A function is monotone if increasing the value of an input can never cause the value of an output to decrease. More formally, $f\colon \{0,1\}^n \to \{0,1\}$ is *monotone* if whenever $x_i \leq y_i$ for all i, it is also true that $f(x) \leq f(y)$.

An example is the majority function. Another example is graph connectivity: Given an undirected graph G represented by its adjacency matrix, is it connected? In this example increasing the value of an input means adding an edge to the graph and this can never cause the graph to disconnect.

An AND/OR circuit is monotone if it does not use any of its negated literals. Any monotone function can be expressed as an OR of ANDs of positive literals and can therefore be computed by a monotone DNF. In this lecture we will be interested in the size of the smallest monotone circuit for a given (monotone) function. The circuit for majority sketched in Lecture 3 was monotone, so $MAJORITY$ on n bits has a monotone circuit of size $n \log n$.

Graph connectivity also has a monotone circuit of size polynomial in the number of vertices. One way to construct this circuit is by emulating the standard breadth first search algorithm for connected components. In stage i, the circuit marks those vertices at distance at most i from some starting vertex. A vertex in stage $i+1$ should be marked if and only if it was marked in stage i or it is connected to a stage i vertex by an edge. Neither of these steps requires the use of a negation. The graph is connected if all vertices in stage n are marked, which does not require a negation.

1 Monotone circuits and slice functions

The relation between monotone and non-monotone circuit complexity is subtle. To begin with, we will show a positive connection: Every non-monotone circuit can be simulated, in some sense, but a small family of monotone circuits of similar size, each one computing one "slice" of the same function.

A *slice function* is a function whose domain consists of all binary strings of length n and Hamming weight w for some n and w. We say a slice function f is computed by circuit C with n inputs if C agrees with f on all inputs of Hamming weight w (but may output arbitrary values on other inputs).

Claim 1. *If a slice function can be computed by an AND/OR circuit of size s and depth d then it can be computed by a monotone AND/OR/MAJORITY circuit of size $s + O(n)$ and depth $d+1$ with all majority gates in the bottom layer.*

Proof. It is enough to show that for all i the negative literal $\overline{x}_i$ can be computed as a $MAJORITY$ of positive literals. Indeed, $\overline{x}_i$ takes value 1 if and only if at least w of the literals $x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$ take value 1; this forces x_i to take value 0. This computation can be performed by a $MAJORITY$ gate after replacing some inputs by constants. □

From this claim we can conclude that for every function $f\colon \{0,1\}^n \to \{0,1\}$ that has a size s AND/OR circuit there exist AND/OR monotone circuits $M_0, \ldots, M_n$ of size $s + O(n^2 \log n)$ each such that M_i computes the i-th slice of f.

2 Cliques are hard for monotone circuits

The main result in this lecture is that the $CLIQUE_k$ function on n-vertex graphs requires monotone circuits of size $n^{\Omega(\sqrt{k})}$ (when $k \leq n^{1/4}$), regardless of their depth. This is an explicit function that requires monotone circuits of size exponential in some root of its input length.

A *k-clique* in an n-vertex graph is a complete subgraph on k of its vertices. The $CLIQUE_k$ function takes an n-vertex graph as its input and indicates the existence of a k-clique in it, namely

$$CLIQUE_k(G) = \begin{cases} 1, & \text{if } G \text{ contains a } k\text{-clique} \\ 0, & \text{if not.} \end{cases}$$

The graph G is specified by its adjacency matrix, or equivalently by an $\binom{n}{2}$-bit string $x = (x_{ij}), i < j$, where x_{ij} indicates the presence of an edge between vertices i and j.

The function $CLIQUE_k$ can be computed by a monotone DNF of size $\binom{n}{k}$. This DNF has a term for each subset S of vertices. The term accepts if all possible $\binom{k}{2}$ edges between vertices in S are present. It is not difficult to show that any DNF for CLIQUE must have size $\binom{n}{k}$. We will show that an arbitrary monotone circuit cannot do much better.

Theorem 2. *Assuming $k \leq n^{1/4}$, any monotone AND/OR circuit for $CLIQUE_k$ has size $n^{\Omega(\sqrt{k})}$.*

Clique indicators and special DNFs

To warm up towards the proof of Theorem 2 let us prove that $CLIQUE_k$ requires large monotone DNFs, but in a different way than usual. Let's say a graph is a *proper clique* if it consists of a clique on some subset of its vertices and no other edges. Suppose ϕ is a monotone DNF of size s computing $CLIQUE_k$. Then ϕ must, in particular, accept all $\binom{n}{k}$ proper k-cliques. A term of ϕ might look like

$$x_{12}x_{14}x_{24}x_{16}$$

If we replaced this term by

$$x_{12}x_{14}x_{24}x_{16}x_{26}x_{46}.$$

then the behavior of this term on proper k-cliques P doesn't change: Both terms accept those P that contain vertices 1, 2, 4, and 6 and reject all other P. By monotonicity it follows that all graphs with a k-clique are still accepted after the modification, and since we only added variables to terms all graphs without a k-clique are still rejected by the DNF.

A *clique indicator* K_W is an AND of variables x_{ij} over i, j in W. In general, if we replace each term in ϕ by the smallest clique indicator implied by it, ϕ still computes the $CLIQUE_k$ function. So we may assume, without loss of generality, that each term of ϕ is a clique indicator. We will call a DNF in which every term is a clique indicator a special DNF.

Now suppose P is a random proper k-clique, chosen uniformly among all $\binom{n}{k}$ possibilities. For K_S to accept P, S must be a subset of P. This occurs with probability at most

$$\frac{k}{n} \cdot \frac{k-1}{n-1} \cdots \frac{k-w}{n-w} \leq \left(\frac{k}{n}\right)^{w+1}, \tag{1}$$

if the size of S is more than w. So if we remove all terms K_S with $|S| > k-1$ from ϕ, the resulting DNF ϕ' still accepts a $(1 - s \cdot (k/n)^k)$-fraction of proper k-cliques. If this number is strictly less than one then ϕ' must accept at least one proper k-clique, so at least one clique indicator must have survived the removal. But this clique indicator also accepts some proper clique of size $k-1$,

and so does ϕ'. Since ϕ is an OR of ϕ' and other terms, it too must accept this input, contradicting the fact that it computes $CLIQUE_k$. It follows that s must be at least $(n/k)^k$.

To summarize this proof, first we argued that if a small monotone DNF computes $CLIQUE_k$ then so does a small special DNF. We then showed that the wide clique indicators can be eliminated from this special DNF with little effect on the proper k-cliques. But such a DNF cannot represent $CLIQUE_k$ as it also accepts graphs that do not have a k-clique.

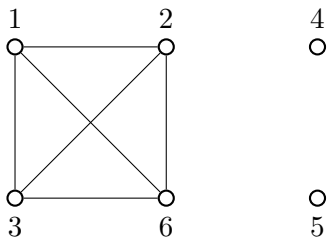
Approximating circuits by special DNFs

To prove Theorem 2 we would like to argue inductively that every gate in a small monotone circuit for $CLIQUE_k$ must compute a small and narrow special DNF. The natural strategy is induction: It would be great if we could say that if circuits C and C' have such representations, then so do C OR C' and C AND C'. (We assume that the gates have fan-in 2.) Unfortunately these operations preserve neither the size nor in the case of AND the structure of the representation. What we will be able to argue, however, is that C OR C' and C AND C' can be *approximated* (in a suitable sense) by small, narrow special DNFs.

The notion of approximation is different for positive and negative instances. Let us give a general definition. Let P and N be two distributions on $\{0,1\}^n$ such that $CLIQUE_k(P) = 1$ and $CLIQUE_k(N) = 0$ with probability one.

Definition 3. We say that $\phi\colon \{0,1\}^n \to \{0,1\}$ *approximates* $C\colon \{0,1\}^n \to \{0,1\}$ with respect to P and N if $\Pr_P[\phi(P) \geq C(P)] \geq 1-\varepsilon^+$ and $\Pr_N[\phi(N) \leq C(N)] \geq 1-\varepsilon^-$. The parameters ε^+ and ε^- are the *positive* and *negative approximation error*, respectively.

In our case P is a uniformly random proper k-clique. Here is what a sample of P looks like for $n = 6, k = 4$:



We will describe N shortly. Let us first see how an OR gate affects the complexity of our special DNF approximators.

OR gates Let us first look at a gate of the form C OR C'. Assuming that C and C' are approximated by special DNFs ϕ and ϕ', C OR C' will be approximated by the special DNF ϕ OR ϕ'; the approximation error simply add up. However, the size of the DNF doubles for each gate so it may grow exponentially with the depth of the circuit. What can we do to control this growth?

Let's look at an example. Suppose that at some point we end up with the following special DNF:

$$\phi = K_{123} \text{ OR } K_{2347} \text{ OR } K_{1245} \text{ OR } K_{12678}.$$

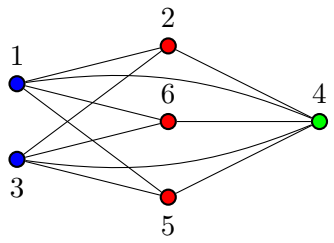
The sets 123, 1245, 1256, and 12789 have the structure of a sunflower: They all share the common core 12, but if the core is discarded the remaining petals are pairwise disjoint. If we replace

the corresponding clique indicators K_{123} OR K_{1245} OR K_{12678} by the indicator K_{12} of their core obtaining

$$\phi' = K_{12} \text{ OR } K_{2347}$$

only more positive instances will be accepted so the positive approximation error won't increase. However, removal of this terms might cause acceptance of some previously rejected negative instances. If we can choose the distribution on negative instances so that this is unlikely to happen then the negative approximation error might grow but not by much.

This brings us to the definition of the distribution N. To choose a random instance from N, we first select a coloring col that assigns to each vertex $v \in \{1, \ldots, n\}$ one of $k-1$ colors. The color of each vertex is chosen uniformly and independently among all possible colors. The graph N then consists of all edges among vertices of different colors. In every clique of N all vertices must have different colors, so N can never have a clique of size k. Here is an example with $n = 6, k = 4$:



What are the chances that a negative instance that was rejected by ϕ is accepted by ϕ', contributing to negative approximation error? For this to happen, the core 12 must form a clique (so vertices 1 and 2 must be assigned different colors) while none of the sets 123, 1245, 12678 can be a clique (so within each set there must be two vertices of the same color). We can calculate the error probability like this:

$$\begin{aligned}&\Pr_N[K_{12} \text{ AND } (\text{NOT } K_{123}) \text{ AND } (\text{NOT } K_{1245}) \text{ AND } (\text{NOT } K_{12678})] \\ &\quad = \Pr_N[K_{12}] \cdot \Pr_N[\text{NOT } K_{123} \mid K_{12}] \cdot \Pr_N[\text{NOT } K_{1245} \mid K_{12}] \cdot \Pr_N[\text{NOT } K_{12678} \mid K_{12}]\end{aligned}$$

because conditioned on the choice of colors of vertices 1 and 2, the events K_{123}, K_{1245}, and K_{12678}, become independent. We can now bound the individual terms like this:

$$\begin{aligned}\Pr_N[\text{NOT } K_{123} \mid K_{12}] &= \Pr[\text{col}(3) = \text{col}(1) \text{ OR } \text{col}(3) = \text{col}(2) \mid \text{col}(1) \neq \text{col}(2)] \\ &\leq \Pr[\text{col}(3) = \text{col}(1) \mid \text{col}(1) \neq \text{col}(2)] + \Pr[\text{col}(3) = \text{col}(2) \mid \text{col}(1) \neq \text{col}(2)] \\ &= \Pr[\text{col}(3) = \text{col}(1)] + \Pr[\text{col}(3) = \text{col}(2)] \\ &= \frac{2}{k-1}.\end{aligned}$$

By a similar calculation it follows that $\Pr[\text{NOT } K_{1245} \mid K_{12}] \leq 5/(k-1)$ and $\Pr[\text{NOT } K_{12678} \mid K_{12}] \leq 9/(k-1)$. So the probability of a negative approximation error is at most $2 \cdot 5 \cdot 9/(k-1)^3 \leq (9/(k-1))^3$ for this instance. This probability drops exponentially in the size of the sunflower.

We can generalize this calculation to prove the following claim. A *sunflower* is a collection of sets $S_1, \ldots, S_p$ such that $S_i \cap S_j = S_1 \cap \cdots \cap S_p$ for all $i \neq j$. This intersection is called the *core* of the sunflower.

Claim 4. *If $S_1, \ldots, S_p$ is a sunflower, each S_i has size at most w, and set elements are assigned one of $k-1$ colors independently at random, then the probability that all vertices in the core get different colors but each S_i has at least two vertices of the same color is at most $(\binom{w}{2}/(k-1))^p$.*

If w is at most $\sqrt{k-1}$ then this probability is at most 2^{-p}. So if we can ensure the existence of a sunflower of size p among the sets in the special DNF, a simplification step with negative approximation error at most 2^{-p} is always possible. It turns out that once the formula grows beyond a certain size such a sunflower always exists:

Lemma 5. *Every collection of more than $(p-1)^w w!$ nonempty sets, each of size at most w, contains a sunflower of size p.*

To summarize, an OR of two special DNFs is always a special DNF. If this DNF becomes very large then it must contain clique indicators of a large sunflower. If we replace these by a clique indicator of their core, the DNF shrinks in size, preserves the positive approximation error, and does not cause a large drop in the negative approximation error.

AND gates Now let's consider a gate of the form C AND C'. If ϕ and ϕ' are special DNFs approximating C and C' then we would like to approximate C AND C' by ϕ AND ϕ'. However, this is not a special DNF. Let's see what this formula looks like. For example, if

$$\phi = K_{12} \text{ OR } K_{235} \qquad \text{and} \qquad \phi' = K_{14} \text{ OR } K_{25}$$

then we can expand ϕ AND ϕ' as

$$\phi \text{ AND } \phi' = (K_{12} \text{ AND } K_{14}) \text{ OR } (K_{12} \text{ AND } K_{25}) \text{ OR } (K_{235} \text{ AND } K_{14}) \text{ OR } (K_{235} \text{ AND } K_{25}).$$

This is a DNF, but not a special one as its terms are not clique indicators anymore. If we proceed as in our warmup and replace every term by the smallest clique indicator implied by it, there will be no additional positive or negative errors:

$$\psi = K_{124} \text{ OR } K_{125} \text{ OR } K_{12345} \text{ OR } K_{235}.$$

This transformation has, however, two undesirable effects. First, the size of ψ is now the product of the sizes of ϕ and ϕ'. Second, the width grows as well: ϕ and ϕ' only involve cliques of size 4, while ψ involves a clique of size 5.

To reduce the width, we eliminate the terms involving cliques of size w or more. Removing terms can never cause acceptance of previously rejected instances, so this transformation preserves negative error. The probability of a positive error can be bounded as we did in our warmup calculation for DNFs. To reduce size, we can apply the same transformation as for OR gates.

Proof of Theorem 2

We now have all the pieces to prove Theorem 2. To every monotone circuit C of size at most c we will associate a special DNF ϕ of size at most s in which each term is an indicator of a clique of size at most w. The values of s and w will depend on n, k, and c and will be determined later.

The formula ϕ is defined inductively as follows: The approximator of a variable x_{uv} is the special DNF $\phi = K_{\{u,v\}}$. If

$$\phi = \text{OR}_{S \in \mathcal{S}} K_S \quad \text{and} \quad \phi' = \text{OR}_{S' \in \mathcal{S}'} K_{S'}$$

are approximators of C and C', then the approximator of ϕ OR ϕ' is the special DNF

$$\text{pluck}(\text{OR}_{S \in \mathcal{S} \cup \mathcal{S}'} K_S)$$

where $\text{pluck}(\phi)$ is the DNF obtained after replacing any OR of clique indicators K_{S_1} OR $\cdots$ OR K_{S_p} for a sunflower $S_1, \ldots, S_p$ of size p by a clique indicator $K_{S_1 \cap \cdots \cap S_p}$ for its core, repeated for as long as sunflowers of size p appear in ϕ.

Finally, the approximator of ϕ AND ϕ' is the DNF

$$\text{pluck}(OR_{S\in\mathcal{S},S'\in\mathcal{S}'}\ \text{trim}(K_{S\cup S'}))$$

where $\text{trim}(K_S)$ is K_S if S has size at most w and 0 (false) otherwise.

By Lemma 5, if we set $s = (p-1)^w w!$, the size of ϕ will remain bounded by s for otherwise ϕ would contain sunflowers of size p that are ripe for plucking. Also, for each clique indicator K_S in ϕ, S will have size bounded by w because larger cliques are trimmed.

Lemma 6. *The positive approximation error of ϕ is at most $cs^2(k/n)^{w+1}$.*

Proof. By our discussion in the previous section, positive approximation error can only be introduced by trimming. A trimming step introduces error only when the trimmed clique K_S is fully contained in P, namely when S is a subset of the vertices of P. By the calculation (1) this probability is at most $(k/n)^{w+1}$. Since each special DNF to which trimming is applied has at most s^2 terms and there are c gates in the circuit, by a union bound the positive approximation error is at most $cs^2(k/n)^{w+1}$. □

Lemma 7. *The negative approximation error of ϕ is at most $cs^2(\binom{w}{2}/(k-1))^p$.*

Proof. By our discussion in the previous section, negative approximation error can only be introduced by plucking. By Claim 4 each iteration of plucking introduces at most $(\binom{w}{2}/(k-1))^p$ negative error. Since each special DNF is plucked at most s^2 times and there are c gates in the circuit, by a union bound the negative approximation error is at most $cs^2(\binom{w}{2}/(k-1))^p$. □

To complete the proof of Theorem 2 we need one more simple claim.

Claim 8. *If ϕ is a nonzero special DNF in which each clique has size at most w then the probability that ϕ accepts a negative instance is at least $1-\binom{w}{2}/(k-1)$.*

We can now complete the proof of Theorem 2. Assume C computes $CLIQUE_k$. By Lemma 6 if $cs^2(k/n)^{w+1} < 1$ then ϕ must accept at least one positive instance. Then ϕ cannot be zero, so by Claim 8 it must accept more than a $1-\binom{w}{2}/(k-1)$-fraction of negative instances. But by Lemma 7 it cannot accept more than a $cs^2(\binom{w}{2}/(k-1))^p$ fraction of such instances. So if $\binom{w}{2}/(k-1) < \frac{1}{2}$ and $cs^2(\binom{w}{2}/(k-1))^p < \frac{1}{2}$ then C cannot compute $CLIQUE_k$.

It remains to set the parameters so all inequalities are satisfied. First we choose $w = \lfloor\sqrt{k-1}\rfloor$ so that $\binom{w}{2}/(k-1) < \frac{1}{2}$. Next we set $p = w\log(n/k)$ so that the only condition remaining to satisfy is $cs^2(k/n)^w < 1/2$. For this choice of p, $s \le w^{2w}\log(n/k)^w$, so all constraints are satisfied as long as $c < \frac{1}{2}(n/w^4 k\log(n/k)^2)^w$. When k is less than $n^{1/4}$, the last expression grows as $n^{\Omega(w)} = n^{\Omega(\sqrt{k})}$.

Proof of Lemma 5. We prove the lemma by induction on w. If $w = 1$ then all p sets are identical so they form a sunflower. For the inductive step, suppose we have more than $s = (p-1)^w w!$ sets of size w each. If at least p are pairwise disjoint, these form a sunflower already. Otherwise, there are at most $p-1$ sets so that any other set intersects at least one of them. These sets span a total of at most $(p-1)w$ elements. So there must be at least one element that belongs to more than $s/(p-1)w = (p-1)^{w-1}(w-1)!$ of the sets. Take all the sets that contain this element and remove it from them. By inductive hypothesis, there must be a sunflower of size p among them. After adding back the removed element we obtain the desired sunflower. □

3 NOT gates are precious

We just showed that monotone circuits that detect k-cliques on n-vertex graphs must be of size at least $n^{\Omega(\sqrt{k})}$. If we allow NOT gates in the circuit, is it possible to do better? The best known circuit for $CLIQUE_k$ has size $(n/k)^{\Omega(k)}$. While it is not known in general if smaller circuits are possible, in the next lecture we will argue that when k is on the order of n it would be very unlikely to get circuits of size $2^{o(n)}$.

Could it then be the case that every monotone function that requires large monotone circuits also requires large circuits? One way to try and prove this would be to give a transformation that converts a general circuit that computes a monotone function into a monotone circuit for the same function of a comparable size. Claim 1 tells us that we can get circuits for each of the function's slices. Surely, if the function is monotone, we can combine all of them into a single monotone circuit?

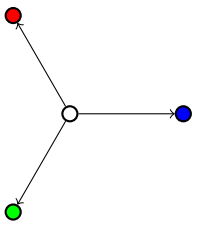
Surprisingly, it turns out that this is not possible: For infinitely many m, there exists a function $f\colon \{0,1\}^m \to \{0,1\}$ that has a circuit of size polynomial in m but requires monotone circuits of size $2^{\Omega(m^{1/8})}$.

The function f is nothing but a modified variant $\overline{\vartheta}_k$ of $CLIQUE_k$ when $k = O(n^{1/4})$. A *vector coloring* of value $v > 1$ is an assignment of unit vectors in $\mathbb{R}^n$ to the vertices of G so that if uw is an edge then the dot product between the vectors assigned to u and w is exactly $-1/(v-1)$. The function $\overline{\vartheta}_k$ is given by:

$$\overline{\vartheta}_k(G) = \begin{cases} 1, & \text{if } G \text{ does not have a vector coloring of value } k-1 \\ 0, & \text{if it does.} \end{cases}$$

This function is monotone: Adding edges to the graph only puts more constraints on dot products, so if G doesn't have a coloring, it cannot obtain one after an edge is added.

While this definition is a bit mysterious, all that we care about here is how the function $\overline{\vartheta}$ behaves on the positive and negative instances of $CLIQUE_k$. First, negative instances do have a coloring of value $k-1$. For example if $k = 4$ the blue, red, and green vertices are assigned three equally spaced vectors on the unit circle. The dot product of every pair is $1/2$.



For general k, we assign each color class to a vector from the corresponding configuration in $k-2$ dimensions in which every pair of vectors has dot product $1/(k-2)$.

Now we argue that positive instances to not have a coloring of value $k-2$. Suppose they did and let $v_1, \ldots, v_k$ be the vectors associated to vertices in the clique. Then

$$\|v_1 + \cdots + v_k\|^2 = \sum_{i=1}^{k} \|v_i\|^2 + \sum_{i \neq j} \langle v_i, v_j \rangle = k - k(k-1) \cdot \frac{1}{k-2} < 0$$

which is impossible.

To summarize, $\overline{\vartheta}_k$ is a monotone function that accepts all positive instances and rejects all negative instances. From our proof of Theorem 2 it follows that $\overline{\vartheta}_k$ cannot have monotone circuits of size $n^{\Omega(\sqrt{k})}$ for $k \leq n^{1/4}$. However, unlike $CLIQUE_k$, $\overline{\vartheta}_k$ has a circuit of size polynomial in n and k. The reason is that the existence of the desired vector configuration can be formulated as a semidefinite program, and semidefinite programs admit algorithms that run in time polynomial in the program size. In the next lecture we will show that every algorithm can be converted into circuits of size polynomial in the input size and in the worst-case running time of the algorithm on inputs of that size.

References

Theorem 2 was proved by Razborov and quantitatively strengthened (in the form presented here) by Alon and Boppana. Lemma 5 was discovered by Erdős, Ko, and Rado. Claim 1 is due to Berkowitz. The example in Section 3 was given by Eva Tardos. The function $\overline{\vartheta}_k$ is essentially the Lovász theta function of the complement graph.