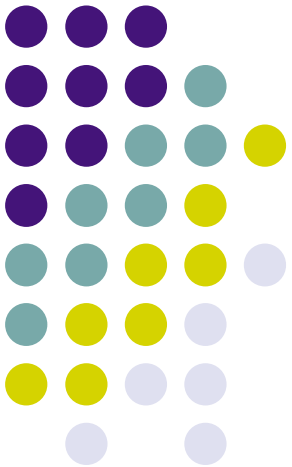
Complexity Classes VI

Expanders and The PCP Theorem

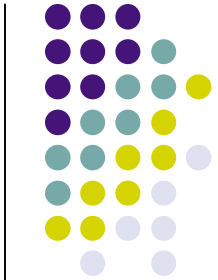
Eric Rachlin



NP = $PCP_{1,\ 1/n}$ [log(n), log(n)],

- PCP theorem: NP = $PCP_{1,\ 1/2}$ [1, log(n)]
- If a PCP verifier's protocol is repeated log(n) times, soundness goes from 1/2 to 1/n.
  - This implies NP = $PCP_{1,\ 1/n}$ [log(n), $\log^2(n)$].
- The gap introducing reduction to CLIQUE given last time relies on O(log(n)) random bits.
- We must simulate log(n) independent runs of of verifier's protocol using a single seed sequence of O(log(n)) bits.
  - We do this next time using expander graphs.
  - Expanders are used in the proof of the PCP theorem, as well as many other areas of theoretical computer science.

Graph Expansion



- Consider a graph G = (V, E).
  - For a vertex v, let T(v) denote the neighbors of v.
  - For any subset S of V, let $T(S) = U_{v \text{ in } S} T(v) - S$.
- A graph G is called an (A, B)-expander if, for all subsets of V where $|S| \leq B$, $|T(S)| \geq A|S|$.
  - Expansion can also be defined in terms of outgoing edges.
- Expanders have many useful properties:
  - They often behave like a random graph.
  - They have very high connectivity.
  - Families of d-regular expanders can be produced in PTIME, and in some cases represented by log space formulas.

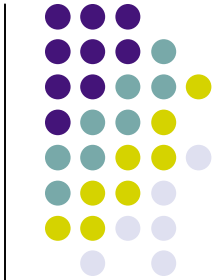
Families of Expanders

- Some well known formulas:
  - Let $V = Z_p$ for any prime p. Let E be all pairs of the form (i, i+1) and (i, i^{-1}), where arithmetic is mod p.
  - Let $V = Z_m \times Z_m$ for any m. Let E be all pairs of the form $(x, x \pm y)$ and $(x, x \pm y)$, again mod m.
- Large expanders can be constructed from smaller expanders using graph products.
- A very simple expander construction appeared recently in SODA 2007.

Expanders and Random Walks

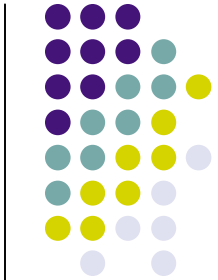
- Results for expanders are often proven using "spectral analysis".
  - Involves eigenvalues of graph's adjacency matrix.
- In an expander, the adjacency matrix has a gap between its largest eigenvalue, 1, and the second largest.
- When the matrix is raised to a power, the gap implies fast convergence to the first eigenvector, the uniform distribution.

Expanders and PCP



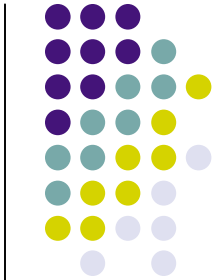
- The verifier for any L in $PCP_{1, 1/2}$ [1, log(n)] uses a sequence of O(log(n)) random bits.
- For any input not in L, at least half of these sequences yield the correct answer.
- Let G be a constant degree PTIME constructible (A, B)-expander with one (or more) vertex for each random sequence.
- A verifier can choose O(log(n)) random sequences using O(log(n)) random bits as follows:
  - First choose a random start vertex using O(log(n)) bits
  - Next take a random walk, using O(1) bits per step.

Soundness of 1/N



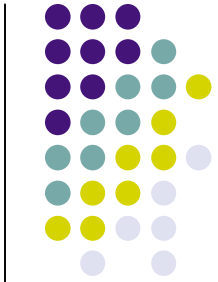
- L is in $PCP_{1,\ 1/2}$ [1, log(n)], so a constant fraction of V represent "good" random sequences.
  - Soundness of 1/2 is arbitrary. We can assume at most B vertices in G are bad, where B/|V| is any constant.
  - Let S_G and S_B be the good and bad vertices respectively.
- Let $T_0 = T(S_B)$, $T_1 = T(T_0) - S_G$ and $T_i(v) = T(T_{i-1}) - T_{i-2}$
  - T_0 are the good vertices that border a bad vertices.
  - T_i is the set of bad vertices i steps from a good vertex.
  - Since G is an expander, T_L is empty for some L = O(log(n)).
  - G is an expander so $|T_{i-1}| = |T(T_L \cup T_{L-1} \cup \ldots \cup T_i)| \geq A|T_i|$.

Reaching Good Vertices



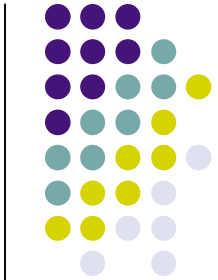
- Let P_{ij}, the probability of transitioning from a randomly selected vertex in T_i to a vertex in T_j.
- Since $|T_{i-1}| \geq A|T_i|$, $P_{i,i-1} \geq o(AP_{i,i}) \geq o(A^2P_{i,i+1})$
- Intuition: Consider a 1D random walk starting at 0
  - Model the walk as a sum of random variables x_i
  - If $P(x_i = -1) = p$, $p(x_i = 0) = cp$, $P(x_i = 1) = c^2p$.
    - $E[x_0 + \ldots + x_n] = n(c^2 - 1)p$
    - $Var[x_0 + \ldots + x_n] = n(c^2 - 1)p$
    - If $n = O(\log(n))$, $x_0 + \ldots + x_n \geq \log(n)$ with high probability.
- A rigorous proof can be given using spectral analysis (see appendix after conclusion).

Another Use of Expanders



- Consider a set of constraints with at most q variables per constraint (example: 3SAT).
- We want a gap preserving reduction such that:
  - The new constraints still have at most q variables.
  - Each variable appears in at most k constraints.
- If k is sufficiently large, we can use expanders.
- If there are n > k copies of x_i appearing in the constraints, replace each with a new variable.
  - Expanders can ensure consistency among the n variables.
  - Treat each as the node of an n vertex expander, add an equality constraint for each edge of the expander.

Proving the PCP Theorem

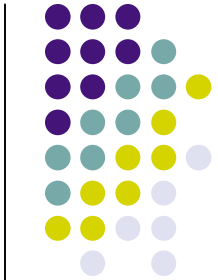


- The PCP Theorem is proven by reduction.
- We begin with a generic NP-complete problem:
  - Constraint Satisfaction (CS)
    - n variables ranging over {1, …, m} for some constant m.
    - C constraints, each a Boolean function over q = o(1) variables
- Reduce CS to an instance of the same problem:
  - If the original instance is satisfiable the new one is as well.
  - If k/C = o(1) fraction of constraints cannot be satisfied in the original instance c(k/C) cannot be satisfied in the new one (for some c > 1).
- We can then repeat the reduction O(log(n)) times.

Structure of the Proof

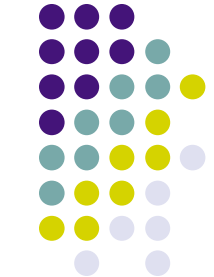
- The reduction is done in three steps:
- 1a) Give a gap preserving reduction such that the number of variables per constraints becomes 2
  - The new problem can be viewed as a graph with variables as vertices and constraints as edges.
- 1b) Give a gap preserving reduction so the constraint graph becomes a d-regular expander.
- 2) Increase fraction of unsatisfiable constraints by c
  - This involves a graph powering operation that increases the domain of each variable.
- 3) Reduce the variables' domains back down to m.

Step 1a



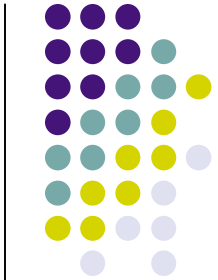
- We can assume m = 2
  - Initially the problem can be reduced to SAT
  - Step 3 can be used to reduce m to 2 anyway.
- We introduce a variable y_i for each constraint.
- $Y_i = \{1, \ldots 2^q\}$ represents the constraint i's input.
  - For each x_j in the constraint, we add a constraint $f(y_i, x_j)$, requiring that y_i be a valid input and x_j be consistent with it.
- The reduction is gap preserving:
  - The new constraints can be satisfied if the old ones can.
  - If an assignment of values to x_i leaves the old i^{th} constraint unsatisfied, one of q constraints involving y_i is unsatisfied.

Step 1b



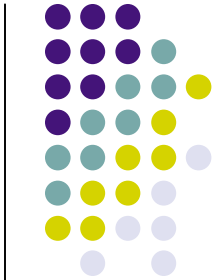
- If all constraints involve 2 variables, CS can be represented as a constraint graph G.
  - vertices are variables, edges are constraints.
  - It is OK if some nodes have multiple edges between them.
- Any constant degree graph can be converted to a constant degree expander by overlaying an expander of the appropriate size.
  - The new edges can be given empty constraints.
- We just need a gap preserving reduction where G becomes a constant degree graph.

Step 1b (cont'd)



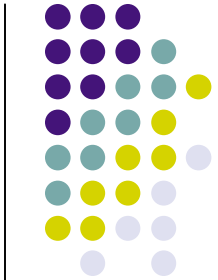
- One at a time, replace each degree n > d vertex with an n vertex constant degree expander.
  - Place equality constraints on expander's edges.
  - Connect each vertex in the expander to a different neighbor of the original vertex.
- Any vertices with too few outgoing edges can be filled in with empty self loops.
- The reduction is gap-preserving:
  - The new constraints can be satisfied if the old ones can.
  - Within each expander, if k vertices "cheat", O(k) equality constraints will be violated. An inconsistent assignment of values to a variable cannot help that much.

Step 2



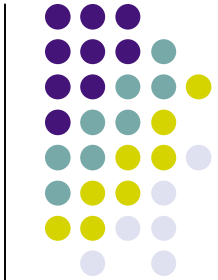
- We are given a constraint graph G that is a d-regular (A, B)-expander. We produce G'' as follows:
  - Consider all $t' = t + t^{1/2}$ step random walks starting at vertex v_i. Let T_i be all vertices reached by the walks. $A^{t'} \leq |T_i| \leq d^{t'}$.
  - The variable associated with v'_i now ranges from $\{1,\ldots,d^{t'}\}$, each value represents a tuple in $\{1, \ldots, m\}^{|T_i|}$. The tuple "asserts" the value of each vertex in T_i.
  - Each edge constraint in G'' corresponds to a $2t + 2$ step path in G'.
    - Let v_i and v_j be the endpoints of the path. The constraint is false if the variables associated with v'_i and v'_j assert values that fail to satisfy any of walk's edges in both $|T_i|$ and $|T_j|$.

Step 2 (cont'd)



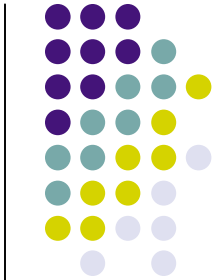
- Since t will be chosen to be some sufficiently large constant, the reduction of G' to G is linear in |V|.
- If G is satisfiable, the satisfying assignment be easily translated to one satisfying G'.
- Any assignment of variables in G that fails to satisfy k constraints "translates" to an assignment failing to satisfy many constraints.
- To show our reduction is gap amplifying, we need to consider arbitrary assignments of values for G'.
  - Does a good assignment in G' imply one for G?

Step 2 (cont'd)



- Given an assignment of variables in G', we can generate one for G by taking "plurality".
  - Starting at v_i in G, consider all t-step random walks, r.
  - For each walks endpoint v_r, let x_r be the value assigned to v_i by v'_r.
  - Assign to v_i the value it is most popular value of x_r.
- If the resulting assignment fails to satisfy k edges:
  - One can show that any random walk in G with edge (v_i, v_j) near the middle is likely to have endpoints which, in G', agree with the values assigned to v_i and v_j.
  - This implies that many vertices in G' agree with the assignments given to vertices in G.
  - The result is that $o(t^{1/2}k)$ constraints in G' are unsatisfied.

Step 3

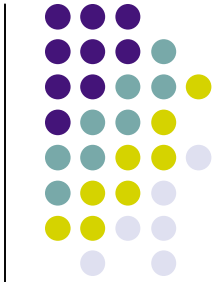


- We rely on the following reduction:
  - Given: A two variable constraint f(x, y) where x and y range over the domain {1, …, m}:
  - Output: Two disjoint sets of m binary variables (one representing x, one representing y) and a third set of poly(m) binary variables such that:
    - Constraints between the new variables are constant-sized.
    - If f(x, y) is satisfied by the binary encoded representation of x and y, all constraints are satisfied.
    - If f(x, y) is not satisfied by the representations of x and y, a (large) constant fraction of the constraints are unsatisfied.

Step 3 (cont'd)

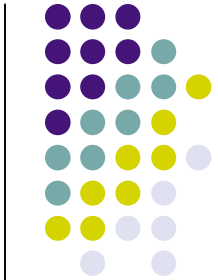
- Next week we will look closely at step 3, for now look at how a constraint of one variable, f(x), can be reduced.
- Represent x using the Walsh-Hadamard Code.
  - Let x_b be the binary representation of x.
  - There are log(x) bits in x_b, and $2^{\log(x)}$ = m linear combinations of these bits (example, $x_{b,1} + x_{b,3}$)
  - Let WH(x) be all m linear combinations.
- If $x \neq y$, WH(x) and WH(y) differ in m/2 locations.
- If WH'(x) differs from WH(x) in at most m/4 locations, any linear combination of x_b can be obtained with high probability.

The Walsh-Hadamard Code



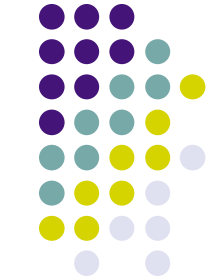
- A linear combination of x_b can be represented as a binary sequence of coefficients.
- Let $WH(x)_s$ denote the value of combination s
  - If WH'(x) is close to WH(x), $WH'(x)_s$ may be in error, but we can instead pick a random linear combination, s' and check $WH(x)_{s'}$ and $WH(x)_{s'+s}$.
  - Note that $WH(x)_s = WH(x)_{s'} + WH(x)_{s'+s}$.
  - If WH'(x) differs from WH(x) in at most m/4 locations, $WH'(x)_{s'} + WH'(x)_{s'+s}$ is correct with probability at least 1/2.
- What if S is not within distance m/4 of a codeword?
  - We'll see that $Prob(WH(x)_s = WH(x)_{s'} + WH(x)_{s'+s}) < 1/4$.

Review of Proof



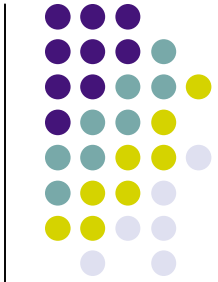
- Given: CS with q binary variables per constraint.
  - Step 1a: gave a gap preserving reduction such that the number of variables per constraints became 2
    - Fraction of unsatisfied constraints was reduced by 1/q.
  - Step 1b: gave a gap preserving reduction so the constraint graph becomes a d-regular expander.
    - Fraction of unsatisfied constraints reduced by a constant.
  - Step 2: created new constraints w/ O(t)-step random walks.
    - Fraction of unsatisfied constraints, if o(n), is amplified by a factor $o(t^{1/2})$.
  - Step 3: gave gap preserving reduction to binary variables.
    - Fraction of unsatisfied constraints reduced by a constant.
- If t is sufficiently large, O(log(n)) introduces o(n) gap!

Conclusion



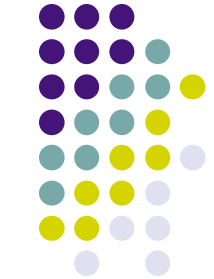
- Expanders allow us to amplify random bits.
- They also allow us to reduce the number of occurrences of a variable in a set of constraints.
- If a constraint graph is a constant degree expander, one can easily amplify the fraction of unsatisfied constraints.
- This, along with alphabet reduction, yields a relatively simple proof of the PCP Theorem.

Expanders and Sampling



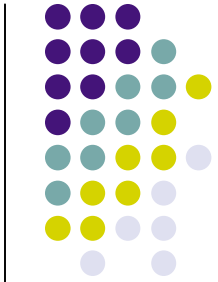
- Let G = (V, E) be an expander and S ⊂ V. Let P be a diagonal matrix where $P_{i,i} = 1$ iff v_i in S.
- For a vector r, let $||r||_1 = \sum |v_i|$, Let $||r||_2 = (\sum v_i^2)^{1/2}$
- Claim: If G is an expander its adjacency matrix, M, has eigenvalues $E_1, \ldots, E_n$ where:
  - $E_1 = 1$, all other $|E_i| \leq c$ for some $c < 1$.
  - 1 - c, is a function of G's expansion, A, and degree d.
    - $(d - c)/2 \leq A \leq [2d(d - c)]^{1/2}$ (Tanner, Alon, and Milman)
- Pick a starting point uniformly at random and take a t-step random walk.
  - The probability of staying in S is $||(PM)^t Pu||_1$
  - Here u is a vector representing the uniform distribution.

Bounding the Exit Probability



- Theorem: If we take a t-step random walk in G, the probability of staying in $S \subset V$, when $|S|/|V|$ is sufficiently small, is at most $(c + |S|/|V|)^t$
- This is shown through the following lemma:
  - For any nonnegative r, $||PMPr||_2 \leq (c + |S|/|V|)||r||_2$
- Proof:
  - $||(PM)^tPu||_1 = ||(PMP)^tu||_1$ Since P = PP
  - $||(PMP)^tu||_1 \leq n^{1/2}||(PMP)\ ^tu||_2$ (Cauchy-Schwarz)
  - $n^{1/2}||(PMP)\ ^tu||_2 \leq n^{1/2}(c + |S|/|V|)^t||u||_2$ (Lemma)
  - $n^{1/2}(c + |S|/|V|)^t||u||_2 = (c + |S|/|V|)^t$

$||PMPr||_2 \leq (c + |S|/|V|)||r||_2$



- Let $Pr = r_u + r_p$, where $r_u \,||\, u$ and $dot(r_u, r_p) = 0$.
- $||PMPr||_2 = ||PM(ku + r_p)||_2$, for $k = dot(Pr, u)/||u||_2^2 = [(1/|V|)\sum (Pr)_i]/(1/|V|) = \sum (Pr)_l \leq (|V|)^{1/2}||Pr||_2$
- $\leq ||PMku||_2 + ||PMr_p||_2$ (triangle inequality)
- $||Pku||_2 + ||PMr_p||_2 \leq ||Pku||_2 + ||Pcr_p||_2$
  - u is an eigenvector with eigenvalue 1, r_p can be written as a sum of eigenvectors with eigenvalues between -c and c.
- So $||PMPr||_2 \leq k||Pu||_2 + c||Pr_p||_2 \leq k||Pu||_2 + c||r_p||_2$
  - Since P is a projection matrix, zeroing elements not in S.
- Finally $k||Pu||_2 = k\,(|S|)^{1/2}/|V| \leq (|S|/|V|)||Pr||_2$
- So we have $(|S|/|V|)||Pr||_2 + c||Pr||_2 \leq (c + |S|/|V|)||r||_2$