6.890: Algorithmic Lower Bounds: Fun With Hardness Proofs Fall 2014

Lecture 14 — October 21, 2014

Prof. Erik Demaine

1 Overview

This lecture is about strong lower bounds on running time that are possible by assuming the Exponential Time Hypothesis: there is no $2^{o(n)}$ algorithm to solve n-variable 3SAT. This conjecture is a stronger form of $P \neq NP$, and it lets us keep track of "problem size blow-up, something we've ignored before in our NP-hardness reductions. Knowing the blow up of a reduction will allow us to use the ETH to imply lower bounds on problems. Many of the reductions from 3SAT we've seen, such as 3-coloring, vertex cover, dominating set, Hamiltonicity, and Independent Set/Clique have linear blow-up, which we will show implies a $2^{o(n)}$ lower bound.

In parameterized complexity, we get a particularly strong form of $FPT \neq W[1]$: there is no $f(k)n^{o(k)}$ algorithm for Clique/Independent Set, for any computable function f. In particular, there's no $f(k)n^{O(1)}$ (FPT) algorithm. By keeping track of the parameter blowup in our parameterized reductions, we can prove similar bounds for other problems, e.g. multicolored Clique/Independent Set, Dominating Set, Set Cover, and Partial Vertex Cover.

Finally, we'll cover a useful base problem, called Grid Tiling, for proving W[1]-hardness of problems on planar graphs or in 2D.

2 Exponential Time Hypothesis

The *exponential time hypothesis* (ETH) is an assumption made in computational complexity introduced by Impagliazzo and Paturi [**?**]. It states that 3SAT has no $2^{o(n)}$ time algorithm. The best algorithm to date is $O(1.31^n)$ [**?**].

ETH is motivated by the fact that we have not found such an algorithm for 3SAT. The usual assumption is that n is the number of variables, m is the number of clauses.

Note that there is some ambiguity as to what n is, since it could be the number of variables, or it could be the number of clauses. It turns out these assumptions are equivalent (using the sparsification lemma) [**?**]. The number of clauses is $m = O(n^3)$ and we can take a dense formula and make it into $O(2^{\epsilon n})$ sparse formulas for any ϵ.

2.1 Strong Exponential Time Hypothesis

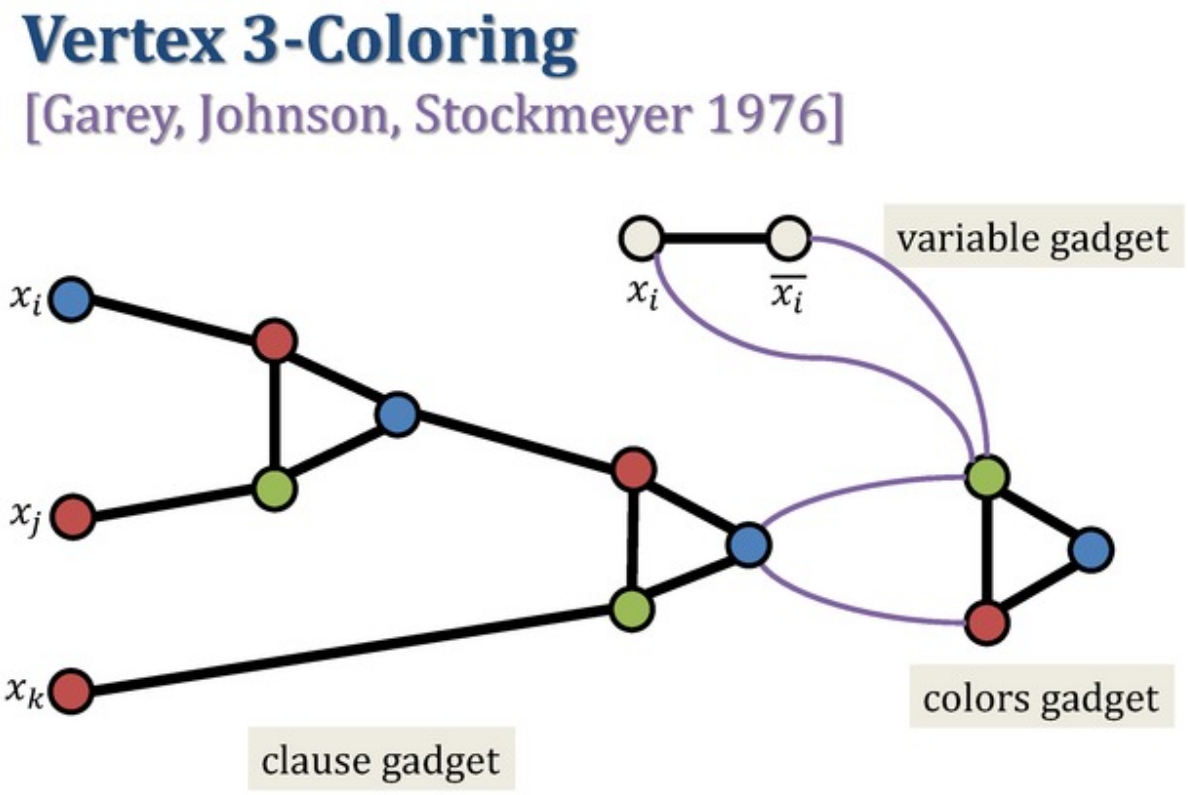
There is a stronger version of ETH that deals with CNF-SAT formulas rather than 3SAT. In particular, the *strong exponential time hypothesis* (strong ETH) says that there exists no $O((2-\epsilon)^n)$-time algorithms for CNF-SAT for any ϵ. In other words, as $k \to \infty$, the constant $(2-\epsilon)$ for k-SAT approaches 2. [**?**].

3 Some problems

ETH is stronger than $P \neq NP$, since it shows exactly what we cannot achieve for each problem. In particular, if we are careful with our reductions, we can show that certain problem cannot achieve algorithms with faster runtimes. Of course, this is assuming ETH.

Vertex 3-coloring

Recall the NP-hardness reduction from 3SAT from Lecture 9 [**?**]. If we have n variables and m clauses in our instance of 3SAT, we create a graph with $O(n+m)$ vertices and edges.



If we could solve vertex 3-coloring in $2^{o(n)}$ time, where $|V|$ and $|E|$ are both $O(n)$, we could solve 3SAT in $2^{o(n)}$ time. The ETH directly contradicts this, and we are forced to conclude that we cannot solve vertex 3-coloring in $2^{o(n)}$ time.

3.1 Size Blowup

Reductions transform an instance x of problem A into in an instance $f(x) = x'$ of problem B. The size blow up $b(n)$ of a reduction is the function relating the size of x to the size of x'. If $|x| = n$, $|x'| = b(n)$.

This means that if we have an algorithm that runs in $T(n)$ time for problem B, then we solve problem A in $T(b(n)) + f$. We are more interested in the hardness reductions, which comes from viewing the contrapositive to that statement: If there is no $2^{o(n)}$ algorithm for A, then there is no $2^{o(b^{-1}(n))}$ algorithm for B, assuming $f \in P$.

In the case that b is linear, then we preserve the fact that there is no $2^{o(n)}$ algorithm.

We can go back and analyze the reductions carefully, we see that vertex cover (from Lecture 7) has linear blowup, which means ETH implies there is no $2^{o(n)}$ time algorithm [**?**]. The reduction from vertex cover to dominating set also has linear blowup, so the same result holds. The same holds for Hamiltonicity (either from Lecture 7 [**?**] or through the max-degree 3 reduction from Lecture 8 [**?**]). Note that planar versions of the problem might require $\Omega(n^2)$ crossover gadgets for the planar case, which blows up the input by at least a factor of $\Omega(n^2)$.

The reduction seen in Lecture 10 from 3-SAT3 to Independent Set also has linear blowup [**?**]. Likewise, assuming ETH, then we get Clique has no $2^{o(|V|)}$-time algorithm.

3.2 Planar Problems

Planar problems usually have quadratic blowup, since some cases require $\Omega(n^2)$ crossovers. This means that ETH only shows there is no $2^{o(\sqrt{n})}$ and no $2^{o(\sqrt{m})}$ where n is the number of variables and m is the number of clauses.

In particular, Planar 3SAT, Planar 3-coloring, Vertex Cover, Dominating Set, Hamiltonicity, and Independent Set have no $2^{o(\sqrt{n})}$ time algorithms for planar graphs with n vertices [**?**, **?**]. This is not the case for Clique.

4 Parametrized Consequences

There are two trivial, but interesting conclusions:

- no $2^{o(k)}n^{O(1)}$ time algorithm for k-Vertex Cover, and k-Path (Longest path), Dominating set, Independent Set, and Clique.
- no $2^{\sqrt{k}}n^{O(1)}$ time algorithms for the planar versions: Planar (no 3-coloring) Vertex Cover, Longest Path, Dominating Set, Independent Set. Also, there are $2^{O(\sqrt{k})}n^{O(1)}$ algorithms known [**?**, **?**]

4.1 Stronger

We can show that ETH implies that there is no $f(k)n^{o(k)}$-time algorithm for Clique, Independent Set for any computable function f [**?**]. We reduce from 3-coloring. Given a graph $G = (V, E)$ with $n = |V|$. We split the vertices into k groups of n/k vertices each. We create a graph with k groups of at most $3^{n/k}$ vertices, one per valid 3-coloring. We make an edge between two vertices if the two colorings they correspond to are compatible. A k-clique corresponds to a 3-coloring.

So if k-Clique is solvable in $f(k)n^{k/s(k)}$ where s is monotone inceasing and unbounded, then set k as large as possible such that $f(k) \leq n$ and $k^{k/s(k)} \leq n$. Let $k = k(n)$ be the minimum of the 2 inverses of the above functions. The running time of 3-coloring on the reduced graph is $f(k)((k3^{n/k})^{k/s(k)}) \leq nk^{k/s(k)}3^{n/s(k)} \leq n^2 3^{n/s(k(n))} \leq 2^{o(n)}$ which contradicts ETH.

4.2 Parametrized Reductions

We say that a function f mapping an instance x of problem A to an instance x' of problem B is *parameter preserving* if $k' * x' \leq g(k(x))$ where g is the parameter blowup. We get that if there is no $f(k)n^{o(k)}$ time algorithm for A, then there is no $f'(k')n^{o(g^{-1}(k'))}$ time algorithm for B. We know that there is no $f(k)n^{o(k)}$ time algorithm for Multicolored Clique/Independent Set, Dominating Set, Set Cover (all these with $k' = k$), and Partial Vertex Cover.

4.3 Grid Tiling

The input to the grid tiling problem is a k by k grid, with each cell $c_{i,j}$ containing a set $S_{i,j}$ of 2D coordinates with values ranging from 1 to n. The goal is to choose one coordinate pair $x_{i,j} \in S_{i,j} \forall i, j$ such that all vertical neighbors have the same first coordinate, and all horizontal neighbors have the same second coordinate. We wish to show this problem is W[1]-hard, which coupled with the ETH, will imply no $f(k)n^{o(k)}$-time algorithm for grid tiling.

We can show this by a reduction from k-Clique. Given the vertices $V = [v_1, v_2, ...v_n]$, we construct a k by k grid, with coordinates representing vertices and edges of our original graph. $\forall i, S_{i,i} = (v, v)|v \in V$ (representing vertex v). Similarly, $\forall i \neq j, S_{i,j} = (v, w) \in E|v \neq w$ (representing edge e between v and w). The vertices we choose in our clique are the coordinates selected along the diagonal $S_{i,i}$. We guarantee there is an edge between every pair of vertices because for any $i \neq j$, we are able to choose coordinates in $S_{i,j}$ and $S_{j,j}$ if and only if there is an edge between the vertices selected in $S_{j,j}$ and $S_{i,i}$. [**?**]

4.4 List Coloring

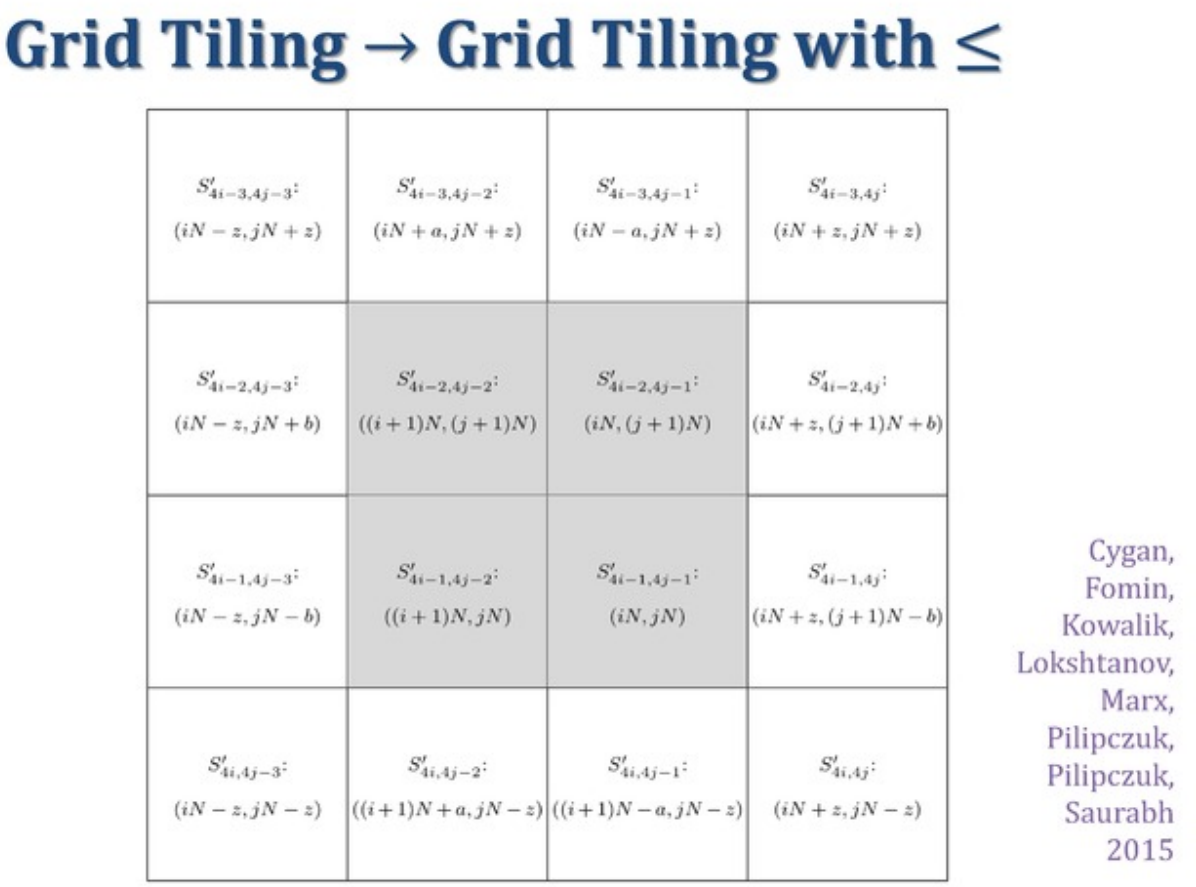
The list coloring problem asks, given a graph and a list L_v of colors for each vertex v, is there a coloring with no monochromatic edges? We can paramterize this problem by *outerplanarity*, which is the number of times all vertices can be removed from the outside face of the graph before it disappears. We will show that this problem is W[1] hard by reduction from Grid Tiling, which with the ETH will imply no $f(k)n^{o(k)}$ algorithm for List Coloring.

The set of possible colors for each list L_v is the set of all possible points (i, j), $0 \leq i, j \leq n$. We create a k by k grid of vertices $v_{i,j}$ for which $L_{v_{i,j}} = S_{i,j}$.

4.5 Grid tiling with $\leq$

Takes the same input as Grid Tiling, but instead requires that the first coordinate of $x_{i,j} \leq$ the first coordinate of $x_{i+1,j}$. Similarly, the second coordinate of $x_{i,j} \leq$ the second coordinate of $x_{i,j+1}$. We can prove this problem is W[1]-hard and imply no $f(k)n^{o(k)}$ algorithm by reduction from Grid

Tiling. We use the following gadget to blow up each tile in our original instance into a a grid of four by four tiles.



[?]

4.6 Scattered Set

Also called d-Independent Set, the Scattered Set problem asks given a graph and numbers k and d, find k vertices with pairwise distances $\geq d$. If we set $d = 2$ this problem is equvilent to Independent Set.

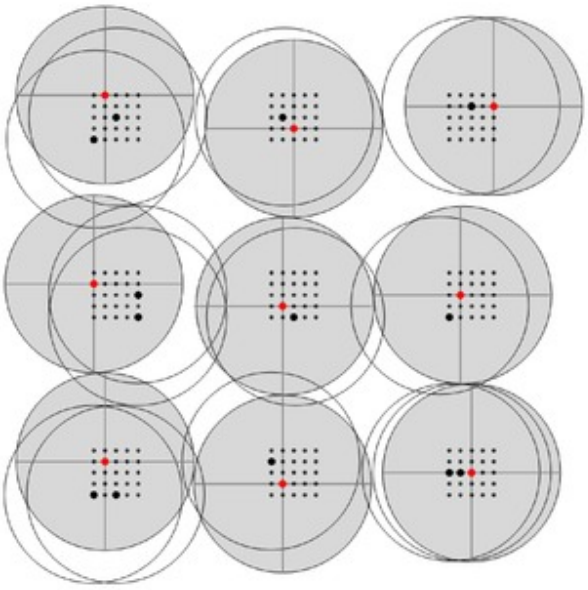
4.7 Unit disk graphs

This problem asks, given a set of points P in 2D, can you draw k unit disks centered on $p \in P$ without the disks intersecting. We can prove this problem is W[1]-hard and imply no $f(k)n^{o(\sqrt{k})}$ algorithm by reduction from Grid Tiling $\leq$. We create arrange k^2 n by n grids of dots in a k by k grid. For each of these n by n grids, only the $S_{i,j}$ points are present. We are forced to choose one point in each n by n grid of dots, and the unit disks around these points will only not intersect if

the coordinates of the points we choose in adjacent n by n grids are monotonically increasing. [**?**]

Grid Tiling with $\leq$
$\rightarrow$ Unit-Disk Independent Set

$S[1,3]$: (1,1) (2,5) (3,3)	$S[2,3]$: (3,2) (2,3)	$S[3,3]$: (5,4) (3,4)
$S[1,2]$: (5,1) (1,4) (5,3)	$S[2,2]$: (3,1) (2,2)	$S[3,2]$: (1,1) (2,3)
$S[1,1]$: (1,1) (3,1) (2,4)	$S[2,1]$: (2,2) (1,4)	$S[3,1]$: (1,3) (2,3) (3,3)



Cygan, Fomin, Kowalik, Lokshtanov, Marx, Pilipczuk, Pilipczuk, Saurabh 2015

[**?**]

6.890 Algorithmic Lower Bounds: Fun with Hardness Proofs

Fall 2014