AUSTRALASIAN JOURNAL OF COMBINATORICS
Volume **32** (2005), Pages 253–257

A note on finding a maximum clique in a graph using BDDs

Mukul Subodh Bansal V.Ch. Venkaiah

International Institute of Information Technology
Hyderabad
India
bansal@iastate.edu venkaiah@iiit.net

Abstract

A new approach for the solution of the maximum clique problem for general undirected graphs was presented in the paper titled "Finding the Maximum Clique in a Graph Using BDDs", by F. Corno, P. Prinetto and M. Sonza Reorda, in 1993. The approach is based on computing the characteristic function of all the completely connected components in the graph, and then finding the maximum cost satisfying assignment of such a function. The novelty of the method is in the use of Binary Decision Diagrams (BDDs) for representing and manipulating characteristic functions.

However, we observe that the algorithm stated does not produce the right answer for certain graphs. In this paper we show how to overcome this inadequacy and improve the algorithm.

1 Introduction

Let an undirected graph be denoted by $G = (V, E)$, where V is the set of vertices and E is the set of edges. Let n denote the cardinality of V. A *clique* of a graph is a set of vertices, any two of which are adjacent. A largest clique is one with the maximum number of vertices. Note that there may be more than one such clique. The Maximum Clique (MC) problem is the problem of finding out a largest clique in the given graph. Finding a maximum clique in a graph is known to be an NP-complete problem.

Corno et al. presented a nice approach in [1] to find a maximum clique in a graph using BDDs. However, we observe that their algorithm need not always produce the right answer. The main contribution of this paper is a correction in the algorithm which overcomes this inadequacy.

Note that the error in [1] appears to lie only within the text of the paper. The cliques computed in the experimental section are consistent and correct. The error lies in section 2.3 of [1]. The problem can be corrected if at each step, we consider all

the missing edges in the sub-graph that we consider, and not the points outside the sub-graph. The authors of [1] have also published a more recent paper, [2], which presents a new approach for finding the maximum clique in realistic graphs. The algorithm is built around a classical branch-and-bound, but exploits the efficiency of Binary Decision Diagrams and Symbolic Techniques to avoid explicit enumeration of the search space. They also show how a well known heuristic based on an approximate coloring of the graph can be computed symbolically and be used to effectively limit the search space.

This paper is organized as follows. The reasons for modifying this algorithm are given in Section 2. In Section 3 we correct the mistake and prove its correctness. Concluding remarks are made in Section 4.

2 Need for Modification

The Corno et al algorithm evolves in three different stages [1]. First the Basic algorithm is presented. Then a bounding condition is introduced which speeds up the algorithm. Lastly an improved algorithm is proposed, to further speed up the computation. The problem, which this paper attempts to address, has been found in the second part i.e. the bounding condition algorithm.

The algorithm for the bounding condition can be shown to be inadequate. The problem lies in the assumption that, "The function f^k_{CCC} represents all the Completely Connected Components (CCCs) built over the vertices whose degree is $\geq k$". Since f^k_{CCC} does not consider the missing edges with weight $\geq k$, it does not always represent CCCs.

The example given below shows that the subset of vertices returned by the algorithm might not even be a clique.

2.1 Counter Example

Consider the graph given in Figure 1:

Here, $n = 6$,

Hence, initially $k = 6$

The missing edges are $\{v_1, v_5\}$, $\{v_2, v_6\}$, $\{v_3, v_6\}$, $\{v_4, v_6\}$

Their weights are 4,2,2 and 2 respectively.

Let us study how the algorithm proceeds. First it calculates f^6_{CCC},

$f^6_{CCC} = \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4} \cdot \overline{x_5} \cdot \overline{x_6} \cdot \hat{f}^6_{CCC}$

Thus $m = 0$

Then it calculates f^5_{CCC},

$f^5_{CCC} = \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4} \cdot \overline{x_5} \cdot \overline{x_6} \cdot \hat{f}^5_{CCC}$

Thus $m = 0$

Then it calculates f^4_{CCC},

$f^4_{CCC} = \hat{f}^4_{CCC} \cdot \overline{x_6}$

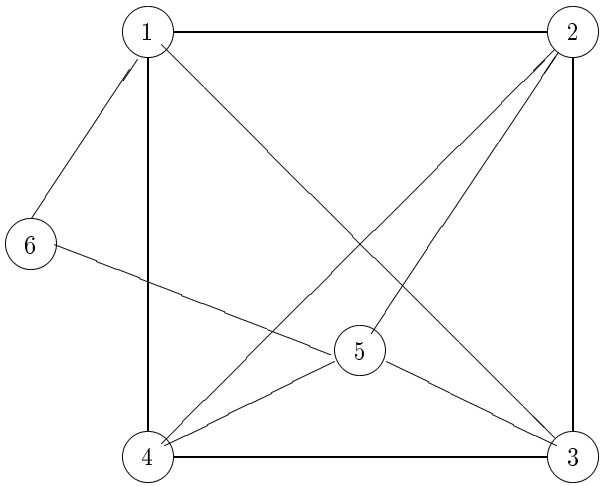

Figure 1: Counter Example

or, $f^4_{CCC} = \overline{x_2 \cdot x_6} \cdot \overline{x_3 \cdot x_6} \cdot \overline{x_4 \cdot x_6} \cdot \overline{x_6}$

The minimal term will thus be $\overline{x_6}$,
The Clique obtained is thus $\{v_1, v_2, v_3, v_4, v_5\}$
However this is not correct since v_1 and v_5 are not connected.
Now $m = 5$, hence the algorithm terminates.
The answer given by the algorithm is wrong.

3 Corrected and Improved Algorithm

Based on the observation that function f^k_{CCC} does not always represent all the CCCs built over the vertices whose degree is $\geq k$, a modified algorithm for the bounding condition is proposed. The algorithm is as follows.

Compute a partial function f^k_{CCC}, which is equivalent to computing the function f_{CCC} for the sub-graph of G composed of all those vertices whose degree is $\geq k$, as in the basic algorithm. For this partial function f^k_{CCC}, compute the maximum clique as in the basic algorithm. The partial function f^k_{CCC} is computed for decreasing values of k, starting from $k = n - 1$ until $m > k$. m denotes the maximum clique size obtained in the previous steps. This is a bounding condition as any vertex whose degree is less than m has a number of adjacent edges too small to enable it to be a part of a CCC bigger than the one already found. This is because each vertex in a clique of size m must have degree at least $m - 1$. When the condition is verified, the algorithm can thus be stopped.

Note that, unlike in the Corno et al bounding condition algorithm, in this algorithm, at each step, we consider all the missing edges in the sub-graph that we consider. We do not consider the points outside the sub-graph.

3.1 Example

As an example consider again the graph in Figure 1:

Proceeding according to the corrected algorithm, we first compute f^5_{CCC}. However, there are no vertices with degree ≥ 5. So, we consider f^4_{CCC}. Vertices $\{v_1, v_2, v_3, v_4, v_5\}$ have degree ≥ 4. Hence, we shall consider the sub-graph with vertices $\{v_1, v_2, v_3, v_4, v_5\}$. In this sub-graph the only missing edge is $\{v_1, v_5\}$. Thus,

$$f^4_{CCC} = \overline{x_1 \cdot x_5} \text{ or,}$$

$$f^4_{CCC} = \overline{x_1} + \overline{x_5}.$$

The maximum clique for this step is thus given by the vertices $\{v_1, v_2, v_3, v_4\}$ and $\{v_2, v_3, v_4, v_5\}$. So, $m = 4$. We now go to the next step and compute f^3_{CCC}. However, since $m > k$ we can terminate the algorithm here.

Thus, the maximum clique size is 4, and both $\{v_1, v_2, v_3, v_4\}$ and $\{v_2, v_3, v_4, v_5\}$ are examples of the maximum clique for the graph. It can easily be verified that this answer is indeed correct.

3.2 Proof of Correctness

We will now prove that this algorithm is correct. Suppose that the maximum clique size in the given graph is MAX. According to the algorithm, a clique of size MAX will be found while computing the function f^{MAX}_{CCC}. So, if the algorithm does not terminate before calculating the function f^{MAX}_{CCC}, then we will get the right answer. The algorithm terminates when $m > k$. The maximum value of m for the given graph is MAX. Thus, in the worst case, the algorithm will terminate when $MAX > k$. So, the maximum value that k can have is $MAX - 1$. This means that the algorithm will not terminate until it has calculated f^{MAX}_{CCC}, immediately after which the next step is to compute f^{MAX-1}_{CCC}, and the algorithm terminates. This completes the proof.

3.3 Another Improvement

We make a suggestion for further improvement in the Improved Algorithm (Section 2.3). The algorithm describes a pruning condition according to which ,"if the sub-graph to be considered at one step is smaller than the size of the MC found in the previous steps, then the step can be skipped". This condition can be made stronger by observing that even if the sub-graph to be considered at one step is equal in size to the MC found in previous steps, it cannot yield a clique greater in size than the MC already found.

The new pruning condition can thus be written as: If the sub-graph to be considered at one step is smaller than or equal in size to the MC found in the previous steps, then the step can be skipped.

4 Conclusion

A new approach to solve the maximum clique problem was proposed in the paper by Corno et al. [1] in 1993. They solve the problem by computing the characteristic function of all the Completely Connected Components in the graph; A maximum clique is then found by determining the maximum-cost satisfying assignment for such a function. The novelty of the approach is mainly in the use of BDDs, the state-of-the-art data structure for Boolean functions representation and manipulation.

However, there was an inadequacy in the algorithm proposed. We point out this mistake and present the corrected algorithm. The application of the improved algorithm using the modified bounding condition algorithm can give the maximum clique of a graph correctly and efficiently.

References

[1] F. Corno, P. Prinetto and M. Sonza Reorda, "Finding the Maximum Clique in a Graph using BDDs", *ICVC'93: IEEE 3rd International Conference on VLSI and CAD,* (Taejon, Korea, November 1993), 269–272.

[2] F. Corno, P. Prinetto, M. Sonza Reorda, "Using Symbolic Techniques to find the Maximum Clique in Very Large Sparse Graphs", *EDTC'95: IEEE European Design and Test Conference,* (Paris, March 1995).

[3] R.E. Bryant, "Graph-Based Algorithms for Boolean Function Manipulation", *IEEE Transactions on Computers,* **C-35** (8), (1986), 677–691.

[4] K.S. Brace, R.L. Rudell and R.E. Bryant, "Efficient Implementation of a BDD Package", *DAC90: 27th ACM/IEEE Design Automation Conference,* (1990), 40–45.

[5] L. Babel and G. Tinhofer, "A branch and bound algorithm for the maximum clique problem", *ZOR-Methods and Models of Operations Research*, **34** (1990), 207–217.

(Received 22 Oct 2003; revised 27 July 2004)