Available online at www.sciencedirect.com
ScienceDirect
Discrete Applied Mathematics 156 (2008) 1875–1891
DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

Facet defining inequalities among graph invariants: The system GraPHedron

Hadrien Mélot

Department of Computer Science, Université de Mons-Hainaut, Avenue du Champ de Mars 6, B-7000 Mons, Belgium

Received 25 October 2005; received in revised form 3 July 2007; accepted 5 September 2007
Available online 24 October 2007

Abstract

We present a new computer system, called GraPHedron, which uses a polyhedral approach to help the user to discover optimal conjectures in graph theory. We define what should be optimal conjectures and propose a formal framework allowing to identify them. Here, graphs with n nodes are viewed as points in the Euclidian space, whose coordinates are the values of a set of graph invariants. To the convex hull of these points corresponds a finite set of linear inequalities. These inequalities computed for a few values of n can be possibly generalized automatically or interactively. They serve as conjectures which can be considered as optimal by geometrical arguments.

We describe how the system works, and all optimal relations between the diameter and the number of edges of connected graphs are given, as an illustration. Other applications and results are mentioned, and the forms of the conjectures that can be currently obtained with GraPHedron are characterized.

Keywords: Graph; Conjecture; Computer-assisted system; GraPHedron

1. Introduction

Since the early eighties, several software systems have been used to make conjectures in graph theory [35,38]. Hansen [35] divides such systems into two classes: *automated systems* that provide conjectures in a fully automated way (i.e., without human intervention apart for the problem statement), and *computer-assisted systems* on the other hand. Derived conjectures are then said to be obtained *by* an automated system or *with* a computer-assisted one. Notice that in practice an automated system can often also be used interactively and leads to both automated and/or computer-assisted conjectures.

Among such computer systems, we just mention the pioneering system *Graph* due to Cvetković et al. [11–16], the *Graffiti* system of Fajtlowicz et al. [18,20–26] and the system *AutoGraphiX* developed by Caporossi and Hansen [1,4–7,17,27,34,39–41]. According to Hansen [35] they are the main operational systems in the field, each one leading to dozens of papers:

> Collectively, this number of papers is large (over 200) and perhaps unequaled in the field of discovery science.

E-mail address: hadrien.melot@umh.ac.be.

0166-218X/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2007.09.005

It appears that the forms of conjectures found by and/or with these systems are often relations among graph *invariants*, i.e., numerical values associated with each graph G preserved by isomorphism. This is also often the case in the literature. However, some theorems—discovered without computer-assistance—have forms that are not considered by the current systems [36].

We present here a new computer-assisted system, called GraPHedron (which can, in some cases, give conjectures in a fully automated way). Considering that conjectures in graph theory are often relations among graph invariants, the system tries to answer the following question:

What are all the best inequalities among a selected set of invariants, valid for graphs of a given class?

As the paper's title suggests, the arguments used to answer this question will be taken from polyhedral theory. Actually, the name GraPHedron is the contraction of the words *graph* and *polyhedron*.

This paper is organized as follows. In Section 2, we give some notations and definitions. In Section 3, we explain the principles of GraPHedron, and give our answer to the above question. The outline of this system is explained in Section 4. We next present applications and first results in Section 5. As the goal of this paper is to present the system, the results presented in Section 5 will be viewed as an illustration and references to more important results will be given. The different forms of conjectures that can be currently derived with GraPHedron are characterized in Section 6. Finally, we draw some concluding remarks in Section 7.

2. Notations and definitions

Classical notions of graph theory and polyhedral theory will be used. Readers that are not familiar with these notions are referred to standard textbooks on graphs [3,19] and on polytopes [46]. However, we recall and fix some notations and definitions in this section.

We consider simple, loop-free, undirected graphs $G(V, E)$ with node set V and edge set E. The number of nodes of G is denoted by $n(G) = |V|$ and the number of edges by $m(G) = |E|$. The *degree* of a node v, denoted by $d_G(v)$, is the number of nodes w such that $\{v, w\} \in E$. The *distance* $d_G(v, w)$ between two nodes v and w is the length (i.e., the number of edges) of a shortest path between v and w. We note $G \simeq H$ if the graph G is isomorphic to the graph H.

Classical graphs will be used: *stars*, denoted by S_n, *paths* denoted by P_n and *complete graphs* K_n, where n is the number of nodes. A complete graph is also called a *clique*.

Definition 1. A (graph) *invariant* $i(G)$ is a numerical value associated with a graph G and preserved by isomorphism.

For instance, the integers $n(G)$ and $m(G)$ are invariants. Invariants can also have boolean, rational or irrational values. As graphs are used in many different applications, there exist dozens of invariants describing specific characteristics of graphs. In the following we will use the *diameter* $D(G)$ which is the maximum distance between two nodes of G; the *maximum degree* $\Delta(G) = \max_{v \in V} d_G(v)$ and the *stability number* $\alpha(G)$ which is the maximum cardinality of a set of non-adjacent nodes.

When the context is clear, we often omit to note explicitly G in the above notations.

Definition 2. The *set of invariants* $\mathscr{I} = \{i_1, i_2, \ldots, i_p\}$ is a set of p algebraic expressions involving one or more invariants.

This definition is not ambiguous because an algebraic expression involving several invariants is also an invariant. For a given set $\mathscr{I}$ of p invariants, one can associate to a graph G a point $(i_1(G), i_2(G), \ldots, i_p(G))$ in the *space of invariants* $\mathbb{S}^p$. Depending of the type of the invariants values, $\mathbb{S}^p$ can be $\mathbb{Z}^p$, $\mathbb{Q}^p$ or $\mathbb{R}^p$ (boolean values are considered as integers where true is 1 and false is 0).

Definition 3. Let S be a finite set of points in $\mathbb{S}^p$. The p-dimensional *polytope* $\mathscr{P} = \text{conv}(S)$ is the convex hull of S.

From a geometrical point of view, a polytope is an intersection of half-spaces. A p-dimensional polytope can thus be defined as a set of solutions $x \in \mathbb{S}^p$ of a system of k linear inequalities

$$Ax \leqslant b, \tag{1}$$

where $A \in \mathbb{S}^{k \times p}$ is a $k \times p$ matrix and $b \in \mathbb{S}^k$ is a k-vector.

A *face* of $\mathscr{P}$ is the intersection of $\mathscr{P}$ with a tangent hyperplane. Zero-dimensional faces are *vertices*, one-dimensional faces are *polytope edges* and $p-1$-dimensional faces are *facets*. An inequality which is a facet of $\mathscr{P}$ is called a *facet defining inequality*. Note that to avoid confusion we use the words *node* and *edge* for graphs and *vertex* and *polytope edge* for polytopes.

Definition 4. $\mathscr{F}(\mathscr{P})$ is the system of facet defining inequalities describing $\mathscr{P}$.

Definition 5. Let $\mathscr{C}$ be a specific *class of graphs* (e.g., connected graphs). We note $\mathscr{C}_n$ the set of all non-isomorphic graphs with n nodes, belonging to $\mathscr{C}$.

3. Principles of GraPHedron

We present here the principles of GraPHedron, a computer system which helps to find optimal conjectures in graph theory. First, in Section 3.1, the polyhedral approach of the system is explained. This approach is illustrated in Section 3.2 by an example. Finally, we explain in Section 3.3 the differences between this approach and a similar procedure used in the system *AutoGraphiX* [6,7].

3.1. A polyhedral approach

As already stressed in the Introduction (and even if it is not always the case [36]), theorems in graph theory are often expressions (equalities or inequalities) involving a set of invariants, under some conditions that graphs should respect. These conditions, or hypotheses, are generally a specific class of graphs $\mathscr{C}$. For example, a theorem can be valid only for connected, bipartite or planar graphs. We ask the following question:

What are all the best linear inequalities among invariants of $\mathscr{I}$, valid for all graphs of $\mathscr{C}_n$?

Actually, to answer this question, one needs to answer some related ones:

- How to define a "best" or "optimal" linear inequality when n is given?
- What means "all" inequalities for a given problem?
- How to find these inequalities?

We derive answers from a polyhedral approach, considering graphs as points in the Euclidian space.

Definition 6. For a given class of graph $\mathscr{C}$, a given set of invariants $\mathscr{I}$ and a fixed integer n, we define the *invariants polytope* $\mathscr{P}_n^{\mathscr{C},\mathscr{I}}$ as

$$\mathscr{P}_n^{\mathscr{C},\mathscr{I}} = \text{conv}\{(x_1, x_2, \ldots, x_p) \in \mathbb{S}^p \mid \exists G \in \mathscr{C}_n, i_1(G) = x_1, i_2(G) = x_2, \ldots, i_p(G) = x_p\}.$$

When no confusion is possible, $\mathscr{P}_n^{\mathscr{C},\mathscr{I}}$ is simply denoted by $\mathscr{P}_n$. This polytope can be described by a system $\mathscr{F}(\mathscr{P}_n)$ of facet defining inequalities. Any such linear inequality can be considered as best possible:

- any valid linear inequality among the invariants of $\mathscr{I}$ is dominated by a positive combination of facet defining inequalities $\in \mathscr{F}(\mathscr{P}_n)$;
- $\mathscr{F}(\mathscr{P}_n)$ constitute a minimal system describing $\mathscr{P}_n$, i.e., no facet defining inequality $\in \mathscr{F}(\mathscr{P}_n)$ can be a logical consequence of any other valid inequalities.

This definition of optimality is thus stronger than a "tight" inequality—a classical argument of quality—which means only that the inequality defines a supporting hyperplane of $\mathscr{P}_n$.

From the computation of $\mathscr{P}_n$, we can derive a fruitful strategy to formulate conjectures. If n is small, $\mathscr{P}_n$ can be computed effectively with a computer. The idea of the computer system GraPHedron is to compute the polytopes $\mathscr{P}_n$ for some reasonable values of n and to display detailed information about them. If similarities between a facet of each $\mathscr{P}_n$ can be pointed out, one often obtains a conjectured generalization of this facet for all n. Sometimes, the complete

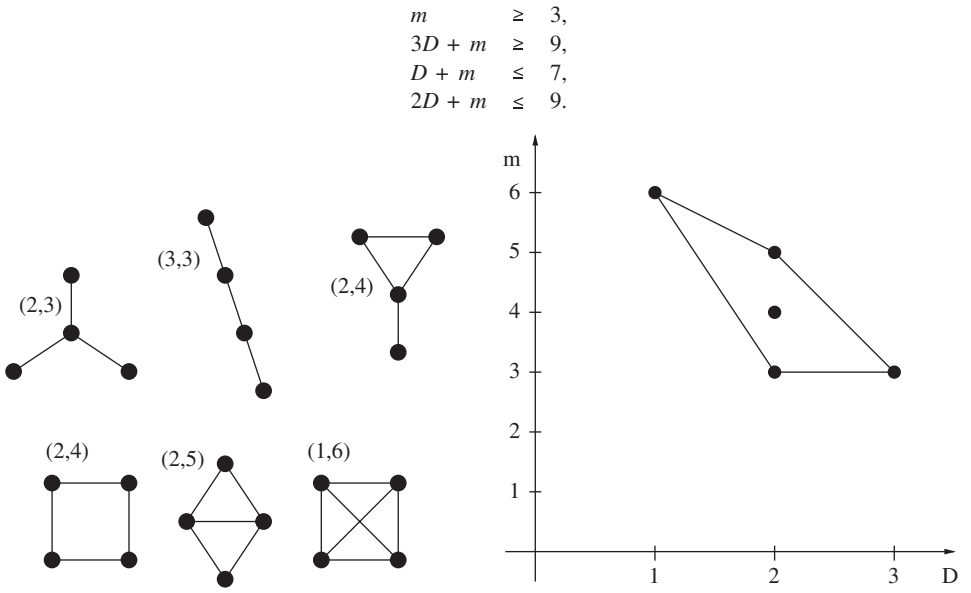

Fig. 1. Graphs of $\mathscr{C}_4$, their coordinates (D, m) and the polytope $\mathscr{P}_4$.

system $\mathscr{F}(\mathscr{P}_n)$ can be conjectured and generalized for each n. The characterization of the *vertex-graphs* is also very helpful to derives conjectures.

Definition 7. Let $G \in \mathscr{C}_n$ be a graph and $\mathscr{P}_n$ an invariants polytope. If the point $(i_1(G), i_2(G), \ldots, i_p(G))$ is a vertex of $\mathscr{P}_n$, then G is called a *vertex-graph* of $\mathscr{P}_n$.

3.2. Illustration

To illustrate the polyhedral approach, we introduce a simple but beautiful example.

Example 1. What are all the best linear inequalities among the diameter D and the number of edges m of connected graphs with n nodes?

Thus $\mathscr{I} = \{D, m\}$ and $\mathscr{C}$ is the class of connected graphs. Fig. 1 shows all graphs in $\mathscr{C}_4$, the associated vectors of coordinates (D, m) and the corresponding polytope $\mathscr{P}_4$. This polytope is defined by the following set of facet defining inequalities:

$$m \geqslant 3,$$

$$3D + m \geqslant 9,$$

$$D + m \leqslant 7,$$

$$2D + m \leqslant 9.$$

Polytopes $\mathscr{P}_n$ with larger values of n (i.e., $n \leqslant 11$) can then be computed. Fig. 2 shows representations of $\mathscr{P}_9$ and $\mathscr{P}_{10}$ produced by GraPHedron.

Figs. 3 and 4 show drawings of the vertex-graphs of $\mathscr{P}_9$ and $\mathscr{P}_{10}$, as displayed by GraPHedron.

A look at the information displayed by GraPHedron for the polytopes $\mathscr{P}_n$ $(n = 4, 5, \ldots, 11)$ leads to the following observations. The polytopes have always four facets and a very similar shape. Each vertex always corresponds to only one graph. These vertex-graphs are easily characterized: the star S_n, the path P_n, the complete graph K_n and the

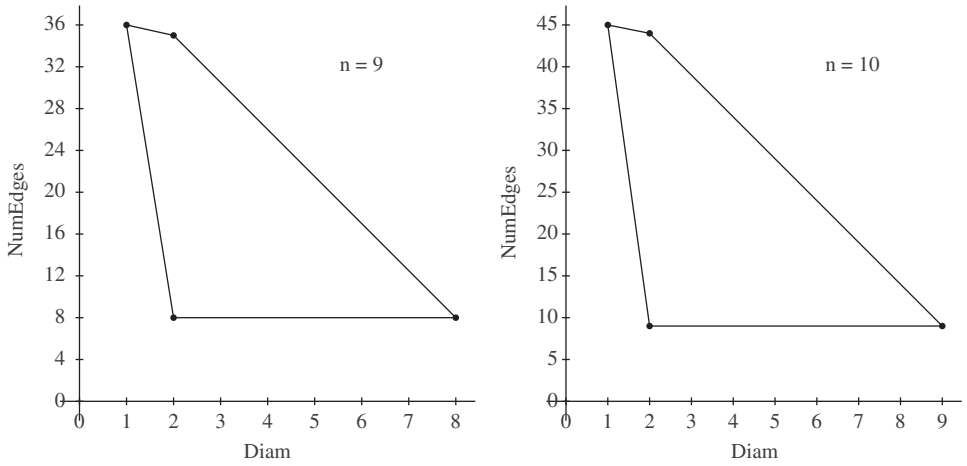

Fig. 2. The polytopes $\mathscr{P}_9$ and $\mathscr{P}_{10}$.

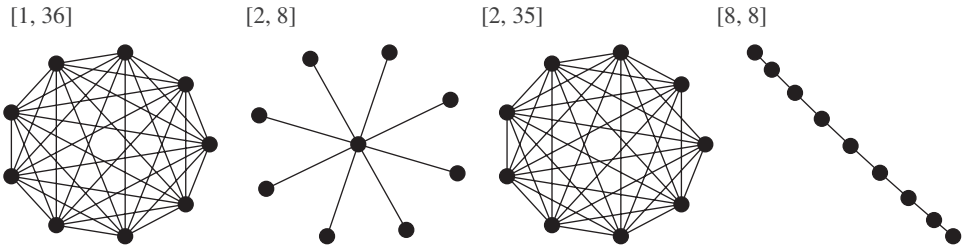

Fig. 3. The vertex-graphs of $\mathscr{P}_9$.

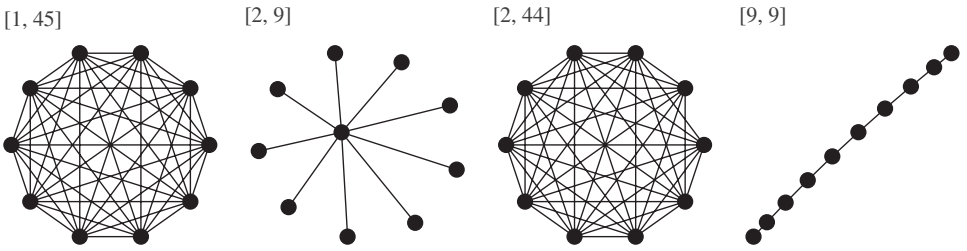

Fig. 4. The vertex-graphs of $\mathscr{P}_{10}$.

complete graph with a removed edge $K_n \backslash e$. The coordinates (D, m) of these graphs are trivially generalized in terms of n:

$$S_n : (2, n-1), \tag{2}$$

$$P_n : (n-1, n-1), \tag{3}$$

$$K_n : \left(1, \binom{n}{2}\right), \tag{4}$$

$$K_n \backslash e : \left(2, \binom{n}{2} - 1\right). \tag{5}$$

If one conjectures that the coordinates associated with these graphs are always vertices of $\mathscr{P}_n$ (and that no other graph corresponds to a vertex), the facet defining inequalities are also easily generalized. To avoid trivialities (as $S_3 \simeq P_3$), we assume that $n \geqslant 4$.

Conjecture 1. *For each connected graph G with $n \geqslant 4$ nodes, m edges and a diameter D,*

$$m \geqslant n - 1,$$
$$\frac{(n-1)(n-2)}{2} D + m \geqslant (n-1)^2,$$
$$D + m \leqslant \frac{n(n-1)}{2} + 1,$$
$$nD + 2m \leqslant (n+2)(n-1),$$

and these inequalities are the only possible facet defining inequalities for each n.

This conjecture is proven in Section 5.1.

3.3. The geometric procedure of AutoGraphiX

This polyhedral approach is simple and natural but was not yet been exploited systematically.

The system *AutoGraphiX* [6,7] uses an efficient meta-heuristic to obtain a set of graphs G_k for a parameter $k = k_{\min}, \ldots, k_{\max}$. These graphs are extremal or near-extremal for a given objective function $f(G)$. *AutoGraphiX* applies three different methods to obtain conjectures automatically from the set G_k: a numerical procedure which uses techniques of Data Mining (a variant of principal component analysis); an algebraic procedure based on the recognition of the graphs G_k and a geometric procedure, which consists in considering the graphs G_k "as points in a space of characteristics, then uses a convex-hull (or gift-wrapping) algorithm to find facets, which correspond to conjectures" [7, p. 83]. The latter procedure is thus similar to our approach but there are important differences.

Caporossi and Hansen restrict the points to a set of graphs which are extremal or near-extremal for $f(G)$ and they study the facets relevant for the type of the optimization problem (minimization or maximization): the coordinates are thus $(f(G_k), k)$. Of course, the order of graphs G_k is larger than what can be obtained by enumeration.

On the contrary, we consider all non-isomorphic graphs of $\mathscr{C}$ and try to characterize all the facets of the convex hull of a set of points in a p-dimensional space. It allows to identify a finite number of optimal relations from which all other relations follow, i.e., a minimal system of optimal relations. Moreover, we output the polyhedral information about the problem which can be useful to understand the relations between invariants of $\mathscr{I}$ and to prove the derived conjectures.

4. Outline of the system

GraPHedron—developed by the author in the context of his PhD dissertation [43]—is written in C++ and is developed to run on Unix-like systems. The current version is a console application. A web interface to this system is publicly available [33] since January 2007.

The input of the program is the definition of a given *problem* written in a text file. A problem is defined by a set of invariants $\mathscr{I}$, a class of graphs $\mathscr{C}$, a value's range for $n = n_{\min}, \ldots, n_{\max}$ and several options. In Section 4.1, we explain which expressions of invariants are recognized by GraPHedron. These expressions can be used in the problem statement, reviewed in detail in Section 4.2.

The output is a set of various files. Among them is a report written in LaTeX containing a complete description of the systems $\mathscr{F}(\mathscr{P}_n)$ (for $n = n_{\min}, \ldots, n_{\max}$), representations of the polytopes and the vertex-graphs, automated conjectures, etc.

To produce this output, GraPHedron includes the following three stages:

1. Data generation (graphs and invariants).
2. Polytope computation.
3. Creation of the report and derivation of conjectures.

Table 1
Arithmetic operators

Description	Accepted syntax	Arity	Prec.
Power	x^q	Binary	11
Unary minus	−x	Unary	10
Multiplication	x*y	Binary	9
Division	x/y	Binary	9
Modulo	x%q or x mod y	Binary	9
Plus	x+y	Binary	8
Minus	x-y	Binary	8
Maximum	MAX(x,y,...)	*k*-ary	7
Minimum	MIN(x,y,...)	*k*-ary	7
Mean	MEAN(x,y,...)	*k*-ary	7
Variance	VAR(x,y,...)	*k*-ary	7
Floor	FLOOR(x)	Unary	7
Ceil	CEIL(x)	Unary	7
Round	ROUND(x)	Unary	7

Table 2
Relational operators

Description	Accepted syntax	Arity	Prec.
Less	x < y	Binary	6
Greater	x > y	Binary	6
Less or equal	x <= y	Binary	6
Greater or equal	x >= y	Binary	6
Equal	a = b or a == b	Binary	5
Not equal	a != b or a <> b	Binary	5

Table 3
Logical operators

Description	Accepted syntax	Arity	Prec.
Not	NOT a or !a	Unary	4
And	a AND b or a && b	Binary	3
Exclusive or	a XOR b	Binary	2
Or	a OR b or a||b	Binary	1

Each stage can be carried out independently. The system will check if previous stages have to be done to realize the current one. Details of each step are explained in Sections 4.3–4.5.

Finally, in Section 4.6, we give the current limitations of GraPHedron in terms of time and storage. We also explain the choices made in the implementation, in order to address these limitations.

4.1. Algebraic expressions of invariants

An invariant of $\mathscr{I}$ can be defined as an algebraic expression of other invariants. One can use three types of operators to build an expression. Arithmetic operators are used to construct *arithmetic expressions* and relational and logical operators allow the construction of *boolean expressions*. Boolean expressions are generally used to define $\mathscr{C}$ (see Section 4.2).

Tables 1–3 list the operators currently available in GraPHedron where x and y are arithmetic expressions, q an integer and a and b are boolean expressions. Operator precedence determines the order in which the terms of an expression

will be evaluated. Operators with the highest precedence will be evaluated before operators with lower precedence. One can use parenthesis to enforce non-default precedence.

If an expression E uses an invariant with an irrational value, or a constant written in floating notation, then E is called an *approximated expression*. If all expressions involved in a given problem are non-approximated, all computations are made in exact arithmetic using the *GMP* library [32]. Otherwise, the user has to be aware of possible round-off errors: all values are converted to floating point numbers.

4.2. Problem statement

A *problem* is defined from this information:

1. *Set of invariants* $\mathscr{I}$: Each element of $\mathscr{I}$ is an algebraic expression of invariants, also called a *coordinate* of the problem. In the case of a boolean expression, its value is considered as an integer (true is 1 and false is 0). There exist currently about 70 invariants implemented in GraPHedron. The code is written in such a way that new invariants can be added easily.
2. *Class of graphs* $\mathscr{C}$: Graphs considered can be restricted to a specific class $\mathscr{C}$ in two ways:
   (a) *Selective generation*: GraPHedron uses, as a sub-routine, the graph generator `geng` of McKay [42]. The program `geng` allows one to restrict the graphs generated by fixing some parameters (e.g., minimum and maximum degree, minimum and maximum number of edges, etc.). These parameters are fixed internally by GraPHedron when the user asks to generate graphs from one of the following classes: trees, bipartite graphs, k-regular graphs, triangle-free graphs (graphs without K_3 sub-graphs) or C_4-free graphs (graphs without cycles with four nodes). These classes can be combined. For instance, one can generate only bipartite cubic graphs, i.e., bipartite 3-regular graphs. For a given problem, we note $\mathscr{C}_n^{\text{sel}}$ the set of all non-isomorphic graphs with n nodes generated by selective generation.
   (b) *Graph filtering*: The user can define a set of *conditions*, i.e., boolean expressions of invariants (see Section 4.1), that graphs should respect. Each graph enumerated by selective generation will be tested and accepted only if all conditions are true for it. We note $\mathscr{C}_n^{\text{fil}}$ the set of all non-isomorphic graphs with n nodes which respect to the conditions defined for a given problem.

   Each method has its advantage: selective generation allows to reach larger values for n and, as explained in Section 4.6, graphs filtering can be used to avoid re-computation of graphs and invariant values.

   Selective generation and graphs filtering are combined to define $\mathscr{C}$. For a given class $\mathscr{C}$ and a value of n, the set of graph $\mathscr{C}_n$ is defined by

$$\mathscr{C}_n = \mathscr{C}_n^{\text{sel}} \cap \mathscr{C}_n^{\text{fil}}.$$

3. *Order of graphs*: The user has to specify the values $n_{\min}$ and $n_{\max}$. The polytopes $\mathscr{P}_n$ for $n = n_{\min}, \ldots, n_{\max}$ will then be computed in a following stage. An optional value n_{step} can be defined to go from $n_{\min}$ to $n_{\max}$ by steps of n_{step}. Acceptable values of $n_{\max}$ depend of the choice made in the selective generation. Some current limits are given in Section 4.6.
4. *Options*: Several options are available allowing to adapt, or to add information in, the output generated by the software. For instance, one can get statistics about the coordinate's values, as illustrated in Section 5.2.

4.3. Data generation

If it was not already done in a previous execution of the program, the graphs of $\mathscr{C}_n^{\text{sel}}$ for $n = n_{\min}, \ldots, n_{\max}$ are generated by `geng` of McKay [42] and stored in a binary format. A graph G is stored as a set of bytes in which a bit represents a boolean value of the upper triangle of the adjacency matrix. The remaining bits are set to 0. Therefore, if G has n nodes, one uses

$$\left\lceil \frac{n(n-1)}{16} \right\rceil$$

bytes to store G. This is smaller than a 4-bytes integer if $n \leqslant 8$.

Every graph has its own *implicit reference*, which is simply its position *pos* in the file. It allows a random access to a graph G if one knows its position.

For each invariant i involved in an algebraic expression of $\mathscr{I}$ or a boolean expression used in graph filtering, its values $i(G)$ is computed for all $G \in \mathscr{C}_n^{\text{sel}}$ and stored in a binary file at the same position *pos* than the graph G. This computation will be made only once. If another problem uses again i and the same set $\mathscr{C}_n^{\text{sel}}$, the data can be directly reused. This is very useful when computing i takes a long time, as for an NP-hard invariant. Moreover, experiences show that, in the whole process, the computation of invariants is the most time-consuming step when n grows, even for invariants with a polynomial complexity (see Section 4.6).

As a graph G and its corresponding values $i_1(G), i_2(G), \ldots, i_p(G)$ are implicitly linked by their positions, one can "forget" completely that a vector of numerical values represents a graph in the next stages.

4.4. Polytopes computation

For each $n = n_{\min}, \ldots, n_{\max}$, the polytope $\mathscr{P}_n$ is computed as follows.

1. *Determine a set S of points*: A *point* of S represents one or more graphs $\in \mathscr{C}_n$. A point is internally defined by
   (a) references (positions) to graphs belonging to a set $\mathscr{G} \subseteq \mathscr{C}_n$;
   (b) the number of graphs in $\mathscr{G}$;
   (c) a vector v of coordinates $(x_1, x_2, \ldots, x_p)$ such that $x_1 = i_1(G)$, $x_2 = i_2(G)$,..., $x_p = i_p(G)$ for all $G \in \mathscr{G}$.
   The number of references stored in a point can be limited. These references are used later, e.g., to draw graphs corresponding to a given point. The number of graphs belonging to $\mathscr{G}$ is always computed.
   To determine S, GraPHedron does not need to access the graphs. It reads numerical values which are in a same position in the files containing invariant values (see Section 4.3). These values are used to determine if the conditions are respected (graph filtering) and to compute a vector v'. If a point q with a vector v of coordinates exists already in S such that $v = v'$, the set $\mathscr{G}$ of q is updated. Otherwise a new point is added in S.
   When the dimension p is small and if the vectors often have same values, e.g., integers between 0 and n, the number of points in S is generally very smaller than the number of graphs in $\mathscr{C}_n$ (see Section 4.6).
2. *Compute the convex hull of S*: This is done using `libcdd` of Fukuda [28]. Several algorithms and tools exist to compute the convex hull of a set of points. See [2] for a survey on these algorithms and an efficiency comparison of the main existing tools. Avis et al. [2] conclude that there are no algorithms that are the best in all cases. Their efficiency depends on the type of polytopes. Unfortunately, as we want to design a system which can be applied with any set of invariants, we cannot know *a priori* the characteristics of the polytopes. However, our requirements are that the convex hull computation software should be able to manage a lot of points, which are not necessarily vertices, and to deal with both exact arithmetic and floating arithmetic. This is the case of `libcdd`. Moreover, `libcdd` is a library and hence avoids calling external software.

Similar to before, if the system detects that a polytope has already been computed, it will not be re-computed a second time.

4.5. Report and conjectures

The output of GraPHedron is a report created with LaTeX. This report contains all the information computed. The polytopes $\mathscr{P}_n$ are described in both representations: a system $\mathscr{F}(\mathscr{P}_n)$ of facet defining inequalities and a set of vertices.

The user has to study the inequalities by hand or by interacting with the system and can see and/or print the vertex-graphs. In the case of two coordinates, the report contains a drawing of the polytopes. For three coordinates, the polytopes can be exported to be visualized and manipulated with *polymake* [30,31]. All the files generated to create the report are reusable and written in classical formats, e.g., EPS files for figures.

Note that when $p = 2$ and the number of facets of $\mathscr{P}_n$ is constant independently of n, the system is able to generate conjectures in a fully automated way. For instance, Conjecture 1 is automatically detected and added to the report. In Section 3.2, we used a characterization of the vertex-graphs to derive Conjecture 1. GraPHedron uses a different method. It tries to detect similarities between facets, analyzing the values of their coefficients. Then, it generalizes similar facets in terms of n. This recognition works effectively if the coefficients $f_k(n)$ are polynomials in n with a

Table 4
Needed resources for general graphs

n	# of graphs	CPU time	File size (in Kb)
5	34	0.00 s	0.17
6	156	0.00 s	0.41
7	1044	0.00 s	3.16
8	12 346	0.02 s	48.33
9	274 668	0.43 s	1341.26
10	12 005 168	18.19 s	7.034×10^4
11	1 018 997 864	27 min 14.68 s	6.966×10^6
12	1.65×10^{11}	(est.) 3.4 days	(computed) 1.451×10^9
13	5.05×10^{13}	(est.) 3.3 years	(computed) 4.932×10^{11}

Table 5
Needed resources for trees

n	# of trees	CPU time	File size (Kb)
5	3	0.00 s	0.11
10	106	0.00 s	0.72
15	7741	1.79 s	105.94
20	823 065	60 min 29.74 s	1.93×10^4

Table 6
CPU times (in second) for each step when solving Example 1

Value of n	7	8	9	10
Graph generation	0.00 (0%)	0.02 (0.2%)	0.43 (4.6%)	18.19 (4.2%)
Computation of c	0.00	0.08	1.49	78.13
Computation of D	0.01	0.12	2.92	161.57
Computation of m	0.00	0.04	0.82	42.75
Total time for invariants	0.01 (16.7%)	0.24 (45.3%)	5.23 (56.5%)	282.63 (66.0%)
Computation of $\mathscr{P}_n$	0.05 (83.3%)	0.27 (50.94%)	3.59 (38.8%)	127.23 (29.7%)
Total time	0.06	0.53	9.25	428.05

degree $\leqslant 6$. Other functions $f_k(n)$ can also be detected (see [37]). Automatization in more general cases, e.g., when the number of facets are increasing when n grows, are under study [37].

4.6. Limits and performance

Table 4 records (estimations of) the resources needed in the case of general graphs (for which the limitation today is $n = 11$). These results were obtained on a Linux computer with a Pentium (R) IV (3 GHz) processor. Of course, for more restricted classes, the manageable number n of nodes can reach larger values. For example, Table 5 refers to the class of trees.

Table 6 shows the CPU times, and relative percentages of total time, spent for each step in the computation of the problem posed in Example 1. The class of connected graphs was specified here using graph filtering. There are thus three invariants that are quite easy to compute: $m(G)$, $D(G)$ and the boolean invariant $c(G)$ which is true only if G is connected. It is thus very efficient in time to keep the data computed in files. Of course, this feature has a cost in terms of storage. Fortunately, the compression of invariant data files greatly reduces the space needed for them.

Generation of graphs and computation of invariants can be separated in a finite number of parts: either separating the computation for different values of n or even splitting computation for a given n. This allows for parallel computations and for keeping file sizes within permissible limits.

5. Applications and first results

The main application of GraPHedron is to find optimal conjectures in graph theory, as illustrated with Example 1. This example has led to Conjecture 1, which is proven in Section 5.1. A second example is also given in this section. Other possible applications will then be presented in Section 5.2.

5.1. Discover new and optimal conjectures

Several results were obtained with the help of GraPHedron [8–10,43]. The system was not only useful to find conjectures but also to write proofs. As an illustration, we now prove Conjecture 1 which can be reformulated as a theorem.

Theorem 1. *For each connected graph G with $n \geqslant 4$ nodes, m edges and a diameter D*

$$m \geqslant n - 1, \tag{6}$$

$$\frac{(n-1)(n-2)}{2} D + m \geqslant (n-1)^2, \tag{7}$$

$$D + m \leqslant \frac{n(n-1)}{2} + 1, \tag{8}$$

$$nD + 2m \leqslant (n+2)(n-1) \tag{9}$$

and these inequalities are the only possible facet defining inequalities for each n.

Proof. Inequality (6) is valid since we consider only connected graphs of order n. The minimum number of edges is thus $n-1$ which occurs only for trees. The diameter of trees varies from 2 for the star S_n to $n-1$ for the path P_n. The corresponding vectors (D, m), namely $(2, n-1)$ and $(n-1, n-1)$ are linearly independent when $n \geqslant 4$. It follows that (6) is facet defining.

If $D(G) = 1$, then $G \simeq K_n$. As $m(K_n) = \binom{n}{2}$, the two sides of (7) are equal and this inequality is valid. If $D(G) \geqslant 2$, and because $m \geqslant n-1$, we have

$$\frac{(n-1)(n-2)}{2} D + m \geqslant \frac{(n-1)(n-2)}{2} 2 + n - 1 = (n-1)^2,$$

and (7) remains valid. When $n \geqslant 3$, vectors (D, m) corresponding to K_n and S_n are linearly independent and satisfy (7) with equality. Inequality (7) is thus facet defining.

Remark that if $D < 2$, Inequality (9) is dominated by Inequality (8). It is the opposite when $D > 2$ and these two inequalities are equal when $D = 2$. We can thus assume that $D \leqslant 2$ (resp. $D \geqslant 2$) to prove that Inequality (8) (resp. (9)) is facet defining. To prove their validity, one has to answer to the following question: What is the maximum number of edges in connected graphs with fixed number of nodes and diameter?

If $D \leqslant 2$, the candidate graphs are trivially K_n and $K_n \backslash e$ which have linearly independent vectors (D, m) if only $n \geqslant 3$. Inequality (8) is thus facet defining.

Suppose now that $D \geqslant 2$. It is known that $\Delta + D \leqslant n + 1$ for any connected graph [10]. Consider two cases to prove the validity of Inequality (9) which can be rewritten as $2m \leqslant n^2 - nD + n - 2$.

(i) If $\Delta + D < n + 1$, then Inequality (9) holds because

$$2m = \sum_{i=1}^{n} d(v_i) \leqslant n\Delta \leqslant n(n-D) \leqslant n^2 - nD + n - 2,$$

if $n \geqslant 2$.

(ii) Suppose now $\Delta + D = n + 1$. In this case, it is proven [10] that any node v^* of maximum degree must be on some diameter path. Let $P = v_1, v_2, \ldots, v_{D+1}$ be a diameter path containing at least one node of degree Δ. By construction, the diameter path P contains $D + 1$ nodes. Consequently, $n - (D + 1) = \Delta - 2$ nodes do not belong

to P. We call them *exterior nodes* (relatively to path P). The extremities v_1 and v_{D+1} of the diameter path P can be adjacent to all exterior nodes but are adjacent only to one node of P. It follows that v_1 and v_{D+1} are of degree at most $\Delta - 1$. An upper bound of the number of edges is thus

$$2m = \sum_{i=1}^{n} d(v_i) \leqslant 2(\Delta - 1) + (n-2)\Delta.$$

As $\Delta + D = n + 1$, we obtain

$$2m \leqslant 2(n - D) + (n-2)(n - D + 1) = n^2 - nD + n - 2,$$

which proves the validity of (9) in this case.

The points corresponding to $K_n \backslash e$ and P_n are again linearly independent when $n \geqslant 4$. Inequality (9) is thus also an inequality defining a facet. The description of the polytope is complete. □

Example 2. What is the minimum number of edges in connected graphs with a fixed number of nodes and a fixed stability number?

This question was listed as an open problem in Ore [44] for 43 years. It constitutes a variant of a famous theorem of Turán [45], which is applied when graphs are not necessarily connected. A complete answer to Example 2, obtained with the help of GraPHedron and proved in [10], is given in the family of inequalities (12):

Theorem 2 (*Christophe et al. [10]*). *For any connected graph G with $n \geqslant 4$ nodes, m edges and stability number α*

$$m \geqslant n - 1, \tag{10}$$

$$k\alpha + m \leqslant \binom{n-k}{2} + kn \quad \text{for } k = 1, 2, \ldots, n-2, \tag{11}$$

$$m \geqslant (t(n,k) - t(n,k-1) + 1)(\alpha - k) + t(n,k) + (k-1)$$
$$\text{for } k = 2, 3, \ldots, \left\lfloor \frac{n+1}{2} \right\rfloor \text{ with } \left\lceil \frac{n}{k-1} \right\rceil \neq \left\lfloor \frac{n}{k+1} \right\rfloor + 1, \tag{12}$$

where

$$t(n,k) = \left(\left\lceil \frac{n}{k} \right\rceil - 1 \right) \cdot \left(n - \frac{k}{2} \left\lceil \frac{n}{k} \right\rceil \right).$$

These inequalities are the only possible facet defining inequalities for each n.

Note that in this case, the number of facets is no more constant as (11) and (12) are families of inequalities for a parameter k. A more readable answer to Example 2, obtained with the characterization of vertex-graphs, is given in [10] and will be recalled in Section 6.

Other illustrations of this strategy to obtain new results in graph theory can be found in [8–10,43].

5.2. Other applications

Here are some other applications of GraPHedron:

(a) *Check existing conjectures and theorems*. One can submit to GraPHedron an existing conjecture C or theorem T expressed as an inequality (or an equality) among p invariants. The system can then be used in two ways:
   (1) *Check for validity and optimality*. This can be done without computing the polytopes. Browsing the graphs of $\mathscr{C}_n$, one can check if:
      - C is *rejected* since a counter-example exists;
      - C is *valid*;
      - C or T is *tight* by exhibiting a graph satisfying the conjecture or theorem with equality;

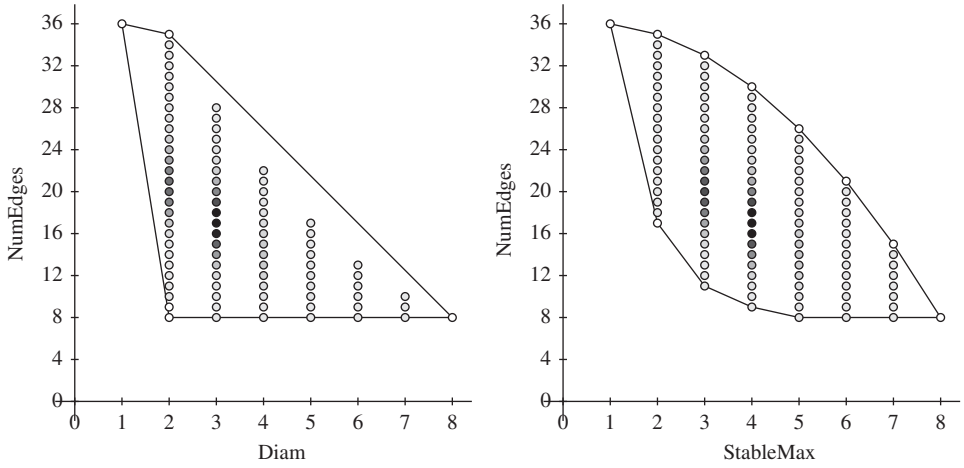

Fig. 5. Distribution of points in $\mathscr{P}_9$ for Examples 1 and 2.

- C or T is *facet defining* by exhibiting a set of p graphs, satisfying the conjecture or theorem with equality, and corresponding to p linearly independent vectors of coordinates.

(2) *Extract invariants and compute associated polytopes*. A conjecture or a theorem which is not facet defining in all polytopes means that one can improve it. Applying the strategy described in Section 3 can lead to a better formulation. Otherwise, showing that an existing theorem is facet defining is a strong criterion of quality.

(b) *Help for theorem-proving*. Because there are extremals for a given problem, the knowledge of the vertex-graphs is of great help when one has to prove the conjectures derived with or by the system. Moreover, if their coordinates are characterized, one gets directly a set of linearly independent vectors. This is intensively used in the proofs, see for example the proof of Theorem 1.
The representation of the polytopes is useful too when one has to detect if an inequality is dominated by another.

(c) *Compare values of heuristics and exact algorithms*. Let $h(G)$ be the value obtained when a heuristic H is applied to G. If the heuristic H contains no random choice, $h(G)$ is unique for each graph and can thus be considered as an invariant. Otherwise, the arithmetic average of all possible values given by H for a graph G is also an invariant. Such an invariant, divided by the value obtained by an exact algorithm, is again an invariant representing a factor of approximation. To illustrate, the maximal-matching heuristic provides a 2-approximation of three classical NP-hard problems (*minimum vertex cover*, *minimum maximal matching* and *minimum edge dominating set*) [29]. Using GraPHedron, finer worst-case approximation factors were obtained in [8], under some assumptions on the density of the graphs.

(d) *Education*. The system is easy to use and some well-chosen problems, as Example 1, can be used with students. Information is displayed in a way that makes basic notions of graph theory and polyhedral theory handy to illustrate. Of course, the proofs of conjectures—which can appear quite easily—is another challenge. This provides an exciting way to be initiated to the world of research in graph theory.

(e) *Get information about the points distribution*. One can ask, as an option, that the system adds the distribution of the points, corresponding to graphs, *inside* the polytopes. It can be of interest as sometimes a vast majority of graphs are far away from some facets. Fig. 5 shows the distribution inside the polytope $\mathscr{P}_9$ derived from Examples 1 (left) and 2 (right). A white point means that there is only one graph corresponding to this coordinate. An increasing level of gray shows the growing frequency of graphs sharing the same coordinates.
In Example 1, some graphs correspond to points which are *Pareto* optimal. GraPHedron was able to detect these points and made the following conjecture automatically:

Conjecture 2. *For each connected graph G with $n \geqslant 3$ nodes, m edges and a diameter $2 \leqslant D \leqslant n-1$*

$$2m \leqslant D^2 - D(2n+1) + (n-1)(n+4). \tag{13}$$

Note that (13) is dominated by the facet defining inequality (9).

(f) *Get information about invariants*. One can ask, as an option, that the system computes, during its process, various statistics about the coordinates of the problem.

6. Forms of conjectures

When n is fixed, facets are linear inequalities among p invariants. A generalization of these inequalities in terms of n leads to a first basic form of conjectures:

$$f_1(n)i_1(G) + f_2(n)i_2(G) + \cdots + f_p(n)i_p(G) \leqslant f_0(n), \tag{14}$$

where $f_i(n)$ are functions, linear or not, of n. Theorem 1 is an example of such conjectures where coefficients $f_i(n)$ are linear or quadratic functions. It follows that (14) is linear for each n, i.e., when n is fixed, but can be non-linear when n is taken as a parameter.

Conjecture 2 shows that even when n is fixed, the system can lead to inequalities that are not linear, when one consider points that are *Pareto* optimal.

A third situation appears with Example 2. This shows the existence of families of inequalities, e.g., the $n-2$ facets described in (11). It gives $n-2$ inequalities of the form (14)

$$m \leqslant \binom{n-k}{2} + k(n-\alpha) \quad \text{for } k = 1, 2, \ldots, n-2. \tag{15}$$

When families of inequalities occur to describe the polytopes $\mathscr{P}_n$, the vertex-graphs of $\mathscr{P}_n$ are often also of a specific family. For instance, graphs which are vertex-graphs of the polytopes described by (15) are *complete split graphs*. A complete split graph $CS(n, \alpha)$ with $1 \leqslant \alpha \leqslant n$ is constructed from an independent set of size α and a clique of size $n-\alpha$. Each vertex in the independent set is adjacent to each vertex in the clique. By construction,

$$m(CS(n, \alpha)) = \binom{n-\alpha}{2} + (n-\alpha)\alpha = \frac{1}{2}[n(n-1) + \alpha(\alpha-1)]. \tag{16}$$

There is exactly one such graph for each possible values of α, i.e., for the integers $\alpha = 1, 2, \ldots, n$. It follows that the family of inequalities (15) can be reformulated, without loss of generality, as

$$m \leqslant \tfrac{1}{2}[n(n-1) + \alpha(\alpha-1)], \tag{17}$$

which is non-linear in α, even if n is fixed. Fig. 6 shows the representations in the plane of (15), the black lines, and (17), the dotted curve, when $n = 5$.

The family of facets expressed in (11), which is the answer to Example 2, can also be reformulated, without loss of generality [10], as

$$m \geqslant \left(\left\lceil \frac{n}{\alpha} \right\rceil - 1\right) \cdot \left(n - \frac{\alpha}{2} \left\lceil \frac{n}{\alpha} \right\rceil\right) + \alpha - 1. \tag{18}$$

In this case the function is not polynomial as the ceiling operator is used.

Therefore, forms of conjectures that can be derived with or by GraPHedron are not limited to the form (14).

7. Concluding remarks

We have presented a formal framework, supplied by an automated tool, allowing to identify facet defining inequalities among graph invariants. More precisely, the following three points are together the specificity of the system GraPHedron:

(a) First, we associate a notion of "optimality of a conjecture". Authors often argue that a bound is *tight*, which means only that the inequality defines a supporting hyperplane. We propose to consider facet defining inequalities as a criterion of conjecture's quality.

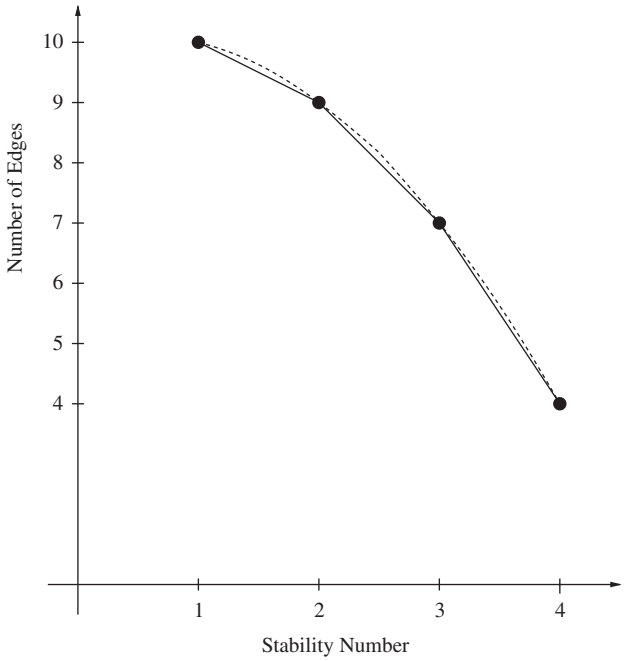

Fig. 6. Family of inequalities (15) and its quadratic generalization (17), when $n = 5$.

(b) Moreover, the polyhedral approach allows to achieve a *complete* study of the relations among the selected set of graph invariants. It is not only a lower and/or an upper bound on a relation, it is a minimal system of linear inequalities describing the relations among the set of invariants.
(c) Finally, the output of the system is not only a list of conjectures. For a given problem, the system gives additional geometrical information which can be correlated to graph theoretical interpretation. The system computes also graphs that are fundamental to the problem: the *vertex-graphs*. All this information is useful in the proofs.

The proposed approach can also be used to check existing conjectures and theorems, to help for proving conjectures, to study approximation ratios, in education, to get the spatial distribution of the graphs inside the polytopes and to get statistics about invariants.

To avoid re-computation of graphs and invariants, the system is quite greedy in storage. Therefore, we think that a web portal, on a dedicated server, allowing to use GraPHedron with much pre-computed data, is preferable to a downloadable version of the software. This is the purpose of the web site www.graphedron.net [33], available since January 2007.

Acknowledgments

The author thanks M. Labbé and P. Hansen for discussions about GraPHedron. The first idea of this system came from such discussions. He also thanks V. Bruyère, T. Mens and the anonymous referees for detailed comments that helped greatly to improve the paper's presentation. Finally, he thanks the members of the Computer Science and Mathematics Departments of Université Libre de Bruxelles, who used GraPHedron and gave useful feedback.

References

[1] M. Aouchiche, G. Caporossi, P. Hansen, Variable neighborhood search for extremal graphs 8. Variations on Graffiti 105, Congr. Numer. 148 (2001) 129–144.
[2] D. Avis, D. Bremner, R. Seidel, How good are convex hull algorithms, Comput. Geom. 7 (1997) 265–301.
[3] C. Berge, Graphes et Hypergraphes, Gauthier-Villars, Paris, 1983.

[4] G. Caporossi, D. Cvetković, I. Gutman, P. Hansen, Variable neighborhood search for extremal graphs 2. Finding graphs with extremal energy, J. Chem. Inform. Comput. Sci. 39 (1999) 984–996.
[5] G. Caporossi, I. Gutman, P. Hansen, Variable neighborhood search for extremal graphs 4. Chemical trees with extremal connectivity index, Comput. Chem. 23 (1999) 469–477.
[6] G. Caporossi, P. Hansen, Variable neighborhood search for extremal graphs 1. The AutoGraphiX system, Discrete Math. 212 (2000) 29–44.
[7] G. Caporossi, P. Hansen, Variable neighborhood search for extremal graphs 5. Three ways to automate finding conjectures, Discrete Math. 276 (2004) 81–94.
[8] J. Cardinal, M. Labbé, S. Langerman, E. Levy, H. Mélot, A tight analysis of the maximal matching heuristic, in: Computing and Combinatorics: 11th Annual International Conference, COCOON 2005, Kunming, China, Lecture Notes in Computer Science, vol. 3595, Springer, Berlin, 2005, pp. 701–709.
[9] J. Cardinal, S. Langerman, E. Levy, Improved approximation bounds for edge dominating set in dense graphs. in: Proceedings of Workshop on Approximation and Online Algorithms (WAOA), Lecture Notes in Computer Science, vol. 4368, Springer, Berlin, 2006, pp. 108–120.
[10] J. Christophe, S. Dewez, J.-P. Doignon, S. Elloumi, G. Fasbender, P. Grégoire, D. Huygens, M. Labbé, H. Mélot, H. Yaman, Linear inequalities among graph invariants: using GraPHedron to uncover optimal relationships, submitted for publication.
[11] D. Cvetković, Discussing graph theory with a computer, II: theorems suggested by the computer, Publ. Inst. Math. (Beograd) 33 (47) (1983) 29–33.
[12] D. Cvetković, Discussing graph theory with a computer, IV: knowledge organisation and examples of theorem proving, in: Proceedings of the Fourth Yugoslav Seminar on Graph Theory, Novi Sad, 1983, pp. 43–68.
[13] D. Cvetković, Discussing graph theory with a computer, VI: theorems proved with the aid of the computer, Cl. Sci. Math. Natur., Sci. Math. T. XCVII (16) (1988) 51–70.
[14] D. Cvetković, L. Kraus, S. Simić, Discussing graph theory with a computer, I: implementation of graph theoretic algorithms, Univ. Beograd Publ. Elektrotehn. Fak, Ser. Mat. Fiz. No. 716–734 (1981) 100–104.
[15] D. Cvetković, I. Pevac, Discussing graph theory with a computer, III: man–machine theorem proving, Publ. Inst. Math. (Beograd) 34 (48) (1983) 37–47.
[16] D. Cvetković, S. Simić, Graph theoretical results obtained by the support of the expert system "graph", Bull. Acad. Serbe Sci. Arts Cl. Sci. Math. Natur. 19 (1994) 19–41.
[17] D. Cvetković, S. Simić, G. Caporossi, P. Hansen, Variable neighborhood search for extremal graphs 3. On the largest eigenvalue of color-constrained trees, Linear and Multilinear Algebra 2 (2001) 143–160.
[18] E. De la Vina, Bibliography on conjectures of Graffitti. Available at ⟨http://cms.dt.uh.edu/faculty/delavinae/research/wowref.htm⟩, 2000.
[19] R. Diestel, Graph Theory, second ed., Springer, Berlin, 2000.
[20] S. Fajtlowicz, Written on the wall. A regularly updated file accessible from ⟨http://www.math.uh.edu/∼clarson/⟩.
[21] S. Fajtlowicz, On conjectures of Graffiti—II, Congr. Numer. 60 (1987) 187–197.
[22] S. Fajtlowicz, On conjectures of Graffiti, Discrete Math. 72 (1988) 113–118.
[23] S. Fajtlowicz, On conjectures of Graffiti—III, Congr. Numer. 66 (1988) 23–32.
[24] S. Fajtlowicz, On conjectures of Graffiti—IV, Congr. Numer. 70 (1990) 231–240.
[25] S. Fajtlowicz, On conjectures of Graffiti—V, Seventh International Quadrennial Conference on Graph Theory, vol. 1, 1995, pp. 367–376.
[26] S. Fajtlowicz, W.A. Waller, On two conjectures of Graffiti, Congr. Numer. 55 (1986) 51–56.
[27] P.W. Fowler, P. Hansen, G. Caporossi, A. Soncini, Polyenes with Maximum HOMO-LUMO Gap. (Variable neighborhood search for extremal graphs 7), Chem. Phys. Lett. 342 (2001) 105–112.
[28] K. Fukuda, cdd/cdd+ reference manual. cdd ver. 0.61, cdd+ ver. 0.76, 1999, available at ⟨http://www.cs.mcgill.ca/∼fukuda/soft/cddman/cddman.html⟩.
[29] M.R. Garey, D.S. Johnson, Computers and Intractability. A Guide to the Theory of NP-completeness, Freeman and Company, New York, 1979.
[30] E. Gawrilow, M. Joswig, Polymake: a framework for analyzing convex polytopes, in: G. Kalai, G.M. Ziegler (Eds.), Polytopes—Combinatorics and Computation, Birkhäuser, Basel, 2000, pp. 43–74.
[31] E. Gawrilow, M. Joswig, Polymake: an approach to modular software design in computational geometry, in: Proceedings of the 17th Annual Symposium on Computational Geometry, ACM, Medford, MA, June 3–5, 2001, pp. 222–231.
[32] GMP. GNU multiple precision arithmetic library, Homepage: ⟨http://www.swox.com/gmp/⟩.
[33] GraPHedron. Web portal at ⟨http://www.graphedron.net⟩.
[34] I. Gutman, P. Hansen, H. Mélot, Variable neighborhood search for extremal graphs 10. Comparison of irregularity indices for chemical trees, J. Chem. Inform. Modeling 45 (2005) 222–230.
[35] P. Hansen, How far should, is and could be conjecture-making automated in graph theory?, in: S. Fajtlowicz et al. (Ed.), Graphs and Discovery, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 69, American Mathematical Society, Providence, 2005, pp. 189–230.
[36] P. Hansen, M. Aouchiche, G. Caporossi, H. Mélot, D. Stevanović, What forms do interesting conjectures have in graph theory?, in: S. Fajtlowicz et al. (Ed.), Graphs and Discovery, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 69, American Mathematical Society, Providence, 2005, pp. 231–252.
[37] P. Hansen, M. Labbé, H. Mélot, Automatization of conjecture-making in GraPHedron, 2006, in preparation.
[38] P. Hansen, H. Mélot, Computers and discovery in algebraic graph theory, Linear Algebra Appl. 356 (2002) 211–230.
[39] P. Hansen, H. Mélot, Variable neighborhood search for extremal graphs 6. Analysing bounds for the connectivity index, J. Chem. Inform. Comput. Sci. 43 (2003) 1–14.
[40] P. Hansen, H. Mélot, Variable neighborhood search for extremal graphs 9. Bounding the irregularity of a graph, in: S. Fajtlowicz et al. (Ed.), Graphs and Discovery, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 69, American Mathematical Society, Providence, 2005, pp. 253–264.

[41] P. Hansen, H. Mélot, I. Gutman, Variable neighborhood search for extremal graphs 12. A note on the variance of bounded degrees in graphs, MATCH Comm. Math. Comput. Chem. 54 (2005) 221–232.
[42] B.D. McKay, Nauty user's guide (version 1.5), Technical Report, Department of Computer Science, Australian National University, 1990.
[43] H. Mélot, On automated and computer aided conjectures in graph theory, Ph.D. Thesis, Université de Mons-Hainaut, 2006.
[44] O. Ore, Theory of Graphs, American Mathematical Society Colloquium Publications, vol. XXXVIII, American Mathematical Society, Providence, RI, 1962.
[45] P. Turán, Eine Extremalaufgabe aus der Graphentheorie, Mat. Fiz. Lapok 48 (1941) 436–452.
[46] G.M. Ziegler, Lectures on Polytopes, revised ed., Springer, Berlin, 1998.