COS 423 Lecture 20
Nonbipartite Matching

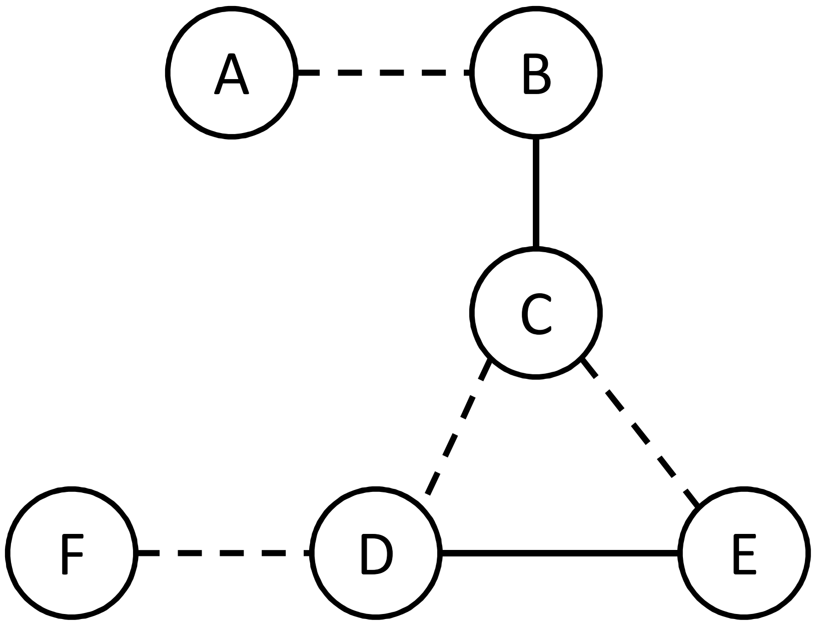
Maximum-size matching in nonbipartite graphs: how to find augmenting paths?

Paths must alternate unmatched, matched edges, but parity of a vertex (odd or even distance from a free vertex) is ambiguous.

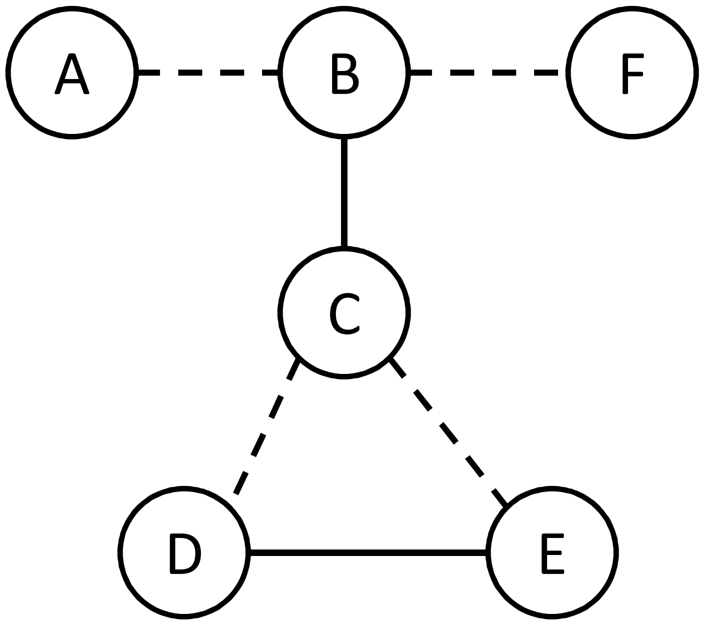
If parity is fixed on initial visit, can miss an augmenting path.

If both parities are allowed, can find a non-simple alternating path between free vertices: not augmenting

One parity, search from A: A even, B odd, C even, D odd, E even, F never visited; A, B C, E, D, F missed

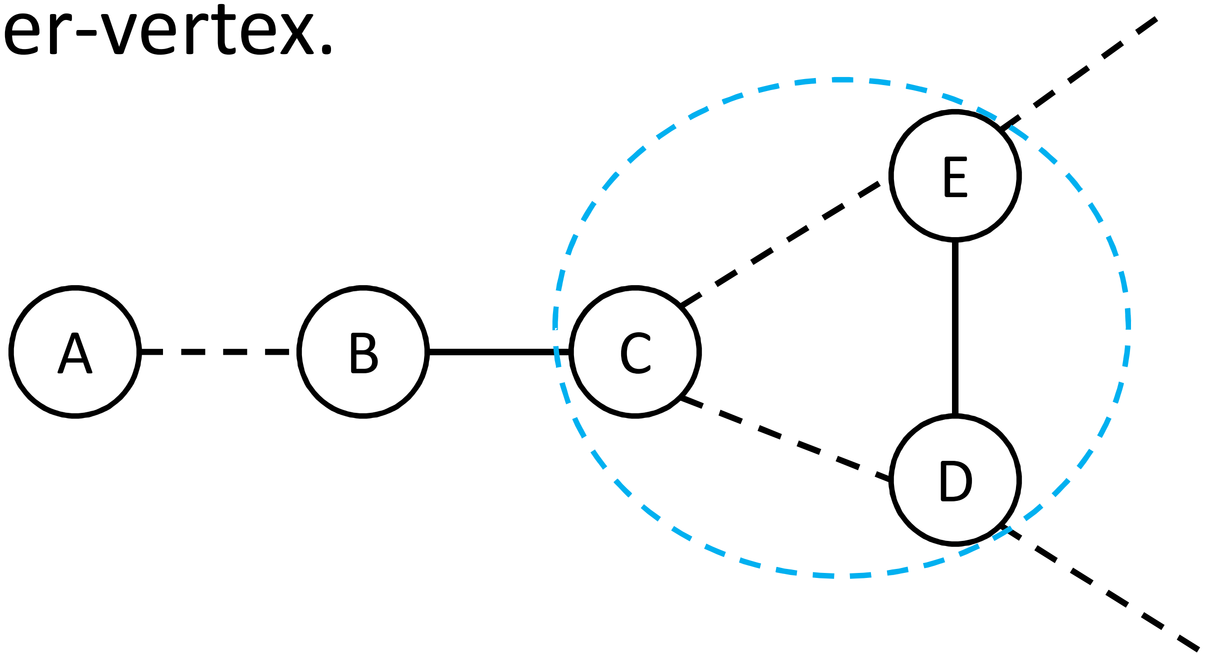


Oboth parities, search from A: A even, B odd, C even, D odd, E even, C odd, B even, F odd: path from A to F alternating but not simple



Edmonds' solution

Blossom-shrinking: Each time an odd alternating cycle is traversed, contract it into a single super-vertex.



A, B, C is *stem*; C, D, E is *blossom*

Edmonds’ algorithm

Start with no matched edges.

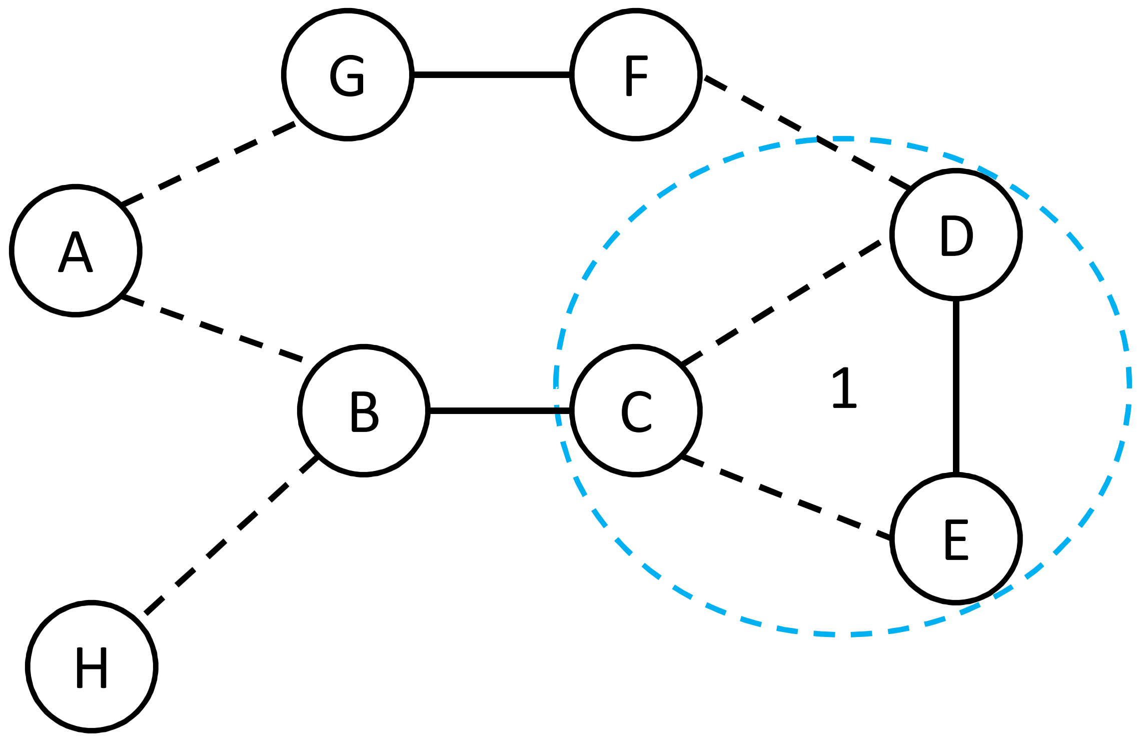
Search from one or more free vertices along alternating paths.

When revisiting a visited vertex, if odd cycle, shrink to a super-vertex, continue search along all unmatched incident edges.

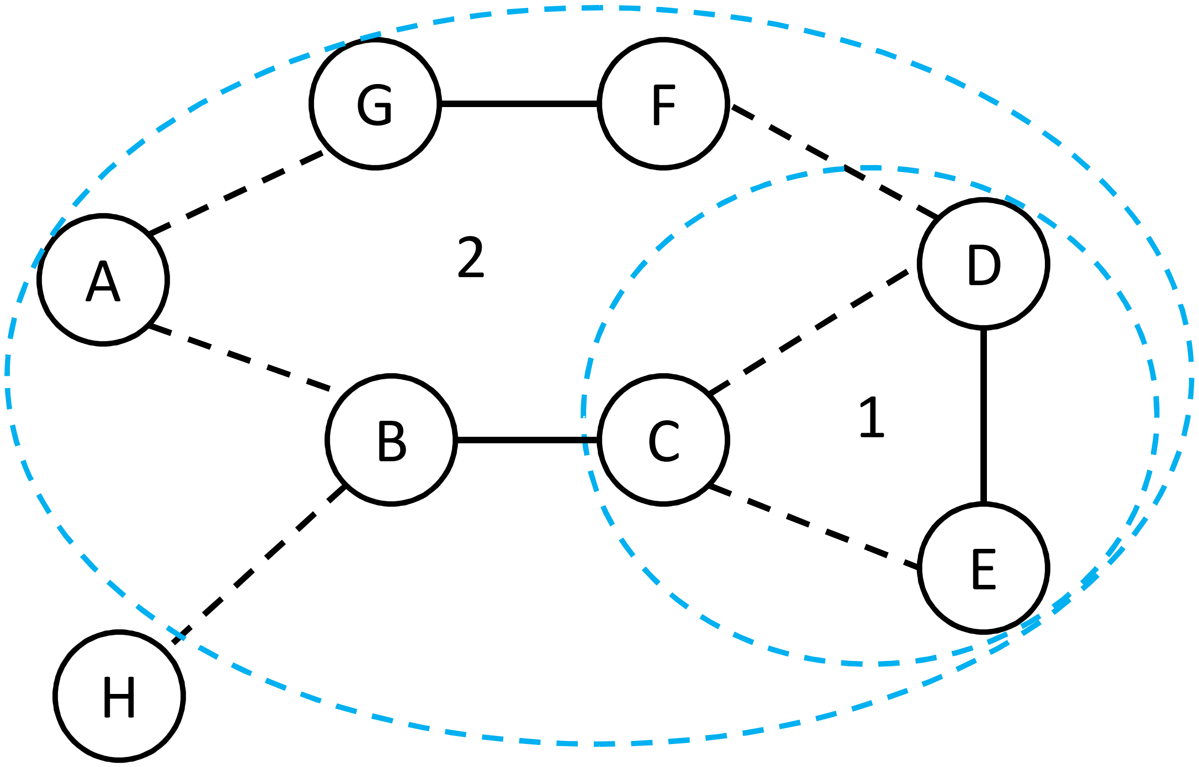
If augmenting path found, expand blossoms on path, augment.

If no augmenting path found, delete all visited vertices.

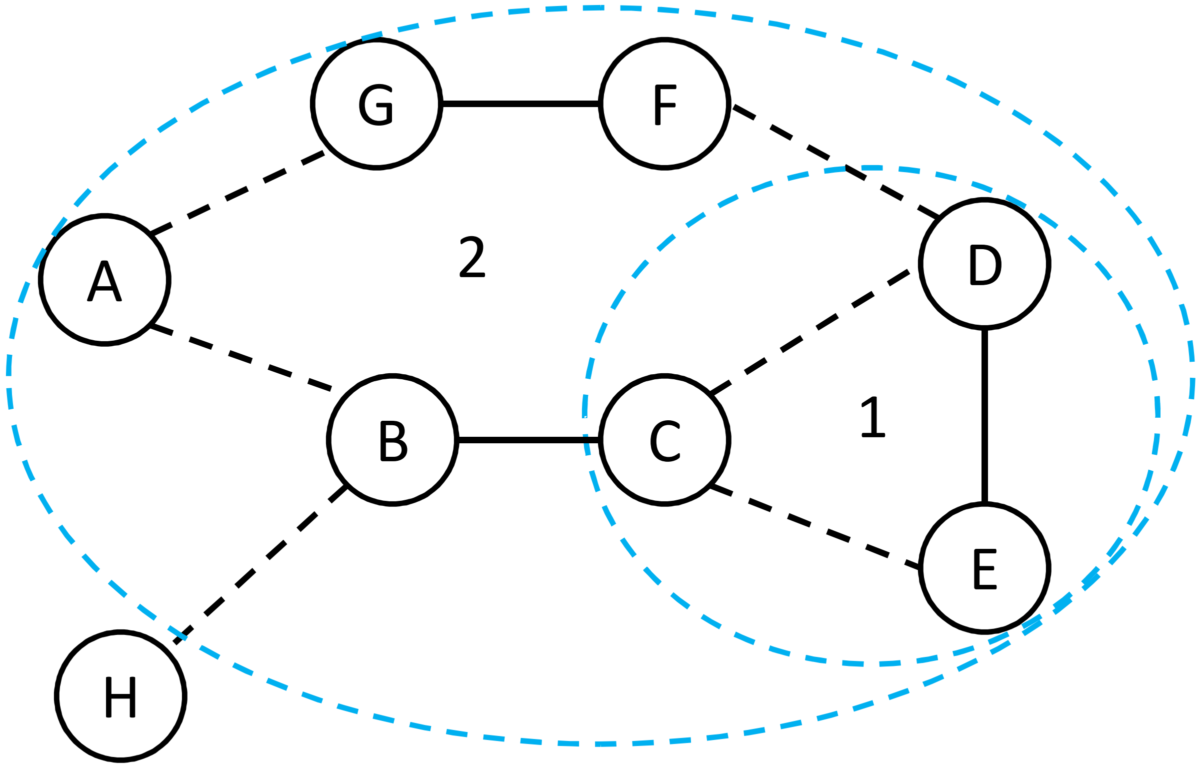
Search from A: A even, B odd, C even, D odd, E even, C odd: blossom 1(even)



Continue search: F odd, G even, A odd: blossom 2 (even)



Continue search: H odd: augmenting path 2, H expands to A, G, F, 1, B, H expands to A, G, F, D, E, C, B, H

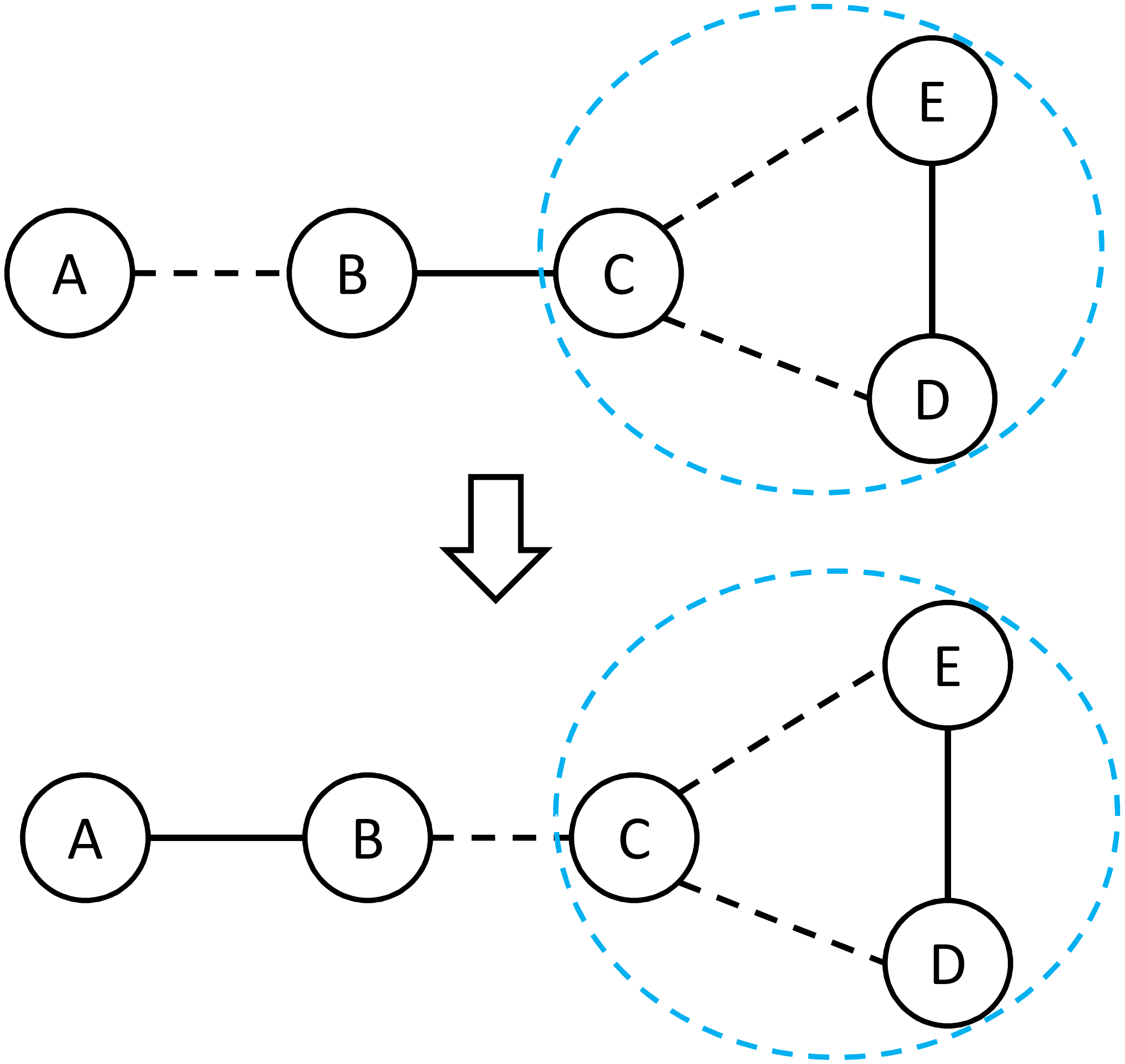


Correctness via two lemmas

Lemma: There is an augmenting path before shrinking a blossom if and only if there is an augmenting path after.

Proof (easy half): If there is an augmenting path after shrinking, there is certainly one before: expand each shrunken blossom on augmenting path and connect two ends. One is the blossom base; to connect, walk around the blossom from the other end, starting with a matched edge in the blossom.

Proof (hard half): A matching has an augmenting path iff it is not of maximum size. Suppose there is an augmenting path in the graph before the blossom-shrinling. Then there is still an augmenting path if the edges along the stem of the blossom are switched from matched to matched and vice-versa (no change in matching size). Now the blossom has no stem. If the aumenting path misses the blossom, it is an augmenting path after the blossom is shrunk.


A
B
C
D
E
A
B
C
D
E

Proof (hard half cont.): If the augmenting path hits the blossom, then the part from the free vertex not the base of the blossom to the blossom is an augmenting path in the shrunken graph: in the shrunken graph, the blossom is a free vertex. Thus the shrunken graph with edges switched along the stem has an augmenting path. Switching the edges along the stem back to their original state does not change the size of the matching in the shrunken graph. Thus the original matching in the shrunken graph has an augmenting path.

Lemma: If a search from a set of free vertices finds no augmenting path, then no visited vertex will be on an augmenting path in the future.

Proof: After blossom-shrinking, the subgraph S induced by the set of visited vertices is bipartite, only odd visited vertices are adjacent to unvisited ones, via unmatched edges, and all the free visited vertices are even. If some future augmenting path contained a visited vertex, it would have to enter via an unmatched edge to a visited odd vertex and leave via an unmatched edge from a visited odd vertex. But then S could not be bipartite.

Implementation

Disjoint set data structure to keep track of blossoms.

Time per search = $O(m\alpha(m, \lceil m/n \rceil)$

Total time = $O(nm\alpha(m, \lceil m/n \rceil)$

Simplest implementation is DFS, one start vertex at a time.

For each vertex, keep track of the blossom containing it, its parity, and whether it is in a blossom (even).

Search proceeds from even vertices via unmatched edges and from odd vertices via matched edges.

Even-to-even edge gives a blossom (on current path)