GRAFFITI.PC

Ermelinda DeLaVina

Department of Computer and Mathematical Sciences
University of Houston-Downtown
Houston, Texas 77002
<delavinae@uhd.edu>

1. Introduction

Graffiti.pc is a conjecture-making computer program whose design was strongly influenced by the design of the conjecture-making program Graffiti. The program Graffiti was created in the mid 1980's by Siemion Fajtlowicz of the University of Houston. As his student in the early 1990's, I contributed to the development of the most recent versions of Graffiti **[3]**. While that experience strongly influenced the design of the new program, it is appropriate to note that the initial goals for the creation of the respective programs were distinct. A main goal in the creation of Graffiti.pc was to have a user-friendly PC platform Graffiti-like program, which my undergraduate students could utilize. Whereas almost from the onset of Graffiti's creation, Fajtlowicz was announcing conjectures to other researchers, but did not design it with other users in mind until very recently **[4]**. Consequently, any comparison of the programs in this paper is intended only as a point of reference. In the summer of 2001, Graffiti.pc's creation was realized and this is the program under discussion in this paper. In particular, the focus will be a description of the program followed by a description of an educational application.

2. Program Description

The three integral components of Graffiti.pc consist of the C++ programs BuildDbs, dalmatians, and the Visual Basic user interface. By comparison to Graffiti's database support program Algernon, Graffiti.pc's BuildDbs is limited in the number of graph theoretical invariants that it computes. Currently, it has the capability of generating about one hundred and thirty graph theoretical values, most of which are on degree and distance invariants of a graph, its complement graph and its second power graph. However, the dalmatians program is similar to the dalmatians heuristic as described briefly below and more thoroughly in **[3]**. Overall, the most striking difference is Graffiti.pc's graphical user interface, since Graffiti is a UNIX platform, menu and file driven program. Moreover, while both programs allow the user to select the relation, invariants and graphs for the database, Graffiti.pc allows a user control over the algebraic operations used to generate expressions that result in conjectures.

Dalmatians accepts as inputs a database (2-dimensional array indexed by models and invariants), a fixed term, and a relation (inequality or equality). Given only this input, the dalmatians program generates conjectures. During or after its execution the user may view conjectures, submit counterexample(s), and choose to re-execute the dalmatians program; at this time, Graffiti.pc, unlike Graffiti, does not have an option for resuming the generation of conjectures. Moreover, as is the case with Graffiti, the program does not prove conjectures. This is still a human function.

2.1. The Dalmatian Heuristic

The dalmatians heuristic as described by Fajtlowicz **[3]** is as follows "the program keeps track of conjectures made in the past and when it runs across a new candidate for a conjecture then first of all it verifies if there is an example (in the database) demonstrating that the conjecture does not follow from the previous conjectures. If there is no such example then the conjecture is rejected as non-informative. If there is one, then the program proceeds with testing the correctness of the conjecture, and finally it verifies whether the conjecture should be rejected by one of its other heuristics. If the conjecture is accepted by the program then the list of conjectures is revised and those conjectures which are less informative than the new one are removed from the list and stored separately in the case the new conjecture will be refuted in the future".

Similarly as in Graffiti, for each conjecture, selected by the program to appear on its list of reported conjectures, the number of models in the database for which the relation, between the selected term and conjectured bound, is actually equality is called the *touch number* of the conjecture.

The implementation of the dalmatians heuristic by Graffiti.pc begins in the same manner as described by Fajtlowicz. The first step in which it differs is that before testing the correctness of a conjecture relative to the database, it first verifies if the touch number is at least the user-specified minimum touch number. Further, at the present time, storing removed conjectures is not an option. The next step at which the dalmatians implementations differ is in the removal of conjectures. In Graffiti.pc, a variant of the Irin heuristic **[3]** is used first to remove conjectures if they follow by transitivity from the new conjecture. Note that in Fajtlowicz's description of dalmatians the removal of such conjectures is accomplished in any case, however for Graffiti.pc the decision was made to seize the opportunity to report such relations. The results are stored separately as they are not a part of the dalmatians list of conjectures. An example of such a conjecture is described in the next section.

The dalmatians program stops if and only if for every graph G in the database there exists a conjecture on the list whose touch number was contributed to by the graph. Thus in addition to providing a list of conjectured bounds, say c_1, $c_2, \ldots, c_k$, for a user-selected term, say x, the entire list is interpreted as the following conjecture. For the sake of example let us assume that the c_i are lower bounds on x that is for every i, $x \geq c_i$.

For every graph G in the class of graphs represented in the database,

$$x \geq \text{maximum of } \{c_1, c_2, \ldots, c_k\}.$$

The interpretation and implication for such a conjecture and for the program's inability (in a reasonable amount of time) to make such a conjecture is beyond the scope of this paper, but is discussed in **[5]**.

2.2. Form of Conjectures

Likewise as in Graffiti, conjectures are inequalities between terms of a **D**-Algebra, on the set of invariants in the database, together with binary, and unary operations. In Graffiti.pc a term is represented as a *syntax tree*, that is, a tree in which each node represents an operator and the children of the node represent the operands. At present, Graffiti.pc provides fourteen unary operations and five binary operations. Examples of such operations are the reciprocal, the natural logarithm, ceiling, addition, and multiplication.

Below are three conjectures for trees of highest touch number (each greater than 30% of the size of the model set) generated by Graffiti.pc. The fourth conjecture is a product of the Irin heuristic implemented by the program as described previously. They are all correct; the first two are easily proven and the last two are a bit more challenging as exercises. Further the bound given in the third conjecture is valid for any graph, which leads to the question of where is the Echo heuristic in Graffiti.pc? At present it is not implemented in Graffiti.pc. For the listed conjectures, the program parameters were set as follows. The fixed term was the path covering number, the relation was greater than or equal, minimum touch was fifty and maximum percentage of undefined was fifteen percent. The model set was comprised of all trees on fewer than twelve vertices, as generated by Brendan McKay's program *makeg* **[6]**, and a hodgepodge of 26 other trees. The invariant set was comprised of 68 of the available invariants.

The *path covering number of a graph*, denoted by $\pi(G)$, is the minimum number of vertex disjoint paths needed to cover the vertices of the graph. The *number of leaves of a tree* is the number of vertices of degree one. We put $\Delta(G)$ to be the maximum degree of a graph G.

Conjecture 1. If the graph G is a tree, then $\pi(G) \geq \Delta(G) - 1$.

Conjecture 2. If the graph G is a tree, then $\pi(G) \geq \left\lceil \frac{\text{number of leaves}}{2} \right\rceil$.

Conjecture 3. If the graph K is a tree, then **3**ÐKÑ is greater than or equal to twice the independence number of K minus the number of vertices of K.

Let lÐ@Ñ be the number of vertices at even distance from vertex @. We put *max of even distance* to be the maximum of lÐ@Ñ over all vertices of the graph, and similarly we put the *min of even distance* to be the minimum of lÐ@Ñ over all vertices of the graph.

Conjecture 4. If the graph K is a tree, then twice the independence number of K minus the number of vertices of K is greater than or equal to max of even distance minus min of even distance.

The main objective of listing the conjectures in this paper was to demonstrate the algebraic form of conjectures. We observe that it seems that the main difference between an educational and research version of the program is the simplicity of the invariant set. Thus, given Graffiti.pc's limited invariant set, only conjectures for which the dalmatians program reported a significantly high touch number were reported. From experience, as was the case in this execution of the program, it seems that conjectures of high touch number (relative to the model set) are usually correct.

On a technical note, the first two conjectures appeared almost immediately and a while later the program reported that the path covering number is not less than max of even distance minus min of even distance. Several hours later the program replaced the conjecture on even distance by conjecture 3. The dalmatians program did not stop naturally; when I stopped the program (after it ran for a day) there were 120 trees in the database that did not contribute to the touch number of any conjecture on the list.

3. Educational Application

Thus far Graffiti.pc's primarily application was educational (which is not surprising considering my goal). In particular, my undergraduate students Barbara Chervenka and Kelly Wroblewski have used the program's conjectures as the topic of their respective Senior Projects. As Barbara's project was completed in December 2001, her activities and results will be described. The first phase of Barbara's project was to resolve conjectures, which were lower bounds on the sum of the independence number and the clique number of a graph. Courtesy of Siemion Fajtlowicz and the University of Houston Mathematics Department, conjectures of this phase of her project were generated by Graffiti on that campus' alpha computers.

At about the time that we opted to change topics for conjectures, which was also about the same time that Graffiti.pc was created, Siemion Fajtlowicz announced a set of rules to follow while working on conjectures. The rules are called the Red Burton rules **[4]**. With the previously mentioned changes in place, Barbara began the second phase of her senior project. The input for the program was the database, which was composed of the complete graph on one vertex as the model set, and the alpha-core number and invariants of the degree sequence of a graph as the invariant set. The fixed term was the alpha-core number, and the relation was greater than or equal. We opted for a modification of the Red Burton rules, which are described below.

1. The first conjecture to appear on the list will be resolved. (Note that in Graffiti.pc, if the conjectures remain unsorted by touch number then it is usually the case that the first is the most simply stated conjecture).

2. If the resolved conjecture is false then find the minimum number of vertices in a counterexample, and next the minimum number of edges of a counterexample with the minimum number of vertices. In this case the counterexample is added to the database.

3. If the resolved conjecture is true then characterize the case of equality and determine if one can verify in polynomial time that a graph has the characterization described. In the case such a characterization is accomplished, graphs from the class are forbidden from the database; further, any counterexamples for subsequent conjectures could not be in this class of graphs. Otherwise, the next conjecture on the list is resolved.

After about two months into the second phase of her project, Barbara's partial result was a characterization of the alpha-core number, in terms of concepts involving only the degree sequence of a graph, for the class of graphs comprised of stars, complete graphs, complete graphs minus an edge, complete graphs minus a triangle, and complete graphs minus a triangle and minus an edge disjoint from the triangle. The result is stated as follows.

Let K be a simple graph, I ÐKÑ its set of edges, and $\overline{\text{K}}$ the complement graph of K. We let O 7 denote a complete graph on 7 vertices, H 7 denote the 7-vertex graph with no edges, and O$_{3,2}$ the complete bipartite graph with the parts having $ and # vertices. The *alpha-core number of* K, denoted **!**-ÐKÑ, was define to be the cardinality of the intersection of all maximum independent sets of the graph K. The *length of a graph* K was defined as the square root of the sum of the squares of degrees of the vertices of K.

Theorem (Chervenka [1]): If K is a simple connected graph, then

Ú ¸ I ÐKÑ¸ if K ¶ joinÐO" ß H 7Ñ for 7 #
#¸ I Ð$\overline{\text{K}}$Ñ¸ if K ¶ O 7 or joinÐO 7 ß H$_2$Ñ for 7 #
!-ÐKÑ œ Û ÚLength($\overline{\text{K}}$ÑÛ if K ¶ joinÐO 7 ß H$_3$Ñ for 7 # or
joinÐO 7 ß O$_{3,2}$Ñ for 7 0
Ü **?**

At the end of her project the pending list, which was list 11, of conjectures for simple connected graphs was as follows:

Conjecture. If K is not isomorphic to O 7, joinÐO 7 ß H$_2$Ñ for 7 #, joinÐO" ß H 7 Ñ for 7 #, joinÐO 7 ß H$_3$Ñ for 7 # nor joinÐO 7 ß O$_{3,2}$Ñ for 7 0, then

!-ÐKÑ Ÿ " € *2nd smallest element in the set of degrees of* K.

!-ÐKÑ Ÿ " € *the maximum degree of the complement of* K.

!-ÐKÑ Ÿ #‡Ð*the frequency of the maximum degree of* K*).*

Formally, the project had to come to an end. Nevertheless, Barbara determined that the first and last conjectures are false and the second is true. The next step called for the determination of a smallest counterexample to the first statement.

During the first phase of her senior project, the goal of working on conjectures was simply to resolve as many as possible. However, through some appropriate questions the program's response to counterexamples was emphasized. As a result, Barbara eventually learned to look for families of counterexamples, special classes of graphs on which the conjecture may be true, and to characterize the case of equality for a proven conjecture. Moreover, in the early phase of the project she was encouraged to find counterexamples on the smallest possible number of vertices and then the smallest number of edges on that number of vertices. Early on, one motivation for this was to provide statements relevant to conjectures, that required proof, but eventually in the second phase of the project this skill was compulsory. By the end of the project, Barbara had examined, in varying degrees, over 80 conjectures, and almost half were resolved. The chronology and details of which are described in her senior project report **[1]**. Aside from the obvious difference of the selected term, the first phase and second phase of her project differed in that the conjectures generated by Graffiti were announced by me, as I was the user, whereas by list 3 of the second phase, Barbara was reporting the conjectures to me as she was the user of Graffiti.pc.

In addition to providing the previously cited student research opportunities, another advantage of using Graffiti.pc (and Graffiti) as a pedagogic tool was that, by the nature of how the programs were utilized, the difficulty level of conjectures increased as the students' knowledge and the number of graphs in the database increased. Further, as an educator, the abundance of good problems accessible to students, even undergraduate students, was stimulating. But the potential is even greater as Fajtlowicz observed in **[4]**, "If the students wish to, they may, run the program according to their own rules or simply by working on conjectures of their own choice, ending up with highly personalized exercises and problems".

References

[1] B. Chervenka; Graph Theory Graffiti/Graffiti.pc Style, Senior Project Report, University of Houston-Downtown, Houston, Texas 77002 (2001).

[2] S. Fajtlowicz; On Conjectures of Graffiti V, Proceedings of the Seventh Quadrennial International Conference on the Theory and Applications of Graphs, 1, 367-376 (1995).

[3] S. Fajtlowicz; On Conjectures of Graffiti III, Congressus Numerantium, 66, 23-32 (1988).

[4] S. Fajtlowicz; Toward Fully Automated Fragments of Graph Theory, Preprint.

[5] S. Fajtlowicz; Written on the Wall, A list of Conjectures of Graffiti accessible at http://www.math.uh.edu/~siemion.

[6] B. McKay; *makeg*, a computer program accessible at http://www.theory.csc.uvic.ca/~cos/gen/grap.html.