A New Algorithm for the Maximum-Weight Clique Problem

Patric R. J. Östergård [1]

Department of Computer Science and Engineering, Helsinki University of Technology, P.O. Box 5400, 02015 HUT, Finland

Abstract

Given a graph, in the maximum clique problem one wants to find the largest number of vertices any two of which are adjacent. In the maximum-weight clique problem, the vertices have weights, and one wants to find a clique with maximum weight. A recently developed algorithm for the maximum clique problem is here transformed into an algorithm for the weighted case. Computational experiments with random graphs show that this new algorithm in many cases is faster than earlier algorithms.

1 Introduction

We denote an undirected graph by $G = (V, E)$, where V is a set of vertices and E is a set of edges, such that each edge is a set of two vertices in V (which are said to be adjacent). A *clique* in G is a subset $S \subseteq V$ of vertices, such that $\{x, y\} \in E$ for all distinct $x, y \in S$. Consequently, any two vertices of a clique are adjacent. A clique is said to be *maximal* if its vertices is not a subset of the vertices of a larger clique, and *maximum* if there are no larger cliques in the graph.

Over the years much effort has been put on developing algorithms for finding cliques in graphs since this is a central problem in graph theory with many practical applications. For a given graph, there are several issues one may be interested in, such as finding all cliques, finding all maximal cliques, or finding one maximum clique (or all). In a recent paper [9], the author presents a new, fast algorithm for solving the maximum clique problem.

[1] Supported by the Academy of Finland

The problem of finding a maximum clique is a computationally difficult problem. In fact, it is NP-hard [6]. It is computationally equivalent to some other central problems for graphs, such as those of finding a maximum independent set and finding a minimum vertex cover. One may generalize this problem by assigning positive, integer weights to the vertices and asking for the maximum-weight clique. Then the clique is not necessarily a maximum clique of the underlying unweighted graph, but it is certainly maximal. Also this problem has several computationally equivalent variants, one being the weighted node packing problem (WNP). Various issues of the maximum clique problem and the maximum-weight clique problem are surveyed in [11].

The algorithm developed in [9] is here transformed into an algorithm for the weighted case. The algorithm is considered in Section 2, and a comparison with old algorithms is discussed in Section 3.

2 The New Algorithm

In the algorithm, we first impose an order on the vertices: $V = \{v_1, v_2, \ldots, v_n\}$ where $|V| = n$. This order only affects the efficiency of the algorithm, not the correctness. Many different orderings of the vertices were tested. One good ordering is as follows. Let v_1 be the vertex with largest weight in G. If there are several such vertices, the one with the largest total sum of the weights of its adjacent vertices is chosen. To determine v_2, the same procedure is repeated for $G \setminus \{v_1\}$. In general, having fixed $v_1, v_2, \ldots, v_k$ with $1 \leq k \leq n-1$, we obtain v_{k+1} by applying this procedure to $G \setminus \{v_1, v_2, \ldots, v_k\}$.

We now calculate values of $C(k)$, $1 \leq k \leq n$, which tells the largest weight of a clique in the subgraph induced by the vertices $\{v_k, v_{k+1}, \ldots, v_n\}$. Let $w(i)$ denote the weight of vertex v_i. Then $C(n) = w(n)$, and the values of $C(n-1), C(n-2), \ldots$, in this order, are determined in a backtrack search.

In the backtrack search, we use the information that for $1 \leq k \leq n-1$, $C(k) > C(k+1)$ exactly when a corresponding clique exists and contains v_k. We search for the largest such clique in the following way (if no such clique exists, $C(k) = C(k+1)$).

We maintain a set of vertices that are adjacent to all vertices fixed so far in the search. This set is called the *working set*. At each level in the search tree, one vertex of the working set is fixed, and the new working set consists of the intersection of the vertices adjacent to the fixed vertex and the vertices in the working set with higher index than the fixed vertex (higher index to avoid repetitions in the search).

Clearly, one may now backtrack if the sum of the weights of the vertices in the working set is so small that even if all these vertices could be added to the current clique, the clique would not have desired weight. Moreover—as we search for a clique with weight greater than or equal to W—if the weight of the fixed vertices is W' and we consider v_k to be the next fixed vertex, then we can prune the search if $W' + C(k) < W$.

3 Experimental Results

The first algorithm for the general maximum-weight clique problem was apparently the one published by Nemhauser and Trotter [8] in 1975. Later results include those by Babel [1], Balas and Samuelsson [2], Balas and Xue [3,4], Carraghan and Pardalos [5], Loukakis and Tsouros [7], and Pardalos and Desai [10].

To fully evaluate the proposed algorithm, it should be compared with old algorithms. Unfortunately, not all published papers present experimental evaluations. The paper [3] by Balas and Xue, however, contains fairly extensive tables of benchmark results for random graphs (including comparisons against two old algorithms), with which we compare our algorithm. The results are presented in Table 1. The algorithm was tested on random graphs with given edge densities and randomly assigned integer weights between 1 and 10.

Table 1
Benchmark results (in CPU seconds)

Vertices	Edge density	New	[3]
100	0.1	0.01	0.06
100	0.5	0.01	0.27
100	0.9	2.68	1.84
200	0.1	0.01	0.26
200	0.5	0.08	5.76
200	0.8	70.06	536.52
300	0.1	0.02	0.75
300	0.5	0.50	48.73
300	0.7	50.67	2590.35
500	0.1	0.05	2.32
500	0.5	7.60	842.21
1000	0.1	0.27	13.25
1000	0.3	3.50	503.91
1500	0.1	0.78	43.42
1500	0.2	2.87	373.13
2000	0.1	1.70	106.47
2000	0.2	8.32	1258.88

Due to the fact that different computers were used in the experiments—we used a 233 MHz PC and a workstation, model HP9000/835, was used in [3]—it is difficult to compare the results. A rough estimation, however, that our computer is ten times faster gives a clear indication: the new algorithm is faster than the old, except in the case of small graphs with high density. A similar conclusion was drawn in [9] for the analogous algorithm for unweighted graphs.

An implementation of the algorithm described, written in C, is available at <URL:http://www.tcs.hut.fi/∼pat/wclique.html>.

References

[1] L. Babel, A fast algorithm for the maximum weight clique problem, Computing 52 (1994) 31–38.

[2] E. Balas and H. Samuelsson, A node covering algorithm, Naval Res. Logist. Quart. 24 (1977) 213–233.

[3] E. Balas and J. Xue, Minimum weighted coloring of triangulated graphs, with application to maximum weight vertex packing and clique finding in arbitrary graphs, SIAM J. Comput. 20 (1991) 209–221; and 21 (1992) 1000.

[4] E. Balas and J. Xue, Weighted and unweighted maximum clique algorithms with upper bounds from fractional coloring, Algorithmica 15 (1996) 397–412.

[5] R. Carraghan and P.M. Pardalos, A parallel algorithm for the maximum weight clique problem, Technical Report CS-90-40, Dept. of Computer Science, Pennsylvania State University, 1990.

[6] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-completeness (Freeman, New York, 1979).

[7] E. Loukakis and C. Tsouros, An algorithm for the maximum internally stable set in a weighted graph, Int. J. Comput. Math. 13 (1983) 117–129.

[8] G.L. Nemhauser and L.E. Trotter, Vertex packings: Structural properties and algorithms, Math. Programming 8 (1975) 232–248.

[9] P.R.J. Östergård, A fast algorithm for the maximum clique problem, submitted for publication.

[10] P.M. Pardalos and N. Desai, An algorithm for finding a maximum weighted independent set in an arbitrary graph, Int. J. Comput. Math. 38 (1991) 163–175.

[11] P.M. Pardalos and J. Xue, The maximum clique problem, J. Glob. Optim. 4 (1994) 301–328.