GrInvIn for Graph Theory Teaching and Research

Adriaan Peeters, Kris Coolsaet
Gunnar Brinkmann, Nicolas Van Cleemput
and Veerle Fack

TWI, Universiteit Gent, Krijgslaan 281 S9, B 9000 Gent,
Adriaan.Peeters,Kris.Coolsaet,Gunnar.Brinkmann,
Nicolas.VanCleemput,Veerle.Fack@UGent.be

Key words: graph, graph invariant, conjecture, software framework

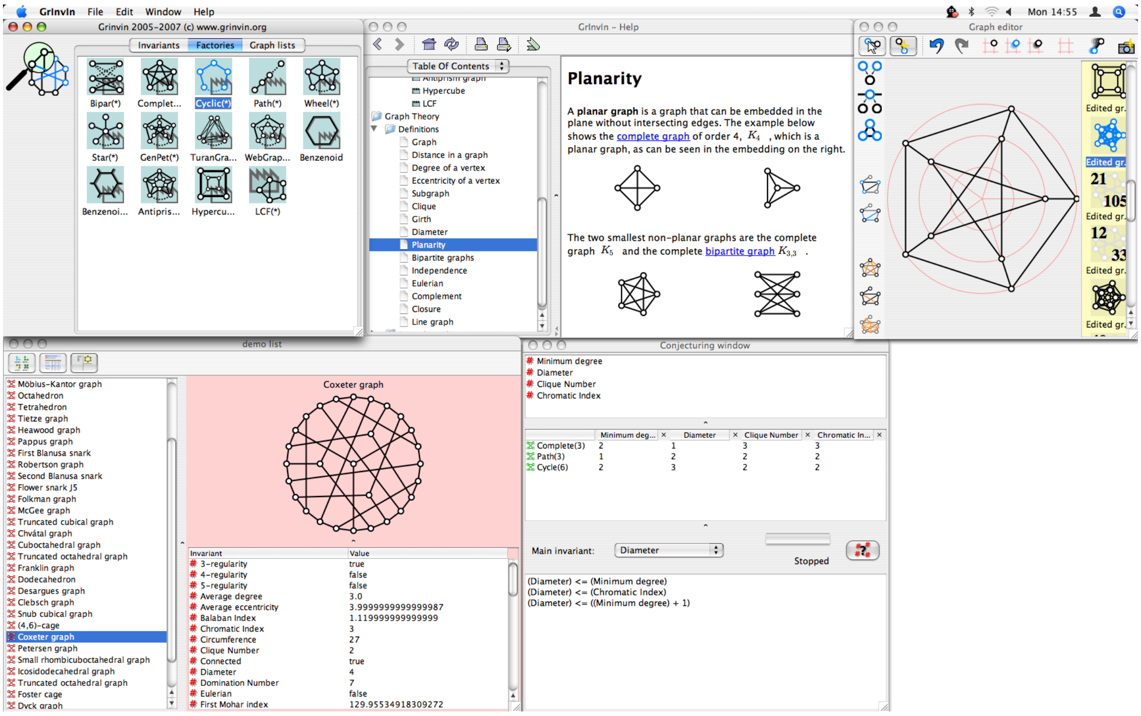
1 GrInvIn in a nutshell

Various programs to support research in graph theory have been developed and successfully used, such as AGX/AGX2[1], Cabri-graph[2], Graffiti[3], Graffiti.pc[4], GRAPH[5], GraPHedron[6], LINK[7], and newGRAPH[8]. Some of them emphasize the manipulation of graphs and computation of invariants, others focus on (graph) conjecturing.

As to the goal of GrInvIn we were most influenced by Graffiti.pc which was developed by Ermelinda DeLaVina. It was created for research in graph theory as well as for teaching graph theory by means of graph conjecturing.

The GrInvIn framework provides the core functionality needed to implement an application for graph theory in general. It includes basic functionality to work with graphs, invariants, and conjectures. In addition to data structures and interfaces for these concepts, the framework also provides a basic graph editor, various invariant computing routines, and an intuitive graphical user interface. GrInvIn is still being developed and soon further functionality (such as graph generation programs) will be added.

In order to guarantee portability, the interface and most of the subroutines are written in the highly portable programming language Java. Some parts that are performance critical and interact less with the operating system are written in C.



Planarity

A **planar graph** is a graph that can be embedded in the plane without intersecting edges. The example below shows the complete graph of order 4, K_4 , which is a planar graph, as can be seen in the embedding on the right.

The two smallest non-planar graphs are the complete graph K_5 and the complete bipartite graph $K_{3,3}$.

Invariant	Value
3-regularity	true
4-regularity	false
5-regularity	false
Average degree	3.0
Average eccentricity	3.9999999999999987
Balaban Index	1.1199999999999999
Chromatic Index	3
Circumference	27
Clique Number	2
Connected	true
Diameter	4
Domination Number	7
Eulerian	false
First Mohar index	129.95534918309272

	Minimum deg...	Diameter	Clique Number	Chromatic In...
Complete(3)	2	1	3	3
Path(3)	1	2	2	2
Cycle(6)	2	3	2	2

Main invariant: Diameter

```
(Diameter) <= (Minimum degree)
(Diameter) <= (Chromatic Index)
(Diameter) <= ((Minimum degree) + 1)
```

Fig. 1. Screenshot of the GrInvIn user interface.

1.1 *User interface*

The GrInvIn user interface (Figure 1) makes use of the *drag and drop* concept to work with graphs and invariants. All GrInvIn objects are represented as icons. This user interface works in a way similar to that of the file manager found in most recent operating systems.

To allow the user to work in their own native language, the entire user interface is internationalized. Translating the framework only involves providing a few language files, without any changes in the application source code. So far GrInvIn comes with language files in English and Dutch.

1.2 *Teaching Graph Theory with GrInvIn*

GrInvIn comes with a conjecturing engine making conjectures, or rather *guesses*, based on a set of graphs given to the program by the user. These graphs can be input via a graph editor or a graph *factory* in case the graph belongs to one of several well known classes – such as complete graphs, cycles, paths, etc.

The intention of this conjecturing engine is to be used for teaching graph theory to university or even high school students who make their first contacts with graph theory.

The basic strategy follows Fajtlovicz and DeLaVina who first applied these principles using their programs *Graffiti* and *Graffiti.pc*: The student starts with a set of invariants and a very small list of graphs – e.g. containing just K_1 or K_3. He chooses an invariant for which he wants to determine (e.g. upper) bounds. Conjectures based on this small set of graphs can in general easily be shown to be false and often very small counterexamples exist. The student's task is now to prove or disprove the conjecture given by the program. In case of a wrong conjecture the student has to prove that his counterexample is smallest possible and add that counterexample to the list. Then a new conjecture is computed for this larger list of graphs and the process is repeated. In case of a true conjecture, the student can prove the conjecture and try to find out whether the conjecture always gives a *good* bound for his invariant or whether the difference between the left hand and the right hand side can be arbitrarily large. Afterwards he can either input a graph where the difference between the two sides is very large or delete one of the invariants from the list to force the program to make new conjectures.

Courses based on this strategy were given in Houston and later also in Bielefeld and Ghent. At the time of the conference, first courses of this kind will also have taken place already on the level of high school students in Ghent.

Experience showed that this way to make contact with graph theory was very motivating for students. They were keen on learning about the new invariants occuring in the conjectures, because the conjecture connected the new invariant to *their* chosen invariant. Furthermore this way of learning has some air of discovery: There is always at least the chance to discover some new result – instead of just learning basic results studied already by generations of mathematicians.

In Bielefeld (where *Graffiti.pc* was used) the effect on the students was very positive: the students did not only enjoy the course – afterwards they also asked for a *normal* graph theory course to learn more about graph theory and proof techniques in graph theory and half of the students later decided to write their thesis in this field.

In our eyes graph theory is also ideal to teach logical thinking and argumentation to students already at high-school level and hopefully *GrInvIn* will turn out to be a good tool to do so.

Unfortunately neither *Graffiti* nor *Graffiti.pc* are publically available so far. *Graffiti* was never meant to be distributed and – also due to lack of resources – *Graffiti.pc* develops also very slowly.

Though being inspired by *Graffiti.pc*, *GrInvIn* is not based on *Graffiti.pc*, but was completely designed from scratch. The emphasis lies on the software engineering aspect with the aim to combine the functionality of *Graffiti.pc* with a modern software design, a user friendly interface, and easy extensibility.

1.3 Content of the Talk

In this talk we will give a short introduction into the use of *GrInvIn* and show an example scenario of a teaching session.

References

[1] Gilles Caporossi and Pierre Hansen, Variable neighborhood search for extremal graphs: 1 The AutoGraphiX system., Discrete Mathematics, 212, 1-2, 2000, p.29–44, http://dx.doi.org/10.1016/S0012-365X(99)00206-X.

[2] Yves Carbonneaux, Jean-Marie Laborde, and Rafaï Mourad Madani, CABRI-Graph: A Tool for Research and Teaching in Graph Theory., Graph Drawing, 1995, p.123–126.

[3] Ermelinda DeLaVina, Some History of the Development of Graffiti. DIMACS Series in Discrete Mathematics and Theoretical Computer Science 69: Graphs and Discovery, 2005, p. 81–118.

[4] Ermelinda DeLaVina, Graffiti.pc: A Variant of Graffiti, DIMACS Series in Discrete Mathematics and Theoretical Computer Science 69: Graphs and Discovery, 2005, p.71–79.

[5] Dragoš Cvetković and Slobodan 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 69: Graphs and Discovery, 2005, p.39–70.

[6] Hadrien Melot, GraPHedron, see `http://www.graphedron.net`.

[7] Jonathan W. Berry, Nathaniel Dean, Mark K. Goldberg, Gregory E. Shannon, and Steven Skiena, Graph Drawing and Manipulation with *LINK*., Graph Drawing, 1997, p.425–437.

[8] Dragan Stevanović, Vladimir Brankov, Dragoš Cvetković, and Slobodan Simić, newGRAPH, see `http://www.mi.sanu.ac.yu/newgraph/`.