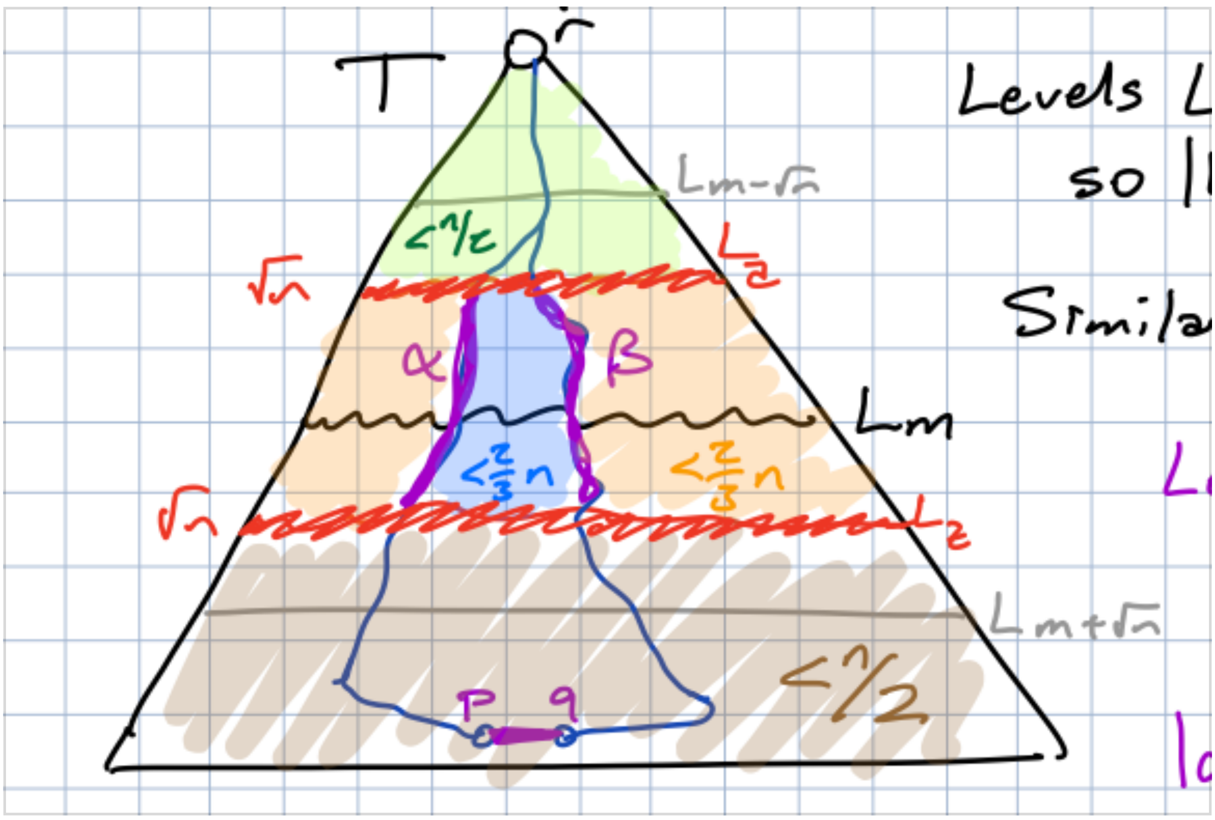
CS 7301.003.20F Lecture 8—September 14, 2020

Main topics are #planar_graph_separators.

Planar Graph Separators

- Let G be a simple planar graph.
- Many algorithms in planar graphs rely on a recursive decomposition of the graph where you partition the vertices into two subsets of approximately equal size and recursively partition and solve the problem on the smaller subsets.
- A *balanced separator* of graph G is a subset of vertices S such that G \ S is disconnected and each component has ≤ 2/3 n vertices.
- The divide-and-conquer paradigm works particularly well with these separators.
- But in order for this strategy to work well, you can't have too much connection between the two subsets of vertices.
- Theorem [Lipton, Tarjan '79]: Any planar graph G has a balanced separator of size O(sqrt{n}).
- To construct the small balanced separator, we're going to find a couple separators that are not necessarily so small, and then combine them in a clever way.
- First, let's triangulate G. That only makes separation harder, but provides more structure as well.
- Let T be a breadth-first search tree rooted at an arbitrary vertex r of G, and let C = (G \ T)^* be the complementary dual spanning tree for T.
- First we have *level separators*:
  - Let L_i denote vertices at distance i from r
  - Let M be such that there are at most n / 2 vertices at distance < M and at most n / 2 vertices at distance > M
  - Edges in G only go between adjacent levels of the BFS tree; otherwise some vertices would be closer to r than we think
  - Therefore, L_M is a balanced separator. But it may also be large!
- And then we have *fundamental cycle separators*:
  - For any edge e not in T, let cycle(T, e) denote the unique cycle in T cup e.
  - I claim there is some e not in T such that cycle(T, e) is a balanced separator.
  - Because G is a triangulation, Euler's formula implies we merely need to separate the faces into multiple components of size at most 2f / 3.
    - m = 3f / 2, which implies n = f / 2 + 2
    - If we have a component with at most 2f / 3 faces, they are each incident to at most f / 3 + 2 < 2n / 3 + 1 vertices. But at least one of those vertices is part of the separator.

    - So now, imagine rooting C at any dual vertex leaf incident to r. Direct the edges away from the root.
    - Now start following a path away from the root. When given a choice, take the edge with the larger subtree. Every vertex of C has degree at most 3, so there are at most two choices each step.
    - Let u be the first node whose subtree has at most 2f / 3 dual vertices, and let v be its parent.
    - v's subtree had at least 2f / 3 + 1 dual vertices, meaning u's subtree has at least f / 3 dual vertices
    - So removing edge uv from C leaves two components with at most 2f / 3 dual vertices each.
    - cycle(T, (uv)^*) separates this same pair of face subsets
- If either of these separators has size ≤ 6sqrt(n), we're already done. So suppose not.
- Levels M - sqrt{n} through M - 1contain at most n vertices, so there is some L_a such that |L_a| ≤ sqrt{n} and m - sqrt{n} ≤ a < m
- Similarly, there is some L_z such that |L_z| ≤ sqrt{n} and m < z ≤ m + \sqrt{n}
- Let pq = (uv)^* be the primal edge on the cycle separator
- Let alpha be the segment of the shortest path from r to p strictly between L_a and L_z. Let beta be the segment of the shortest path from r to q strictly between L_a and L_z.
- Both alpha and beta have at most 2sqrt{n} vertices.
- Now consider L_a cup L_z cup alpha cup beta. It has size at most 6sqrt{n}.
- There are at most n / 2 vertices in any components above L_a and at most n / 2 vertices in any components below L_z.
- Similarly, components inside or outside the (truncated) fundamental cycle have at most 2n / 3 vertices. We found a small balanced separator!



- This construction can even be done in O(n) time!

Application: Independent Set

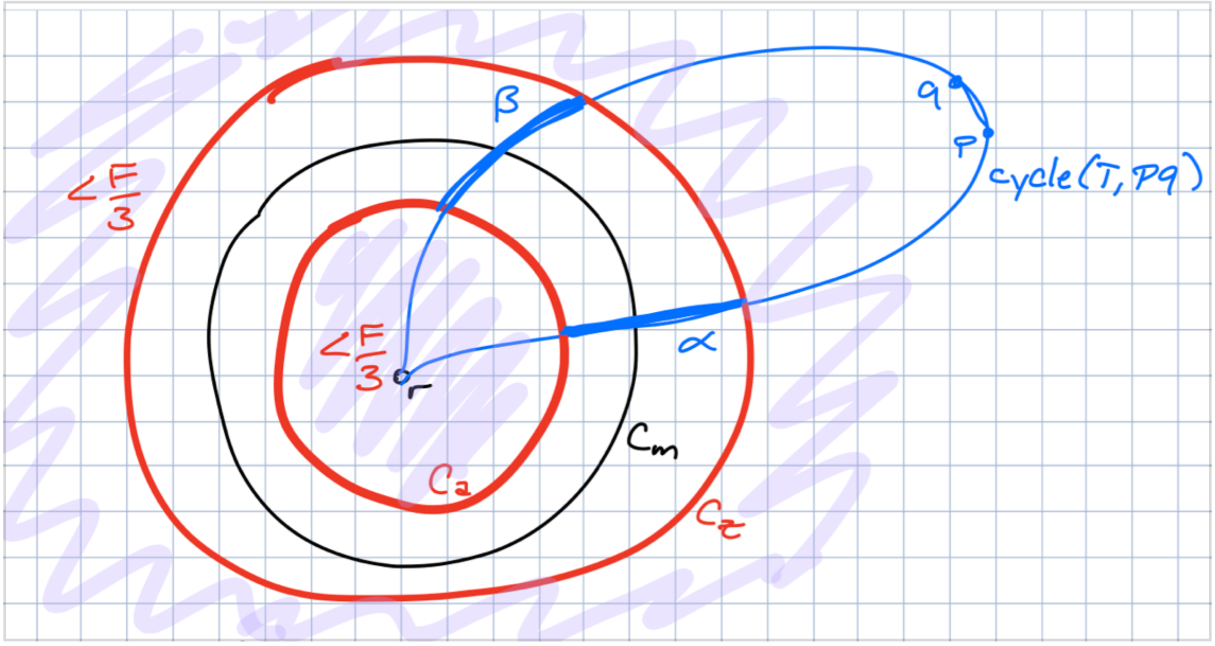
- Given a graph G, an independent set S′ is a subset of the vertices such that no edge has two endpoints in S′.
- In the maximum independent set problem, we want to find a maximum cardinality independent set.
- This is one of the more infamous NP-hard problems. The best algorithms have a running time very close to 2^n. We're practically forced to just try nearly all the subsets of vertices and see which ones are independent. There are no good polynomial time approximation algorithms, either.
- The problem is still NP-hard in planar graphs, but we can do a lot better anyway using divide-and-conquer.
- Suppose G is planar. Find a balanced separator S of size O(sqrt{n}).
- Now, for each independent subset S_0 of S
    - For each component G_j of G \ S
        - Let G′_j be the induced subgraph of vertices in G_j not adjacent to S_0. Recursively compute a maximum independent set S_j in G′_j.
    - Let S′ be the union of S_0 and these recursively computed independent sets.
- Return the best S′ over all those choices of S_0.
- There are no edges between components G_j and we deleted vertices adjacent to our choices of S_0, so we're safe to do those independent recursive calls.

- We get a (very loose) running time recurrence T(n) ≤ 2^{6sqrt{n}} * n * T(2 n / 3).
- There's a couple ways to solve this. We could figure out lg T(n) ≤ 6 sqrt{n} + lg n + lg T(2n / 3) = lg T(2n / 3) + O(sqrt{n}). Using recursion trees or whatever, we see lg T(n) = O(sqrt{n}), so T(n) = 2^{O(sqrt{n})} <<< 2^n.
- We can use a similar algorithm to find an independent set of size k in 2^{O(sqrt{k}) poly n time if one exists.

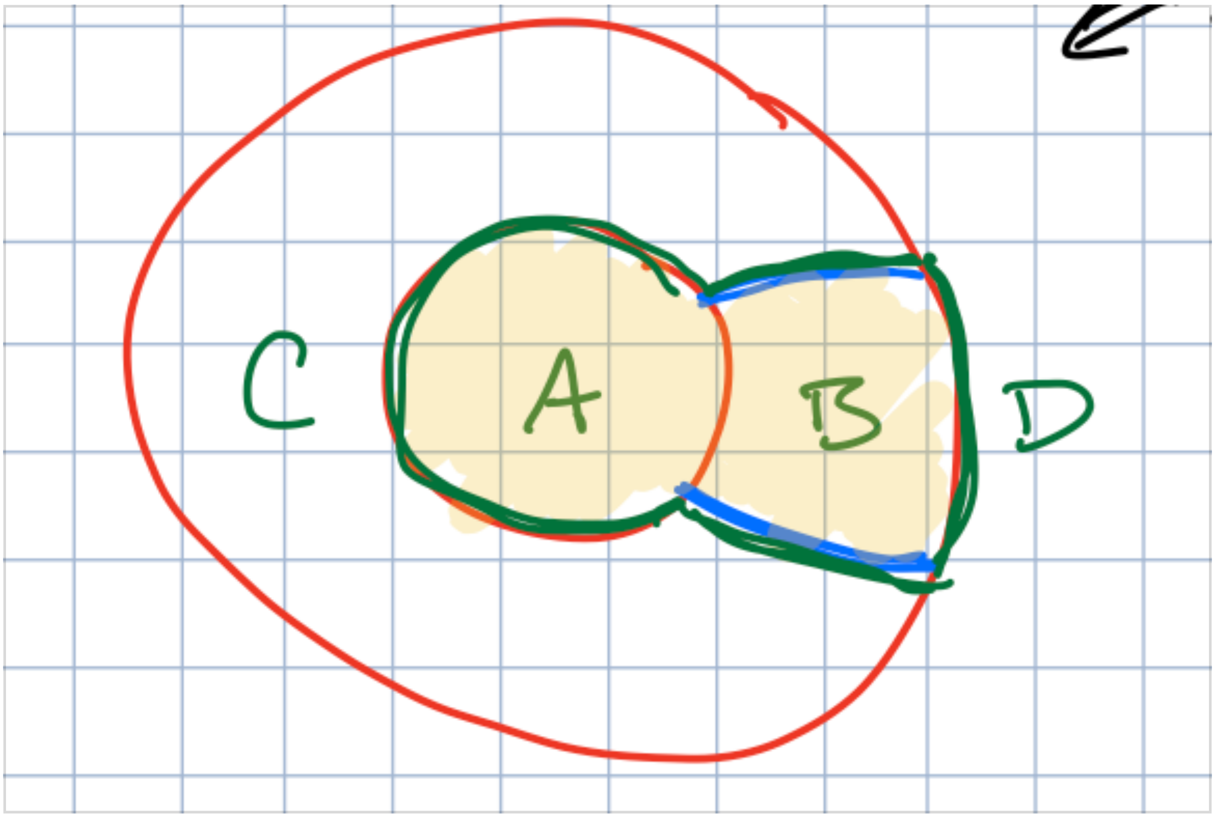
Cycle Separators

- Now suppose we want to separate the (triangular) faces, but we want to use a simple cycle to do so. We need a cycle of only O(sqrt{n}) vertices [Miller '86]. I'll give a proof by Nayerri and Har-Peled ['17].
- As before, we start with a BFS tree T' and let cycle(T', pq) denote the balanced fundamental cycle separator. If it's not too big, we're already done. Otherwise…
- Let r be the lowest common ancestor of p and q, and let T be the BFS tree rooted at r st. cycle(T, pq) = cycle(T', pq).
- Let p be farther from r than q and let h:= dist(r, p) ≥ dist(r, q).
- We'll embed G so p is on the outer face of G (imagine T growing "outward" for the first h levels).
- Define the *level* of a face u to be the maximum distance from r to vertices of f.
- For i < h, let R_{≤i} denote the faces with level ≤ i, and let C_i be the outer boundary of R_{≤i}.
  - C_i includes only vertices distance i from r
    - Any vertex x of smaller distance to r lies within R_i, so x is internal to R_{≤i}
  - C_i is simple
    - If not, then it uses the same vertex x twice, and x is a cut vertex. But then one side of the cut has vertices of distance > i
  - Any two C_i, C_j are disjoint unless i = j
  - Each cycle C_i intersects cycle(T, pq) exactly twice.
- Now, consider the following sets of cycles:
  - C_1, C_{1+sqrt{n}}, C_{1+2sqrt{n}}, …
  - C_2, C_{2+sqrt{n}), C_{2+2sqrt{n}}, …
  - …
  - C_{i0}, C_{i0 + sqrt{n}), C_{i_0 + 2sqrt{n}}, … for all i_0 ≤ sqrt{n}
- These sets are disjoint, there are sqrt{n} of them, and they contain n vertices total.
- Therefore, at least one set contains at most n / sqrt{n} = sqrt{n} vertices total. Call this set the *small ladder*.
- If there is a C_i in the small ladder such that C_i encloses between f / 3 and 2f/ 3 faces, then C_i itself is the small balanced separator we want.

- Otherwise, there are two consecutive cycles in the small ladder C_a and C_z = C_{a + sqrt{n}} such that C_a encloses fewer than f / 3 faces but C_z encloses more than 2f / 3 faces.



- These C_a, C_z, and cycle(T, pq) partition the faces into four sets as follows:



- Each of these regions contain < 2f / 3 faces.
- If any one region contains ≥ f/3 faces, we take the boundary of that region as our cycle separator.
- Otherwise, f / 3 < |A cup B| < 2f / 3 so we take the boundary of A cup B.
- Like before, we can construct the cycle separator in O(n) time.
- And we can extend the result to find a cycle separator that balances arbitrarily weighted

vertices, edges, and faces, as long as no single thing has too much weight.

r-divisions

- Sometimes, it helps to recursively compute a bunch of separators before doing anything else.
- By recursively finding cycle separators for the edges, we can, for any r, partition the edges into O(n / r) pieces, each with r edges, and at most O(sqrt{r}) vertices that sit on the boundary incident to edges in other pieces.
- That means there's a total of O(n / sqrt{r}) boundary vertices across all pieces.
- A *hole* of a piece is a face of the piece that doesn't exist in the original graph.
- By alternating between partitioning the edges and the hole faces, we can find an r-division with only O(1) holes per piece. I'll show an application next week.
- Naively, all this recursive partitioning would take O(n) time across all separations in each recursion level, for O(n log (n / r)) time total.
- But by being more careful, we can compute it in O(n) time [Klein, Mozes, Somner '13].