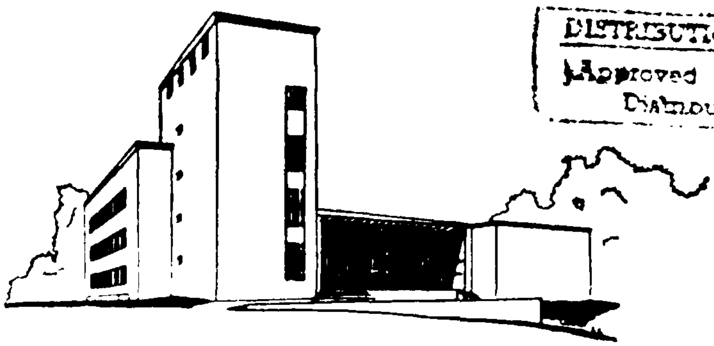
1

Carnegie Mellon University

PITTSBURGH, PENNSYLVANIA 15213

GRADUATE SCHOOL OF INDUSTRIAL ADMINISTRATION

WILLIAM LARIMER MELLON, FOUNDER



1

W.P. #1991-18

Management Sciences Research Report No. MSRR 565

AN EXACT ALGORITHM FOR FINDING UNDIRECTED HAMILTONIAN CYCLES BASED ON A TWO-MATCHING PROBLEM RELAXATION

MARCH 22, 1991

W. P. #1991-18

Management Science Research Report No. MSRR 565

AN EXACT ALGORITHM FOR FINDING UNDIRECTED HAMILTONIAN CYCLES BASED ON A TWO-MATCHING PROBLEM RELAXATION

March 22, 1991

D. L. Miller

Central Research and Development Department
E. I. du Pont de Nemours and Company Inc.
Wilmington, Delaware 19898

J. F. Pekny[†]

Department of Chemical Engineering
Purdue University
West Lafayette, IN 47907

G. L. Thompson[‡]

Graduate School of Industrial Administration
Carnegie Mellon University
Pittsburgh, PA 15213

[†] This work has been supported by the Engineering Design Research Center at Carnegie Mellon University, an NSF Engineering Research Center.

[‡] Work supported in part by the Office of Naval Research, under Contract No. N00014-85-K-0-0198 NR 047-048, and in part by the Department of Energy, under Contract No. DE-FG02-85ER13396-IH121A. Reproduction in whole or in part is permitted for any purpose of the U. S. Government.

Management Science Research Group
Graduate School of Industrial Administration
Carnegie Mellon University
Pittsburgh, PA 15213

An Exact Algorithm for Finding Undirected Hamiltonian Cycles Based on a Two-Matching Problem Relaxation

by

D. L. Miller, J. F. Pekny and G. L. Thompson

March 22, 1991

ABSTRACT

We describe an algorithm for finding two-matchings in undirected graphs. This algorithm is used as a basis for a simple exact algorithm for determining the hamiltonicity of undirected graphs. Results are presented for random graphs with up to 30,000 vertices and for knight's tour problems having up to 10,000 vertices.

I. Introduction

The task of identifying hamiltonian cycles has long been of both theoretical and practical interest, starting in 1856 with Kirkman [1]. In the general case, this problem is known to be *NP*-Complete. Much research has been done on criteria and algorithms for identification of hamiltonian cycles for special classes of graphs [2-4]. The case of arbitrary graphs has proven difficult since theoretical criteria [5-8] are either too vague or computationally expensive to be of use. A review of theoretical results can be found in [9].

Many heuristic and probabilistic techniques have been proposed. For undirected and directed random graphs, recent work includes Angluin and Valiant [10], Perepelica [11], Thompson and Singhal [12, 13], and Thomason [14]. Results similar to Angluin and Valiant are reported for digraphs by Frieze [15]. Many of these authors include probabilistic results concerning the asymptotic performance of their heuristics on random graphs, however, none provide a guarantee for finite sized graphs or structured graphs. The algorithm presented here is simple, exact, and remarkably well behaved for the same class of random graphs.

From a computational standpoint, heuristic methods have serious drawbacks. In particular, heuristic methods use arbitrary criteria for termination. Typically, heuristics employ stopping rules that are coupled to the amount of computational effort. In worst case, a heuristic may fail to find a hamiltonian cycle when in fact the graph contains one. In contrast, the exact algorithm described in this paper has well defined termination criteria and always correctly reports the hamiltonicity of an undirected graph. Of course, since the exact algorithm is enumerative in nature, certain graphs may lead to unacceptably large execution times, but heuristics also suffer from this problem. Our computational experience shows that for random graphs the algorithm enumerates only a small fraction of the possible search space, resulting in acceptable execution times.

For ten 30,000 vertex graphs, we determined hamiltonicity in about 200 seconds on average using a Sun 4/330 workstation.

Some exact techniques have been proposed for the hamiltonian cycle problem. Algebraic methods which have been proposed such as [16] become computationally intractable for graphs having more than a few dozen vertices. Enumerative methods, such as Roberts and Flores [17] have been successful for small graphs. Multipath (or multiway) algorithms proposed by Selby [18] and Christofides [16] are based on enumerative techniques. All the enumerative methods, however, require backtracking and can lead to combinatorially explosive search.

The remainder of this paper is divided into four sections. In Section 2 we briefly describe an unweighted two-matching algorithm with worst case complexity $O(nm)$, where n is the number of vertices and m is the number of edges. Section 3 describes how the two-matching problem is used as a relaxation in an enumerative algorithm for determining the hamiltonicity of an undirected graph. Section 4 and Section 5 present computational results for the two-matching and hamiltonian cycle algorithms, respectively.

2. An Unweighted Two-Matching Algorithm

The unweighted two-matching problem may be stated as follows: Given a graph $G = (V,E)$, find a subgraph of G such that each vertex $v \in V$ has degree two, i.e. if possible determine $M \subseteq E$ such that each vertex appears in two edges of M. Subset M is known as a two-matching. A two-matching consists of a collection of one or more disjoint cycles, each containing at least three vertices. If the two-matching has only a single cycle, then it is a hamiltonian cycle. Because of this relationship, the two-matching problem is a relaxation of the hamiltonian cycle problem.

We have developed an algorithm for solving the unweighted two-matching problem based on the work of Anstee [19], rather than the traditional Edmond's approach [20]. The algorithm involves solving a two-matching problem on a bipartite graph $G_b = (V_b, E_b)$ that is closely related to G. Vertex set V_b contains two vertices b_{i1} and b_{i2} for every vertex $i \in V$. For each member of the original edge set $(i,j) \in E$, E_b contains two members: (b_{i1}, b_{j2}) and (b_{j1}, b_{i2}). The bipartite two-matching problem may be easily solved as a network flow problem or it may be solved directly using the theory of alternating paths. We solve bipartite two-matching using an alternating path algorithm with $O(nm)$ worst case complexity.

Two-matching solutions on graph G_b may be interpreted as follows. Let M be a set of edges and directed arcs which is initially empty: if both of the edges (b_{i1}, b_{j2}) and (b_{j1}, b_{i2}) appear in the bipartite two-matching solution on G_b, place edge $(i,j) \in E$ in set M; if edge (b_{i1}, b_{j2}) appears but edge (b_{j1}, b_{i2}) does not, place directed arc (i,j) in the set M. If graph G_b does not posses a two-matching, then neither does graph G.

If set M contains no directed arcs, the edges of M represent a two-matching solution. If M contains directed arcs, then they form a collection of directed cycles on graph G. Using transforms proposed by Anstee [19], directed cycles of even length and non-disjoint pairs of directed cycles of odd length may be replaced by half the number of edges. Residual odd disjoint cycles may be eliminated by finding alternating walks connecting pairs of these cycles. Here we use the term alternating in the usual sense, the edges in the walk are alternatively in and out of M. Each alternating walk is used to replace the arcs of a pair of directed cycles by half the number of edges. We denote this act of replacement a *transfer* because of the analogy with ordinary matching theory. When (and if) all directed cycles are replaced, set M represents a two-matching. If an alternating walk cannot be found between one or more pairs of odd directed cycles of set M, then graph G does not possess a two-matching.

Alternating walks are located by growing alternating trees rooted at directed cycles. Because graph G is not necessarily bipartite, the alternating trees may form pseudovertices (blossoms) whenever addition of an edge produces a cycle. We have developed data structures for efficient tree growth and a recursive procedure for carrying out the transfer along an identified alternating walk. The worst case complexity for eliminating directed arcs is $O(nm)$, yielding a two-matching algorithm that is $O(nm)$ in the worst case. Naturally, if graph G is bipartite, M cannot contain odd directed cycles.

3. Unweighted Hamiltonian Cycle Algorithm

Given the unweighted two-matching algorithm described in the previous section, the hamiltonian cycle algorithm is a standard application of partial enumeration techniques. At each vertex of the search tree the two-matching algorithm is applied to a graph. The graph at the root vertex of the search tree consists of the original graph. If the graph at any vertex of the search tree does not contain a two-matching, that vertex is fathomed. If the graph contains a two-matching, one child is created for each non-fixed edge incident to a vertex of the graph having minimum degree. Each child uniquely differs from the parent by having fixed one of these edges. Fixed edges must appear in subsequent two-matchings. We improved the effectiveness of the enumeration algorithm by using a heuristic that attempts to patch the cycles of the two-matching into a single cycle. This heuristic is a modification of Karp's asymmetric patching algorithm [21].

The enumeration algorithm can be improved by using other relaxations to complement two-matching. For example the graphs at each vertex of the search tree could be checked to see that they contain spanning trees, or that they are biconnected. Either of these checks would guard against searching for hamiltonian cycles in graphs that consist of dense but disconnected components. The biconnectivity condition is superior because it also guards against the case of two dense components connected by a single edge. No such refinements are required to handle random problems.

4. Unweighted Two-Matching Algorithm Computational Results

Table 1 presents results for the unweighted two-matching algorithm for random graphs at two densities, collected on a Sun 4/330 workstation. The tables indicates the total number of trials for each size and density. The "success/fail" column indicates the number of trials that did or did not possess a two-matching. Phase one symmetrization refers to the process of eliminating nondisjoint odd directed cycles and even directed cycles of arcs in set M. Phase two symmetrization refers to the process of finding alternating walks connecting two disjoint odd directed cycles. The table indicates the number of graphs possessing odd cycles after phase one symmetrization ("cases w/odd cycles column"). The "avg. odd cycles" column reports the average number of disjoint odd cycles for cases in which symmetrization was not complete after phase one. Finally, the table lists the execution times of various phases of the algorithm.

Table 1 - Unweighted Two-Matching Algorithm Performance on Random Graphs

n	cases	success/fail two-matchings	cases w/ odd cycles	avg. odd cycles	bipartite matching time(sec)	phase one sym. time(sec)	phase two sym. (sec)	total time (sec)
				Density = 0.02				
500	25	20/5	14	2.429	0.086	0.012	0.002	0.122
1000	25	25/0	23	3.304	0.209	0.036	0.018	0.304
2000	25	25/0	24	3.333	0.618	0.079	0.023	0.755
				Density = 0.25				
500	25	25/0	11	2.000	0.074	0.020	0.003	0.113
1000	25	25/0	7	2.000	0.284	0.041	0.002	0.357
2000	25	25/0	2	2.000	1.042	0.085	0.001	1.182

5. Hamiltonian Cycle Algorithm Computational Results for Random Graphs

In this section we consider the performance of our exact algorithm for determining hamiltonicity on random graphs. Our test problems were generated as follows: Each edge $e \in E_c$ in a complete undirected graph $G_c = (V, E_c)$ was randomly assigned a number p_e between zero and one. For any value of ε the graph $G(\varepsilon) = (V, E)$ may be derived from G_c by letting $E = \{e \in E_c | p_e \leq \varepsilon\}$ Obviously $G(0)$ is not hamiltonian and $G(1)$ is. Furthermore, there is some ε_t at which the graph becomes hamiltonian. In graphs with $\varepsilon \gg \varepsilon_t$ it is easy to find hamiltonian cycles and graphs with $\varepsilon \ll \varepsilon_t$ are easily shown to be non-hamiltonian. We used our algorithm to determine the value of ε_t to an accuracy of one part in ten million. This requires not only that the algorithm find hamiltonian cycles in graphs with ε just above ε_t, but also that graphs which are just short of the threshold be proved to be non-hamiltonian. This cannot be done by using heuristics.

The following table presents computational results from the collected on a Sun 5/330 workstation. The "cases" column indicates the number of trials at each size. Each line in the table summarizes performance according to graph hamiltonicity during the binary search. The execution times reported in the table show the average times necessary to determine the hamiltonicity of a single graph, i.e. the average time to determine the hamiltonicity of $G(\varepsilon)$ for fixed ε.

Table 2 - Undirected Hamiltonian Circuit Algorithm

n	cases	threshold	execution time (sec)	
			hamiltonian	non-hamiltonian
1000	10	0.009235	0.447	0.277
5000	10	0.002385	6.777	3.643
10000	10	0.001320	20.284	13.898
20000	10	0.000646	143.509	51.414
30000	10	0.000471	226.031	90.945

We also applied the algorithm to the problem of finding a knight's tour on an $n \times n$ chessboard. This corresponds to finding a hamiltonian cycle in a graph of n^2 nodes in which the degree of each node is between two and eight. Knight's tour graphs are also bipartite so that phase two symmetrization is not needed. Table 3 shows the time required for the algorithm to fird knight's tours on chessboards ranging from 8x8 to 100x100. Performance is shown with and without the use of the patching algorithm. Patching provides an excellent heuristic for the knight's tour problem, solving ten of the twelve test problems using only the initial two-matching. Solution of the other two problems occurred at the second and third search tree nodes. Without patching, the number of nodes in the search tree and the execution times grow dramatically, however the algorithm is still able to solve problems with thousands of nodes in reasonable time.

Table 3 - Knight's Tour Problem

	with patch		without patch	
board size	nodes solved	execution time	nodes solved	execution time
8x8	1	0.08	5	.05
10x10	1	0.05	7	.08
12x12	1	0.10	11	.16
14x14	1	0.12	8	.18
16x16	1	0.16	27	.52
18x18	1	0.23	13	.44
20x20	1	0.34	31	.88
30x30	1	1.28	57	3.37
40x40	2	3.11	256	26.1
50x50	1	6.26	876	160.2
70x70	3	17.0	-	-
100x100	1	46.8	-	-

6. Conclusions

We have described an exact algorithm for unweighted two-matching and used that algorithm as a basis for an exact hamiltonian cycle algorithm for undirected graphs. Computational results on random graphs indicate that two-matching is a good relaxation for hamiltonicity; there is a strong correlation between the probability that a random graph possesses a two-matching the probability that the graph is hamiltonian. The reasonable computational times required to solve large random problems indicate that such random problems are not difficult, and this is in agreement with predictions of asymptotic behavior for probabilistic algorithms [10,14]. Clearly this algorithm can fail on problems with certain structure, for example graphs consisting of dense unconnected components. The addition of a simple connectivity check, or better a check for biconnectivity would improve algorithmic performance on non-random problems. The performance on knight's tour problems with up to 10,000 nodes is surprisingly good because this problem has been considered difficult for some heuristics [22]. A case may be made that this algorithm is superior to currently known heuristics, although this is difficult to establish given the shortage of computational results for heuristic methods. The results also suggest that the combination of two-matching followed by patching may well be competitive with tour construction heuristics.

References

1. T. P. Kirkman, "On the Representation of Polyhedra," *Phil. Trans. Roy. Soc. London Ser.* A, vol. 146, pp. 413-418, 1856.

2. M. Meyniel, "Une Condition suffisante d'existence d'un circuit hamiltonien dans un grapphe oriente.," *J. Combinat. Theory B*, vol. 14, pp. 137-147, 1973.

3. W. T. Tutte, "On Hamilton circuits," *J. London Math. Soc.*, vol. 21, pp. 98-101, 1946.

4. F. Ewald, "Hamiltonian Circuits in Simplicial Complexes," *Geo. dedicata*, vol. 2, pp. 115-125, 1973.

5. Nash-Williams, "On Hamiltonian Circuits in Finite Graphs," *Proc. American Mathematical Society*, vol. 17, p. 466, 1966.

6. O. Ore, *Theory of Graphs*, American Mathematical Society, New York, 1962.

7. G. A. Dirac, "Some Theorems on Abstract Graphs," *Proc. London Mathematical Soc.*, vol. 2, pp. 68-81, 1952.

8. M. Gondran and M. Minoux, *Graphs and Algorithms*, Wiley, New York, 1986.

9. V. Chvatal, "Hamiltonian Cycles," in *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, ed. E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, D. B. Shmoys, pp. 403-429, John Wiley & Sons, New York, 1985.

10. D. Angluin and L. G. Valiant, "Fast Probabilistic Algorithms for Hamiltonian Circuits and Matchings," *Journal of Computer and System Sciences*, vol. 18, pp. 155-193, 1979.

11. V. A. Perepelica, "On Two Problems from the Theory of Graphs," *Soviet Math. Dokl.*, vol. 11, pp. 1376-1379, 1970.

12. G. L. Thompson and S. Singhal, "A Successful Algorithm for Solving Directed Hamiltonian Path Problems," *Operations Research Letters*, vol. 3, No. 1, pp. 35-42, 1984.

13. G. L. Thompson and S. Singhal, "A Successful Algorithm for the Undirected Hamiltonian Path Problems," *Discrete Applied Mathematics*, vol. 10, pp. 179-195, 1985.

14. A. Thomason, "A Simple Linear Expected Time Algorithm for Finding a Hamilton Path," *Discrete Mathematics*, vol. 75, pp. 373-379, 1989.

15. A. M. Frieze, "An Algorithm for Finding Hamilton Cycles in Random Directed Graphs," *Journal of Algorithms*, vol. 9, pp. 181-204, 1988.

16. N. Christofides, *Graph Theory, an Algorithmic Approach*, Academic Press, New York, 1975.

17. S. M. Roberts and B. Flores, "Systematic Generation of Hamiltonian Circuits," *Communications of the ACM*, vol. 9, p. 690, 1966.

18. G. R. Selby, *The Use of Topological Methods in Computer-Aided Circuit Layout*, London University, 1970, Ph.D. Thesis.

19. R. P. Anstee, "An Algorithmic Proof of tutte's f-Factor Theorem," *Journal of Algorithms*, vol. 6, pp. 112-131, 1985.

20. J. Edmonds, "Paths, Trees and Flowers," *Canad. J. Math.*, vol. 17, pp. 449-467, 1965.

21. R. M. Karp, "A Patching Algorithm for the Nonsymmetric Travelling-Salesman Problem," *Siam J. Computers*, vol. 8, no. 4, pp. 561-573, 1979.

22. C. H. Papadimitriou and K. Steiglitz, "Some Examples of Difficult Traveling Salesman Problems," *Operations Research*, vol. 26, no. 3, pp. 434-443, 1978.