. This is then useful for conjecturing an inequality among graph invariants,
which is perhaps the most often form of a graph theory result. Also, the structure of extremal graphs may further give insights into a way of proving the conjecture.

GRAPH is the oldest among these programs—it was written from 1980-1984 by Dragoš Cvetković and his collaborators. Unlike previous programs, its goal is not to replace a researcher, but to help him pose, verify or disprove a conjecture. It represents an integrated environment for graph theory, so that the user may input, modify, visualize and store graphs, form new graphs by operations, calculate invariants, check properties, list families of subgraphs and sort graphs according to a given property, all without leaving a program. The most important feature of GRAPH is that it requires no programming by the user: communication is done using a subset of English language, bounded and formalized by a few rules. System GRAPH is widely used among spectral graph theorists: paper [5] surveys 55 papers by 16 authors published from 1982-1992, while paper [6] surveys 92 papers by 28 authors published from 1982-2001.

Our goal is to study a newGRAPH concept, a new version of GRAPH, which is currently being developed by the present authors, with Dragoš Cvetković and Slobodan Simić serving as consultants.

2 newGRAPH concept

Despite the fact that GRAPH is a fully integrated environment and that it was produced according to the highest standards of its time, computer operating systems and working environment improved very much in the last twenty years. It is now obvious that most of GRAPH features can be implemented in a much more time-efficient manner. Thus, we propose a new working environment which is described in the following subsections.

2.1 Research project in newGRAPH

Each research problem we are working on represents a separate research project in newGRAPH, consisting of a set of graphs, which, if necessary, may be further organized in subfolders, a set of graph invariants that we are interested in, a set of expressions, made up of these invariants, forming our conjectures, and, of course, a text file in which the researcher may write the description and notes for the problem studied.

A set of graphs in a project is presented within a treeview, and the main part of newGRAPH window is designated for graph windows. Many graph windows may be shown simultaneously, and each graph window shows a drawing of the graph, together with a table showing the values of user-defined sets of invariants and expressions.

Certain invariants, such as degree or main angles, may be visualized by showing the values next to vertices in the graph drawing. Actually, several invariants may be visualized at once by placing values in a specified order next to vertices.
Graph may be modified at any time, and the table of values is automatically recalculated. Usually, graphs we are working with are small and recalculation takes a small amount of time. However, if it turns out that it takes too much time for a specific graph, there is an option to turn off recalculation of any given invariant and depending expressions within that graph's window.

2.2 Graph editing

Most available graph editors permit only the basic operations. While these are enough to draw any graph, often there are situations when you get into waste-of-time mode with such editors: for example, the user might have drawn a graph occupying most space of its window and then she wants to add new vertices. Motivated by this and similar experiences, we designed a powerful, yet simple graph editor.

First of all, to resolve situations as the one mentioned above, the editor features commands that pan, zoom in or zoom out the drawing. There is also a command to scale in or scale out a selection of vertices, which may be used when the user wants to draw a new vertex in a space entirely occupied by other vertices. The editor features advanced selection commands that, for example, may select incident objects or an induced subgraph, making the selection process faster. There are also commands that transform a graph: for example, the user can make a selected part of the graph empty or complete in just one step.

Further, the editor enables faster drawing of graphs by being able to draw more than one edge at a time by drawing edges from a set of selected vertices to another vertex: thus, to draw edges from a vertex to all of its \( m \) neighbors the user can select the neighbors first, indicate that she wants new edges and then click a vertex asking for a total of \( m + 2 \) basic operations instead of \( 2m \), as would be needed by most other editors. Similarly, to draw a complete bipartite subgraph \( K_{m,n} \) the user would select one set of vertices, indicate the need for new edges and then select the other set of vertices, asking for a total of just \( m + n + 1 \) basic operations compared to \( mn! \).

However, simplicity was the most important when designing this editor. It is easy to learn, follows standard principles of selection and mouse performs almost every basic drawing operation. Advanced commands are placed in a context menu and therefore there are no buttons, avoiding clutter in a window and leaving more space for a drawing.

2.3 Plug-in architecture

newGRAPH is an open system, written to be highly extendable. Namely, its developers are interested mostly in spectral graph theory, and it is evident that they cannot implement everything a graph theorist might want. Thus, we think that newGRAPH would benefit much if researchers in other parts of graph theory have an option to implement new features.

From that reason, newGRAPH is highly modular: it consists of a small core enabling the most basic functionality, while virtually all commands and
invariant calculators are plugged into the core. The user is able to add new plug-ins without recompiling newGRAPH. Further, using newGRAPH class library, a programmer has an access to a graph structure and other plug-ins and is able to write new plug-ins. Thus, the user can easily modify newGRAPH to suit her own needs by selecting from available, downloading new or writing her own plug-ins.

3 Instead of a conclusion

newGRAPH is an attempt to make research much more productive. We presented features of newGRAPH, which grew out from our experience with existing programs, mainly with GRAPH.

newGRAPH is being implemented in Java, using a new SWT graphical library, and thus it is a cross-platform solution. At the moment of writing, it is still in its pre-1.0 version, shown in Fig. 1, which is freely available from http://www.pmf.ni.ac.yu/dragance/. But even as such, it is already used in research.
References


[6] D. Cvetković, S. Simić, Graph theoretical results obtained by the support of the expert system “GRAPH”—an extended survey, DIMACS Series in Discrete Mathematics and Theoretical Computer Science. Computers and Discovery in Graph Theory with Applications to Chemistry, accepted for publication.

