Average-case bounds on detecting large sets of cliques, in small graphs

Josh Burdick (`josh.t.burdick@gmail.com`)

February 23, 2026

Abstract

Shannon's counting argument [1] shows that many functions are hard to compute. However, it is nonconstructive, and so doesn't explicitly state a hard-to-compute function. Previously, we tried to adapt that counting argument to functions *vaguely* related to CLIQUE [2].

Here, we consider a weaker question: how hard is it to detect *most* of the cliques in a graph? We combine the counting argument with random restrictions, using linear programming for tiny graphs ($n = 8, k = 3$). Although we don't get a bound on CLIQUE, we do get average bounds on the complexity of detecting *most* of the cliques in a graph. There is a trade-off: the more "large" sets of cliques we try to detect, the weaker the bound we get. The bound essentially vanishes at CLIQUE.

Previously, we used a counting argument to get an average-case lower bound on the size of the smallest circuit which detects some subset of cliques in a graph [2]. This didn't give a bound for $CLIQUE$; that function was present in the "zeroing-out" graph Z, but it was only "above" a tiny number of other functions.

What if we ask a weaker question: is detecting *more* cliques harder than detecting *fewer* cliques, on average? Based on the structure of Z, this seems plausible. Even if we don't know how many gates correspond to one arrow in Z, the arrows are still all pointing up!

Thus, here we try to get a bound on the number of gates needed to detect the top p fraction of cliques, for $p = 1, 1/2, 1/3, \ldots$.

1 A linear program

We phrase the question as a linear program. First, we define the variables.

We consider a graph $G = (V, E)$ with $n = |V|$ vertices and $m = \binom{n}{2}$ possible edges. We are interested in detecting cliques of size k; there are $N = \binom{n}{k}$ possible cliques. Let C be the set of all cliques of size k in G.

For a given set of cliques $A \subseteq C$, let $\mathcal{C}_A$ be the smallest circuit which detects the cliques in A (and ignores the others). We write $|\mathcal{C}_A|$ for the number of gates in $\mathcal{C}_A$.

We assume that the range of number of possible cliques, from 0 to N, is divided into L layers, where each layer has size $\approx N/L$. Let L_i be the sets of cliques in layer i, for $i = 1, \ldots, L$.

We only consider circuits with up to some number of gates G. (If G is too small, the LP will be infeasible, because we won't be able to include all the possible sets of cliques.)

We also will group sets of cliques by how many vertices they have. Let V_j be the sets of cliques which contain exactly j vertices, with $k \leq j \leq n$. (For a set V_j, we require that some clique includes each of the j vertices.)

The main variables are $x_{l,v,g}$ for $l \in \{1, \ldots, L\}$, $v \in [k, n]$, and $g \in \{1, \ldots, G\}$. Here, l is the layer, v is the number of vertices, and g is the number of gates.

1.1 Weighted averages

We will need the expected number of gates, for each (l, v) pair. First, we define $w_{l,v}$ to be the number of sets of cliques which are in layer l and have v vertices. Note that sets of cliques with more vertices *tend* to be in the higher layers. However, it is certainly possible for a small set of cliques (in a low layer) to have many vertices, and vice-versa.

We then add variables for the average number of gates in each of these groups.

$$|E[|C_{V_v \cap L_l}|] = \frac{1}{w_{l,v}} \left(\sum_{g=1}^{G} g x_{l,v,g} \right)$$

We also define variables for the marginal distributions for each number of vertices, $E[|\mathcal{C}_{V_v})]$, and each layer, $E[|\mathcal{C}_{L_l})]$. These are defined using constraints based on the weights $w_{l,v}$, in the obvious way.

1.2 Counting constraints

We apply the Shannon counting argument: for a given number of gates g, there are a limited number of functions which can be computed. Clearly, more functions can be expressed with more gates, in a way which depends on the basis (we might visualize this as a conical martini glass). Thus, for a given number of gates g, we have:

$$\sum_{l,v} x_{l,v,g} \leq \text{(number of functions with g gates)}$$

This will depend on the basis used; here, we only use 2-input NAND gates.

1.3 Zeroing constraints

Suppose we have a set of cliques $A \subseteq V_n$, and we restrict all the edges in some vertex to be 0. The resulting graph has $n - 1$ vertices. Since we're using NAND gates, we know that we "hit" least one NAND gate. Thus, we have the constraint:

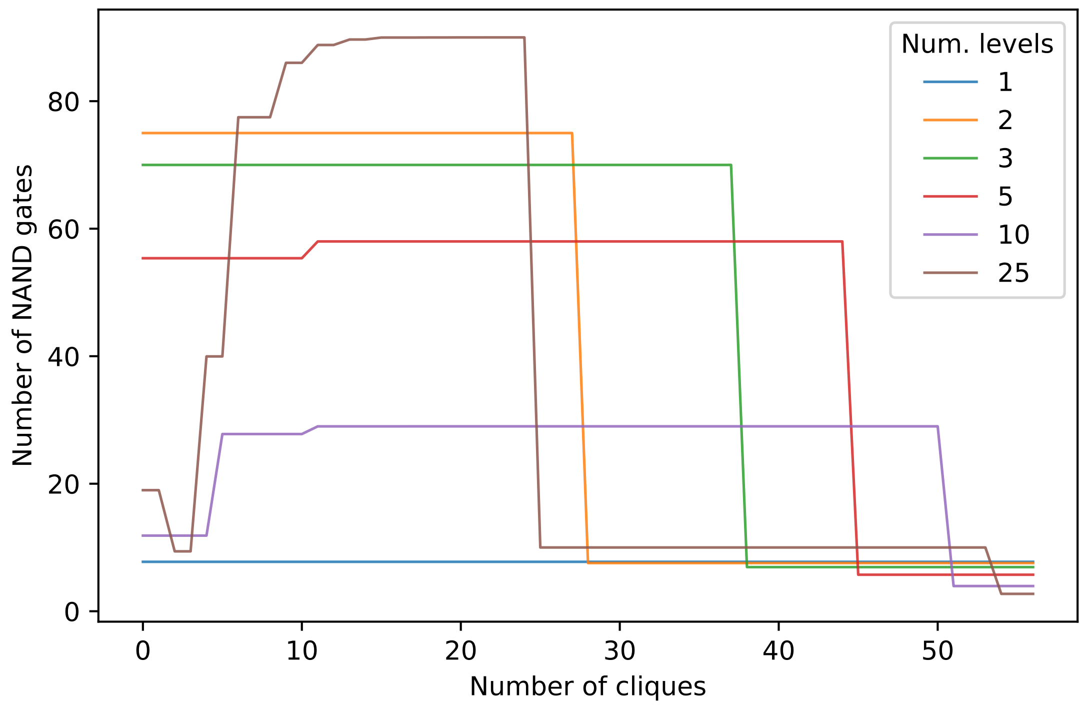

Figure 1: Bounds for layers of $BUGGYCLIQUE$ with $n = 8$ and $k = 3$, with the number of cliques divided into different numbers of layers. In each case, the LP minimized the *average* number of gates in the highest layer; the bounds for the other layers show one possible feasible solution for the LP.

$$E[|\mathcal{C}_{V_v}|] \geq E[|\mathcal{C}_{V_{v-1}}|] + 1$$

On its own, this constraint is not very useful. For instance, it only implies a lower bound of $n - k$ gates for any of the functions in $BUGGYCLIQUE$, which is not very interesting.

2 Results

Having set up the LP, we minimize the number of gates in the highest layer, $E[|\mathcal{C}_{L_L}|]$ for different numbers of layers L, using up to 90 gates.

The results are shown in Figure 1. Note that what was minimized was only the average number of gates in the highest layer. The bounds for the other layers are just one possible feasible solution to the LP.

(FIXME Emphasize the right-most line for each level, e.g. by making it darker and/or thicker).

It may look incorrect, or at least surprising, that the LP is conjecturing that finding small sets of cliques is in fact harder. One explanation would be that as the LP pushes the average for the functions with many cliques down, it "pushes out" the functions with few cliques, making it appear that they are harder to compute. (Indeed, it seems possible that adding the naive upper bound, for the lower layer, would improve the bounds for the top layer.)

At the bottom-right corner, it looks like the bound for the top *third* of the cliques is near the lower bound for the entire set of $BUGGYCLIQUE$ functions. However, the bound drops off rapidly as we increase the number of layers.

3 Conclusion

It appears that we can get a reasonable bound for, e.g., the top third of the cliques, but the bound drops off rapidly as we increase the number of layers.

We emphasize that these are *average-case* bounds. $CLIQUE$ is only one of a very large number of functions, and so even when the bound on the highest layer is large-ish, $CLIQUE$ might still be easy to compute.

The motivation for using "layers" was to get a non-trivial bound. However, the resulting LP has many fewer variables than previous attempts (which included a set of variables for $0..\binom{n}{k}$ cliques), and so is much faster to solve (which is a nice side effect).

4 Acknowledgements

This was written with the assistance of Google Antigravity. However, the author is responsible for any errors or omissions.

References

[1] Claude E Shannon. The synthesis of two-terminal switching circuits. *Bell System Technical Journal*, 28(1):59–98, 1949.

[2] Josh Burdick. How hard is it to detect *some* cliques? *SIGACT News*, 55(2):38–52, Jun 2024.