Clique and Coloring Problems
A Brief Introduction, with Project Ideas

Last revision: December 2, 1992

This a work in progress, intended as a quick introduction to the clique and coloring problems, not an exhaustive survey. (More complete surveys and bibliographies are listed in the references, some of them available via anonymous FTP from `dimacs.rutgers.edu`.) *Here we restrict our attention to key references (the set of which may change as the document evolves). Feel free to send us any suggested additions or clarifications, or additional project suggestions. The email address for suggestions is* `challenge@dimacs.rutgers.edu`.

1 Introduction

Given an undirected graph $G = (V, E)$, a *clique* of the graph is a set of mutually adjacent vertices. A *maximum clique* is, naturally, a clique whose number of vertices is at least as large as that for any other clique in the graph. If the vertices have weights then a *maximum weighted clique* is a clique with the largest possible sum of vertex weights.

A (vertex) *coloring* of an undirected graph is an assignment of a label to each node. It is required that the labels on the pair of nodes incident to any edge be different. A *minimum coloring* of a graph is a coloring that uses as few different labels as possible.

Clique and coloring problems are very closely related. It is straightforward to see that the size of the maximum clique is a lower bound on the minimum number of labels needed to color a graph.

Many problems of practical interest can be modeled as clique and coloring problems. The general form of these applications involves forming a graph with nodes representing items of interest. An edge connects two "incompatible" items. The maximum clique problem is then to find as large a set of pairwise incompatible items as possible. The minimum coloring problem is to assign a color to each item so that every incompatible pair is assigned different colors.

Both of these problems are formally NP–hard for general graphs [41]. It therefore seems unlikely that it will be possible to find a fast (i.e. polynomial-time) algorithm to solve these problems. In fact, based on the results of [35, 6, 5, 66] it seems unlikely that it is even possible to find an approximate solution to these problems quickly: Assuming P $\neq$ NP, there is an $\epsilon > 0$ such that no polynomial time approximation algorithm for either problem can find a solution that is guaranteed to be within a ratio of $|V|^\epsilon$ of optimal.

Despite these negative results, it is still necessary to find solutions to large clique and coloring problems. The applications do not go away just because the problem is intractable in the worst case. Therefore, it is important for us to determine how difficult clique problems are to solve in practice. Do hard instances, or at least hard–to–approximate instances, only occur rarely? What structural properties of instances make it difficult to solve these problems, and how can those structures be avoided? The purpose of this note (which is currently an evolving document) is to outline some past research on solving clique and coloring problems and to identify some possible research directions.

In the next section, we discuss a few typical applications of the maximum clique and minimum coloring problems. In section 3, we discuss several solution approaches that have been tried in the past. In section 4 we talk about some suggested research directions.

2 Sample Applications

Both coloring and clique models are useful in a variety of applications. In this section we discuss a few typical applications, first for coloring and then for clique (although note that since the two problems provide bounds for each other, both problems may well be applicable in several of these domains). A more extensive survey of applications for clique can be found in [48] and is available via anonymous FTP from DIMACS.

2.1 Time Tabling and Scheduling

Many scheduling problems involve allowing for a number of pairwise restrictions on which jobs can be done simultaneously. For instance, in attempting to schedule classes at a university, two courses taught by the same faculty member cannot be scheduled for the same time slot. Similarly, two course that are required by the same group of students also should not conflict. The problem of determining the minimum number of time slots needed subject to these restrictions is a graph coloring problem. This problems has been studied by many researchers, including Leighton [64], Opsu and Roberts [74], and de Werra [33].

2.2 Frequency Assignment

Gamst [38] examines a problem in assigning frequencies to mobile radios and other users of the electromagnetic spectrum. In the simplest case, two customers that are sufficiently close must be assigned different frequencies, while those that are distant can share frequencies. The problem of minimizing the number of frequencies is then a graph coloring problem.

If one considers situations where users (vertices) enter and leave the system, this application also provides a setting where on–line algorithms would be relevant.

2.3 Register Allocation

One very active application for graph coloring is register allocation. The register allocation problem is to assign variables to a limited number of hardware registers during program execution. Variables in registers can be accessed much quicker than those not in registers. Typically, however, there are far more variables than registers so it is necessary to assign multiple variables to registers. Variables conflict with each other if one is used both before and after the other within a short period of time (for instance, within a subroutine). The goal is to assign variables that do not conflict so as to minimize the use of non–register memory.

A simple approach to this is to create a graph where the nodes represent variables and an edge represents conflict between its nodes. A coloring is then a conflict–free assignment. If the number of colors used is less than the number of registers then a conflict–free register assignment is possible. Some papers that outline and expand on this method include Chaitin [24], Chaitin et. al [25], Chow and Hennessy [27, 28], and Briggs, Cooper, Kennedy, and Torczon [19].

2.4 Printed Circuit Board Testing I – Coloring

Garey, Johnson, and So [39] consider the problem of testing printed circuit boards for unintended short circuits (caused by stray lines of solder). This gives rise to a graph coloring problem in which the vertices correspond to the nets on board and there is an edge between two vertices if there is a potential for a short circuit between the corresponding nets. Coloring the graph corresponds to partitioning the nets into "supernets," where the nets in each supernet can be simultaneously tested for shorts against all other nets, thereby speeding up the testing process.

2.5 Printed Circuit Board Testing II – Clique

A printed circuit board tester involves placing probes onto a board. A probe can determine if a portion of a board is working correctly. Since probes have a particular size, not every component can be checked in one pass. The problem of maximizing the number of components checked in one pass can be formulated as a clique problem: each node represents a component and an edge represents two nodes that are too close to be checked simultaneously. A clique in this graph is then a set of components that can be checked in one pass. This is an example of a *geometric* clique problem and has been studied by Yu, Goldschmidt and Chen [86] and Yu. Kouvelis, and Luo [87].

2.6 Pattern Matching

Ogawa [73] has an interesting application involving pattern recognition. Given a "target" picture and an input picture (which involve only a set of points), a related compatibility graph is created whose vertices correspond to pairs of points. There is an edge between two vertices if the corresponding pairs are "mutually consistent" (where this can depend on a variety of restrictions, including angular relationships as well as the requirement that no point be matched with more than one other). A large clique represents a large number of mutually consistent pairs, and its size can be used as a measure of the corresponding fit. This model seems to correctly recognize affine transformations as well as moderately nonlinear transformations.

2.7 Analysis of Biological and Archeological Data

In biology and archeology, a standard model for relating objects is that of a tree. Trees can represent the division of a species into two separate species or the division of features of some artifact (like pottery or pins). Species do not come with histories, however, nor are artifacts completely dated. Therefore, it is necessary to deduce the tree structure from the features of the items.

One approach to this is to create a distance measure between the items. If the distance measure represents distances along a tree, then that tree is a good estimate for the underlying, “real” tree. Normally, the distances do not represent a tree, so it is necessary to find a tree that accurately estimates the true distances. One approach to this, suggested by Barthélemy and Guénoche [11], creates a graph as follows: the nodes of the graph represent partitions of the items. These partitions are chosen because items within a partition are closer to each other than to those in the other side of the partition. Two nodes are adjacent if the partitions are consistent with coming from the same tree (which reduces to an inclusion condition). A clique in this graph represents a set of partitions that can be formed into a tree. Maximum cliques attempt to encapsulate as much of the partition data as possible. For more information, see Chapter 5 of [11].

3 Solution Approaches

3.1 Cliques

There is a vast literature covering algorithms for this problem. An extensive survey is [76], available by anonymous FTP from DIMACS.

A very common approach to the problem of finding (truly) maximum clique is to use some variant on implicit enumeration. Some examples of use of this technique are Bron and Kerbosch [20], Akkoyunlu [2], Gerhards and Lindenberg [42], Loukakis and Tsouros [65], and Carraghan and Pardalos

[22].

Approaches based on branch and bound include Babel [7] Balas and Yu [10], and Balas and Xue [9], which use coloring to obtain the required bounds, and Pardalos and Rodgers [75], which uses quadratic programming to obtain its bounds.

Polyedral approaches have been suggested by Nemhauser and Trotter [72] and Balas and Samuelsson [8].

Tarjan and Trojanowski [79] develop a recursive routine which has a worst case bound of $O(2^{n/3})$.

There has been much theoretical work on the application of approximation algorithms to random graphs (where typically a random graph is one chosen uniformly at random from the set of all labeled graphs). Bollobás and Erdös [16] and Bollobás and Thomason [17] have very strong bounds on the size of the largest clique. Pittel [78] has analyzed a number of heuristics for such graphs, and found none that will find the largest clique in this model. Jerrum [50] has extended this work to include heuristics based on simulated annealing.

Other heuristic approaches that have been considered for clique include the "GRASP" technique of Feo, Resende, and Smith [36] and the use of interior point methods by Karmarkar, Ramakrishnan, and Resende [55].

3.2 Coloring

Again, there is a vast literature, but in this case no current survey, so we will try to be a bit more complete.

Let us first consider approximations algorithms for graph coloring. The most common technique used is that of successive augmentation. In this approach a partial coloring is found on a small number of vertices and this is extended vertex by vertex until the entire graph is colored. Examples of varients of this approach include Welsh and Powell [82], Wood [85], Williams [84], Matula [67], Matula, Marble and Isaacson [68], Johnson [51, 52], Grimmet and McDiarmid [46], Brélaz [18], Leighton [64], Johri and Matula [54], Wigderson [83], Berger and Rompel [12], and Kučera [62],. The algorithm providing the currently best worst-case ratio (number of colors used divided by optimal number) is due to Halldórsson [47], guaranteeing a ratio of no more than $O(n(\log\log n)^2/(\log n)^3$, where n is the number of vertices.

More general heuristic techniques that have been tried include simulated annealing (Chams, Hertz, and De Werra [26] and Johnson et. al [53]) and tabu search (Hertz and De Werra [49]). Issues related to finding an appropriate neighborhood structure for coloring graphs are described by Morgenstern and Shapiro [71].

Algorithms for finding optimal colorings have been for the most part based on implicit enumeration. See for instance Brown [21], Christofides [29, 30], Corneil and Graham [32], Lawler [63], Korman [57], Brélaz [18], Peemöller [77], Kubale and Kusz [59], Kubale and Jackowski [58].

Other work has considered algorithms that find k-colorings quickly on average for various models of random k-colorable graphs [60, 80, 61, 14, 34, 37]. The question of the best worst-case bounds obtainable in polynomial time for k-colorable graphs (fixed k, typically $k = 3$) is addressed in [83, 13, 14]. Papers on the expected chromatic number of a "usual" random graph include Bollobás [15], and Matula and Kučera [69].

4 Suggested Research Projects

1. Implement and test algorithms from the literature. Our references contain many papers that already do this, and these should be examined closely before undertaking a study of this sort, but there is still room for such studies. One can implement theoretical algorithms not previously studied experimentally, compare algorithms that performed well in earlier studies but had not previously been directly compared, test old algorithms on new test beds, etc.

2. Use detailed profiling of an algorithm's implementation to better understand the circumstances that make it perform well or poorly, and to help extrapolate performance to very large instances sizes.

3. Design new algorithms for the problems and compare them to the previous best. These could involve substantially new algorithmic ideas or simply more efficient use of tuned data structures.

4. Adapt general purpose heuristic methods (such simulated annealing[1, 23, 31, 56, 81], tabu search [43, 49], and genetic algorithms [44]), to the clique and/or coloring problems. Some work has already been done

along these lines, as cited above, but typically there are a variety of ways in which one of these approaches can be adapted to a particular problem, and such new adaptations can be compared to previous techniques.

5. Examine neighborhood structures for local search. One aspect of these problems that makes solution difficult is the lack of a natural neighborhood structure for local search. Neighborhood structures for these problems tend either to be too narrow (leading to poor local optima) or too broad (leaving a difficult neighborhood problem). What are some alternative neighborhoods and how can they be effectively searched?

6. Explore the role of randomization. Most of the algorithms created are completely deterministic. Can randomization help in solving these types of problems?

7. Experiment on alternative machine architectures, including parallel and vector processors. How can multiple processors be exploited in solving these problems? This could be either via distributed computing (generally with high communications costs, heterogeneous machines, and/or limited connectivity) or with such special purpose machines as Connection Machine computers.

8. Examine real world problems. The clique problem is commonly claimed to have widespread applicability. Most computational work to date, however, has been done with random graphs. How do real world problems differ from such random graphs? How would the conclusions drawn from past research be different if real problems were solved? Identify a real-world application from which you can obtain instance data, and then answer the above questions.

9. Create and examine alternative instance generators. Consider the "geometric graphs" of [53], where the nodes of a graph as points in the Euclidean plane where an edge occurs only if the corresponding nodes are sufficiently close (or sufficiently far away). Consider also the generators proposed in [48]. Are these interesting problem generators? Do algorithms perform well or poorly for them? What other generators have interesting properties? (One "product" we hope this Challenge

to produce is a broad library of generators and real-world instances for future researchers to use.)

10. Examine "pathological" examples. For each optimization algorithm developed, there is a class of examples for which the algorithm does well and a class for which it does poorly. Can you characterize why an algorithm does poorly? Alternatively, can you determine a large, interesting, class for which certain algorithms work well?

11. Examine planar graphs. The four color theorem, as proved by [3, 4] is actually a constructive theorem that gives an $O(n^2)$ algorithm for 4–coloring a graph. This algorithm seems completely impractical due to the large constants involved. Morgenstern and Shapiro [70] provide some heuristics for such graphs and might be seen as a starting point for this class of interesting graphs.

12. Examine more fundamental issues in computational testing using these problems as a testbed. How should algorithms and heuristics be compared across environments? Does good computational work require outstanding "hacking" ability, or is there some more fundamental measure independent of implementation?.

References

[1] E.H.L. Aarts and J.H.M Korst. *Simulated Annealing and Boltzmann Machines*. John Wiley & Sons, Chichester, U.K., 1989.

[2] E.A. Akkoyunlu. The enumeration of maximal cliques of large graphs. *SIAM Journal on Computing*, 2(1):1–6, 1973.

[3] K. Appel and W. Haken. Every planar map is 4-colorable – 1: Discharging. *Illinois Journal of Mathematics*, 21:421–490, 1977.

[4] K. Appel and W. Haken. Every planar map is 4-colorable – 2: Reducibility. *Illinois Journal of Mathematics*, 21:491–567, 1977.

[5] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. In *Proceedings*

33rd IEEE Symposium on the Foundations of Computer Science, pages 14–23, Los Angeles, CA, 1992. IEEE Computer Society.

[6] S. Arora and S. Safra. Probabilistic checking of proofs; a new characterization of np. In *Proceedings 33rd IEEE Symposium on Foundations of Computer Science*, pages 2–13, Los Angeles, CA, 1992. IEEE Computer Society.

[7] Luitpold Babel. Finding maximum cliques in arbitrary and in special graphs. *Computing*, 46:321–341, 1991.

[8] Egon Balas and H. Samuelsson. A node covering algorithm. *Naval Research Logistics Quarterly*, 24(2):213–233, 1977.

[9] Egon Balas and Jue Xue. Minimum weighted coloring of triangulated graphs, with application to maximum weight vertex packing and clique finding in arbitrary graphs. *SIAM Journal on Computing*, 20(2):209–221, 1991.

[10] Egon Balas and Chang Sung Yu. Finding a maximum clique in an arbitrary graph. *SIAM Journal on Computing*, 15(4):1054–1068, 1986.

[11] Jean-Pierre Barthélemy and Alain Guénoche. *Trees and Proximity Representations.* John Wiley and Sons, New York, NY, 1991.

[12] B. Berger and J. Rompel. A better permance guarantee for approximate graph coloring. *Algorithmica*, 5:459–466, 1990.

[13] A. Blum. An $\tilde{O}(n^{0.4})$-approximation algorithm for 3-coloring (and improved approximation algorithms for k-coloring). In *Proceedings 21st ACM Symposium on Theory of Computing*, pages 535–542, New York, 1989. ACM.

[14] A. Blum. Some tools for approximate 3-coloring. In *Proceedings 31st IEEE Symposium on Foundations of Computer Science*, pages 554–562, Los Angeles, CA, 1990. IEEE Computer Society.

[15] B. Bollobás. The chromatic number of random graphs. *Combinatorica*, 8:49–56, ???

[16] B. Bollobás and P. Erdös. Cliques in random graphs. *Mathematical Proceedings of the Cambride Philosophical Society*, 80:419–427, 1976.

[17] B. Bollobas and A. Thomason. Random graphs of small order. *Annals of Discrete Math.*, 28:47–97, 1985.

[18] D. Brélaz. New methods to color vertices of a graph. *Communications of the ACM*, 22:251–256, 1979.

[19] P. Briggs, K. Cooper, K. Kennedy, and L. Torczon. Coloring heuristics for register allocation. In *ASCM Conference on Program Language Design and Implementation*, pages 275–284, 1989.

[20] Coen Bron and Joep Kerbosch. Finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577, 1973.

[21] R. J. Brown. Chromatic scheduling and the chromatic number problem. *Management Science*, 19:451–463, 1972.

[22] R. Carraghan and P. M. Pardalos. An exact algorithm for the maximum clique problem. *Operations Research Letters*, 9:375–382, 1990.

[23] V. Cerny. A thermodynamical approach to the traveling saleman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45:41–51, 1985.

[24] G.J. Chaitin. Register allocation and spilling via graph coloring. In *Proceedings of the ACM SIGPLAN 82 Symposium on Compiler Construction*, pages 98–105, New York, NY, 1982. ACM.

[25] G.J. Chaitin, M. Auslander, A.K. Chandra, J. Cocke, M.E. Hopkins, and P. Markstein. Register allocation via coloring. *Coputer Languages*, 6:47–57, 1981.

[26] M. Chams, A. Hertz, and D. De Werra. Some experiments with simulated annealing for coloring graphs. *European Journal of Operations Research*, 32:260–266, 1987.

[27] Fred C. Chow and John L. Hennessy. Register allocation by priority–based coloring. In *Proceedings of the ACM SIGPLAN 84 Symposium on Compiler Construction*, pages 222–232, New York, NY, 1984. ACM.

[28] Fred C. Chow and John L. Hennessy. The priority–based coloring approach to register allocation. *ACM Transactions on Programming Languages and Systems*, 12(4):501–536, 1990.

[29] N. Christofides. An algorithm for the chromatic number of a graph. *Computer Journal*, 14:38–39, 1971.

[30] N. Christofides. *Graph Theory: An Algorithmic Approach*. Academic Press, London., 1975.

[31] N.E. Collins, R.W. Eglese, and B.L. Golden. Simulated annealing: An annotated bibliography. *American Journal of Mathematical and Management Sciences*, 8:205–307, 1988.

[32] D. G. Corneil and B. Graham. An algorithm for determining the chromatic number of a graph. *SIAM Journal on Computing*, 2:311–218, 1973.

[33] D. De Werra. An introduction to timetabling. *European Journal of Operations Research*, 19:151–162, 1985.

[34] M. Dyer and A. Frieze. The solution of some random np–hard problems in polynomial expected time. *Journal of Algorithms*, 10:451–489, 1989.

[35] U. Feige, S. Goldwasser, L. Lovasz, S. Safra, , and M. Szegedy. Approximating clique is almost NP-complete. In *Proceedings 32nd IEEE Symposium on Foundations of Computer Science*, pages 2–12, Los Angeles, CA, 1991. IEEE Computer Society.

[36] T. A. Feo, M. G. C. Resende, and S. H. Smith. Greedy randomized adaptive search procedure for maximum independent set. *Operations Research*, page (to appear), 1993.

[37] M. Fürer and C. R. Subramanian. Coloring random graphs. In *Proceedings 3rd Scandinavian Workshop on Algorithmic Theory*, Berlin, 1992. Springer Lecture Notes in Computer Science.

[38] Andreas Gamst. Some lower bounds for a class of frequency assignment problems. *IEEE Transactions of Vehicular Echnology*, 35(1):8–14, 1986.

[39] M. R. Garey, D. S. Johnson, and H. C. So. An application of graph coloring to printed circuit testing. *IEEE Trans. on Circuits and Systems*, CAS-23:591–599, 1976.

[40] Michael R. Garey and David S. Johnson. The complexity of near–optimal graph coloring. *Journal of the Association for Computing Machinery*, 23:43–49, 1976.

[41] Michael R. Garey and David S. Johnson. *Computers and Instractability: A Guide to the Theory of NP–Completeness.* W.H Freeman, San Francisco, CA, 1979.

[42] L. Gerhards and W. Lindenberg. Clique detection for nondirected graphs: Two new algorithms. *Computing*, 21:295–322, 1979.

[43] Fred Glover. Tabu search, part 1. *ORSA Journal on Computing*, 1:190–206, 1989.

[44] D. E. Goldberg. *Genetic Algorithms in Search, Optimization & Machine Learning.* Addison-Wesley, Reading, MA, 1989.

[45] M. Golumbic. *Algorithmic Graph Theory and Perfect Graphs.* Acadmeic Press, New York, NY, 1980.

[46] G.R. Grimmet and C.J.H. McDiarmid. On colouring random graphs. *Mathematical Proceedings of the Cambridge Philosophical Society*, 77:313–324, 1975.

[47] M. M. Halldórsson. A still better performance guarantee for approximate graph coloring. Technical Report 91–35, DIMACS, New Brunswick, NJl, 1990.

[48] J. Hasselberg, P. M. Pardalos, and G. Vairaktarakis. Test case generators and computational results for the maximum clique problem. Working paper, University of Florida, 1992.

[49] A. Hertz and D. De Werra. Using tabu search techniques for graph coloring. *Computing*, 39:345–351, 1987.

[50] Mark Jerrum. Large cliques elude the metropolis process. *Random Structures and Algorithms*, 3(4):347–360, 1992.

[51] David S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9:256–278, 1974.

[52] David S. Johnson. Worst–case behavior og graph coloring algorithms. In *Proceedings 5th Southeastern Conference on Combinatorics, Graph Theory, and Computing*, pages 513–527, Winnipeg, Canada, 1974. Utilitas Mathematica Publishing.

[53] David S. Johnson, Cecilia R. Aragon, Lyle A. McGeoch, and Catherine Schevon. Optimization by simulated annealing: An experimental evaluation; part ii, graph coloring and number partitioning. *Operations Research*, 39(3):378–406, 1991.

[54] A. Johri and David W. Matula. Probabilistic bounds and heuristic algorithms for coloring large random graphs. Technical report, Southern Methodist University, Dallas, Texas, 1982.

[55] N. K. Karmarkar, K. G. Ramakrishnan, and M. G. C. Resende. An interior point approach to the maximum independent set problem in dense random graphs. In *Proceedings of the XV Latin American Conference on Informatics I*, pages 241–260, 1989.

[56] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220(13 May 1983):671–680, 1983.

[57] S. M. Korman. The graph-colouring problem. In N. Christofides, A. Mingozzi, P. Toth, and C. Sandi, editors, *Combinatorial Optimization*, pages 211–235, New York, 1970. Wiley.

[58] M. Kubale and B. Jackowski. A generalized implicit enumeration algorithm for graph coloring. *Communications of the ACM*, 28:412–418, 1985.

[59] M. Kubale and E. Kusz. Computational experience with implicit enumeration algorithms for graph coloring. In M. Nagl and J. Perl, editors, *Proceedings of the WG'83 International Workshop on Graphtheoretic Concepts in Computer Science*, pages 167–176, Linz, 1983. Trauner Verlag.

[60] Ludek Kučera. Expected behavior of graph coloring algorithms. In M. Karpinski, editor, *Foundations of Comupation Theory 77*, volume 56 of *Lecture Notes in Computer Science*, pages 447–451, Berlin, 1977. Springer.

[61] Ludek Kučera. Graphs with small chromatic numbers are easy to color. *Information Processing Letters*, 30:233–236, 1989.

[62] Ludek Kučera. The greedy coloring is a bad probabilistic algorithm. *Journal of Algorithms*, 12:674–684, 1991.

[63] E. L. Lawler. A note on the complexity of the chromatic number problem. *Information Processing Letters*, 5:66–67, 1976.

[64] F.T. Leighton. A graph coloring algorithm for large scheduling problems. *Journal of Reasearch of the National Bureau of Standards*, 84:489–506, 1979.

[65] E. Loukakis and C. Tsouros. Determining the number of internal stability of a graph. *International Journal of Computer Mathematics*, 11:207–220, 1982.

[66] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. In *Proceedings 25th ACM Symposium on Theory of Computing*, page (submitted), New York, NY, 1993. ACM.

[67] D. G. Matula. Bounded color functions on graphs. *Networks*, 2:29–44, 1972.

[68] D. W. Matula, G. Marble, and J. D. Isaacson. Graph theory and computing. In R. C. Read, editor, *Graph coloring algorithms*, pages 109–122, New York, NY, 1972. Academic Press.

[69] David Matula and Ludek Kučera. An expose–and–merge algorithm and the chromatic number of a random graph. In M. Karonski, J. Jaworski, and A. Rucinski, editors, *Random Graphs'87*, pages 175–187. John Wiley and Sons, 1990.

[70] C. Morgenstern and H. Shapiro. Heuristics for rapidly four–coloring large planar graphs. *Algorithmica*, 6:869–891, 1991.

[71] Craig Morgenstern and Harry Shapiro. Coloration neighborhood structures for general graph coloring. In *First Annual ACM–SIAM Symposium on Discrete Algorithms*, 1990.

[72] George L. Nemhauser and Les E. Trotter. Vertex packings: Structural properties and algorithms. *Mathematical Programming*, 8:232–248, 1975.

[73] Hideo Ogawa. Labeled point pattern matching by delaunay triangulation and maximal cliques. *Pattern Recognition*, 19(1):35–40, 1986.

[74] R.J. Opsut and Fred S. Roberts. On the fleet maintenance, movile radio frequency, task assignment and traffic phasing problems. In G. Chartrand, Y. Alavi, D.L. Goldsmith, L. Lesniak-Foster, and D.R. Lick, editors, *The Theory and Applications of Graphs*, pages 479–492, New York, NY, 1981. John Wiley & Sons.

[75] P. M. Pardalos and G. P. Rodgers. A branch and bound algorithm for the maximum clique problem. *Computers and Operations Research*, 19:363–375, 1992.

[76] P. M. Pardalos and J. Xue. The maximum clique problem. *Journal of Global Optimization*, page (to appear), 1993.

[77] J. Peemöller. A correction to Brélaz's modification of Brown's coloring algorithm. *Communications of the ACM*, 26:595–597, 1983.

[78] B. Pittel. On the probable behaviour of some algorithms for finding the stability number of a graph. *Mathematical Proceedings of the Cambride Philosophical Society*, 92:511–526, 1982.

[79] Robert E. Tarjan and A.E. Trojanowski. Finding a maximum independent set. *SIAM Journal on Computing*, 6:266–283, 1977.

[80] J. S. Turner. Almost all k-colorable graphs are easy to color. *Journal of Algorithms*, 9:63–82, 1988.

[81] P.J.M Van Laarhoven and E.H.L. Aarts. *Simulated Annealing: Theory and Practice.* Kluwer Academic Publishers, Dordrrecht, The Netherlands, 1987.

[82] D.J.A. Welsh and M.B. Powell. An upper bound on the chromatic number of a graph and its application to timetabling problems. *Computer Journal*, 10:85–86, 1967.

[83] A. Wigderson. Improving the performance guarantee of approximate graph coloring. *Journal of the Association for Computing Machinery*, 30:729–735, 1983.

[84] M. R. Williams. The coloring of very large graphs. In *Combinatorial Structures and their Applications*, pages 477–478, New York, 1970. Gordon and Breach.

[85] D. C. Wood. A technique for coloring a graph applicable to large scale time-tabling problems. *The Computer Journal*, 3:317–319, 1969.

[86] G. Yu, O. Goldschmidt, and H. Chen. Clique, independent set, and vertex cover in geometric graphs. ORSA/TIMS Presentation, San Francisco, CA, 1992.

[87] G. Yu, P. Kouvelis, and Luo. A weighted vertex packing problem for specially structured geometric graphs arising in the design of electronic testing fixtures. ORSA/TIMS Presentation, San Francisco, CA, 1992.