Cliques via Cuts: Refuting the Strong Exponential Time Hypothesis with a Subexponential Algorithm for Clique

Alexander Bistagne
Independent Research 2023

July 28, 2023

Abstract

This paper presents a short algorithm which finds a max clique in time $2^{O(n^{\frac{2}{3}}log(n))}$. Rather than try and improve the average runtime of the algorithm, this paper attempts to cut all the unnecessary heuristics from the algorithm I originally formulated in my research.

1 Introduction

Off and on now for a while, I have been researching clique problems and PvsNP more generally. The most recent endeavor into Clique research was focused on the idea that if one could find good algorithms for clique in both the dense case and sparse case, one might be able to combine them into a good general algorithm for clique. The following algorithm combines one sparse strategy -the inclusion-exclusion principle- with a well known dense strategy -co-disconnected graphs- and a new dense strategy: balanced cuts in the complement.

2 Preliminaries

This paper assumes a basic knowledge of graph theory for sake of brevity. (connectedness, complement, minimum degree, neighborhoods, cuts, trees and edges)

Definition 2.1. *An algorithm has subexponential time if it runs in time* $2^{o(n)}$

Conjecture 2.2. *The strong exponential time hypothesis[CITATION] claims at least the following: there is no subexponential time algorithm for k-SAT for all* $k \geq 3$.

Theorem 2.3. *The strong exponential time hypothesis holds if and only if there is no subexponential algorithm for clique [CITATION]*

Definition 2.4. *A Gomory-Hu Tree[CITATION] is a weighted spanning tree of a graph where the weight of minimum st cuts in T match the minimum weight of st cuts in the original graph*

Theorem 2.5. *A Gomory-Hu Tree always exists and can be constructed in polynomial time [CITATION]*

Theorem 2.6. *Every tree has a balanced cut vertex and it can be found in polynomial time[CITATION]*

Definition 2.7. *Clique: a clique is an induced subgraph of some other graph which is a complete graph. The complement of a clique is an independent set. A clique is maximal if it is not the induced subgraph of some other clique in the original graph*

Theorem 2.8. *All Maximal cliques (or Independent sets) can be listed in time polynomial per clique (or independent set)[[CITATION]*

Definition 2.9. *The Maximum Clique Search Problem is the problem of finding the largest clique contained in a finite simple undirected graph.*

3 Algorithm

Remark 3.1. *For sake of brevity we will assume all basic data manipulations on sets: union, intersection, size take linear time.*

Theorem 3.2. *Algorithm 1 Cliques via Cuts is a subexponential algorithm for the maximum clique search problem*

Lemma 3.3. *Lines 2-6 of algorithm 1 are correct, and have worst case recursive structure* $T(n) \leq T(n-1) + O(n^2)$

Proof. We will begin by proving the correctness of lines 2-6 of the algorithm. The maximum independent set on a disconnected graph is clearly the disjoint union of the maximum independent set on each of the components. Thus, the maximum clique is the disjoint union of the maximum clique on each of the co-components. The runtime for this case is $T(n) \leq T(a) + T(b) + O(n^2)$ where a + b = n if there are only two components a similiar structure if there are more because the connectivity test takes at most quadratic time. In the worst case, b= 1 and this is a chip and conquer recursive relation as explained. □

Lemma 3.4. *Lines 7-11 of algorithm 1 are correct and have worst case recursive structure* $T(n) \leq T(n-1) + T(n - \sqrt[3]{n}) + O(n^2)$

Proof. Here is the inclusion exclusion principle. Either the vertex v is in the maximum clique and we can restrict our search to its neighborhood or it is not in the maximum clique and we can safely remove it without effecting the size of the maximum clique. Clearly if we explore both cases, the correct clique is the larger one. Now the degree bound gets us a runtime guarantee of at most $T(n) \leq T(n-1) + T(n - \sqrt[3]{n}) + O(n^2)$ because the connectivity test and degree checks take at most quadratic time. □

Algorithm 1 Cliques via Cuts

procedure CLIQUE (G)
  if $\bar{G}$ is disconnected **then**
    max_clique = []
    for C connected component of $\bar{G}$ **do**
      max_clique append CLIQUE(C)
    return max_clique
  if $\delta(G) < n - \sqrt[3]{n} - 1$ **then**
    Let v have degree = $\delta(G)$
    a = CLIQUE($G \setminus v$)
    b = CLIQUE($G \cap N(v)) \cup v$
    return $|a| \geq |b|$? a : b
  $T \leftarrow$ Gomory_Hu_Tree($\bar{G}$)
  $v \leftarrow$ BalancedCutVertex(T)
  C $\leftarrow$ []
  for edge uv in T **do**
    Let *cut* be the edges represented by uv
    $C \leftarrow C \cup cut$
  Let H = (V(G) , C)
  max_clique $\leftarrow$ []
  for maximal independent set I in H **do**
    current_clique = CLIQUE(G[I])
    if size(current_clique) > size(max_clique) **then** max_clique $\leftarrow$ current_clique
  return max_clique

Lemma 3.5. *In line 21 of Algorithm 1, G[I] always has co-components of size at most* $\frac{n}{2}+1$

Proof. $G[I]$ is an induced subgraph of $G \cup H$ proof:$(G \cup H)[I] = G[I] \cup H[I] = G[I] \cup (I, \emptyset) = G[I]$ $G \cup H$ is co-disconnected with co-components of size at most $\frac{n}{2}+1$ proof: The cuts which isloate v in T must also isolate v in G by the gomory hu tree correspondence, and furthermore the size of the graph on the not v side of a cut in T is the same size as the not v side of a cut in G. Thus since the cuts isolate v in T and make each component at most size $\frac{n}{2}+1$. These edges make $G \cup H$'s co-components at most size $\frac{n}{2}+1$ which makes the co-components of G[I] at most size $\frac{n}{2}+1$ □

Lemma 3.6. *In line 20 of algorithm 1, there are at most* $2^{n^{\frac{2}{3}}}$ *such maximal independent sets.*

Proof. Because to reach line 20, line 7 must be false, We know that when line 20 is reached, the minimum degree of G is at least $n - \sqrt[3]{n} - 1$ Thus the maximum degree of $\bar{G}$ is at most $\sqrt[3]{n}$. Because the maximum degree bounds the minimum degree and the minimum degree bounds the minimum cut, we know that the edge weight of T is at most $\sqrt[3]{n}$ and because T is a spanning tree of $\bar{G}$, the degree of v is at most $\sqrt[3]{n}$ in T. Therefore the number of edges in H is at most $n^{\frac{2}{3}}$. Thus, by choosing only one vertex of each edge we can get an upper bound on the number of maximal independent sets as $2^{n^{\frac{2}{3}}}$. □

Lemma 3.7. *Lines 12-23 of algorithm 1 are correct and imply recursion* $T(n) \leq 2^{n^{\frac{2}{3}}}(T(n/2+1) + O(n^2)) + poly(n)$

Proof. Because H is a cut of $\bar{G}$ it is a subgraph; thus, G is a subgraph of $\bar{H}$. So the maximum clique of G is a subclique of a maximal clique of $\bar{H}$ which is identical to a maximal independent set of H. Thus by checking every maximal independent set of H we must find the maximum clique of G. Now combining this correctness with lemmas 3.5 and 3.6, and the nature of this third case always feeding back into the first case. Theorems 2.5 and 2.6 bound our work per case on line 12 and 13, Remark 3.1 bounds 14-19 & 22-23 and we get the third case. □

Proof. Now we can finish theorem 3.2. By combining the cases it should be clear, that this algorithm is dominated by the recursion tree size, and on their own each case implies a subexponential recursion tree of size $2^{O(n^{\frac{2}{3}} log(n))}$ time. Since the recursion cases have no ability to interact to make large subproblems, the theorem follows. □

Corollary 3.8. *The Exponential time Hypothesis is False*

Proof. Take theorem 3.2 and combine with theorem 2.3 □

4 Conclusion

I know that claiming this disproof is an very high stakes claim; therefore, if this paper is wrong, please politely refute. Regardless, I think this paper presents a simple interesting algorithm. I think a lot of interesting heuristics could fit in between lines 6 and 7 of the algorithm, but that was not the goal with this paper. The goal of this paper was to demonstrate competency in comutational complexity and related topics.