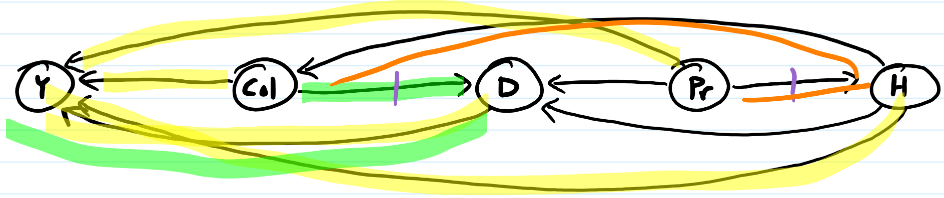
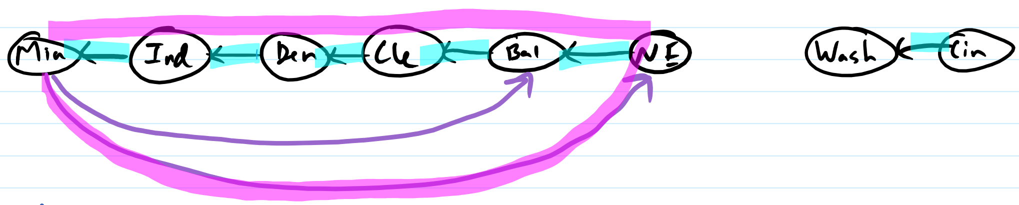
Min Feedback Arc Set





vertices   teams

edges $u \to v$ team u lost to team v

interesting problems : is there a cycle?

easy [if not, find an ordering of vertices so all edges in same direction

hard! [else find ordering that minimizes number of wrong-way edges

brute force : for each ordering $n!$ orderings

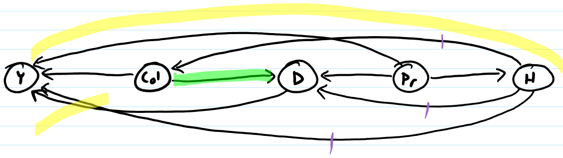
count wrong-way edges, keeping track of min

no one has done significantly better

no one has proven that significantly better is impossible

NP-complete

Graph Representation



O(1) if labels are integers 0, ..., n-1

Adjacency Matrix

to

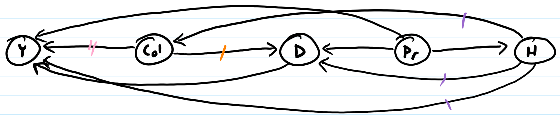
from	Y	Col	D	Pr	H
Y	F	F	F	F	F
Col	T	F	T	F	F
D	T	F	F	F	F
Pr	T	F	T	F	T
H	T	T	T	F	F

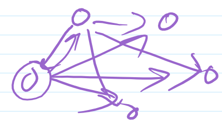
has_edge(H, Col)? O(1) expected

map vertex labels → indices

Y → 0
Col → 1
D → 2
Pr → 3
H → 4

space: $O(n^2)$





Adjacency List — for each vertex, a list of vertices it has edges to

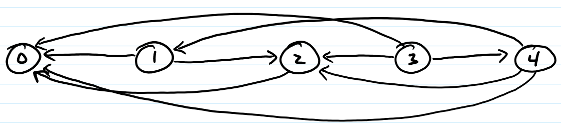
Y :
Col : Y D
D : Y
Pr : Y D H
H : D Y Col

has_edge(u, v): get list for u by indexing into array of adjacency lists
sequential search on that list for v

space: $O(n^2)$ worst case
$O(n + m)$
m = # edges

$O(n)$ worst case
n = # of vertices

$0 \le m \le n(n-1)$ m is $O(n^2)$ worst case
but $O(n)$ for a lot of apps (sparse)



for each vertex u
  [for each edge u → v
   do something

Adj list

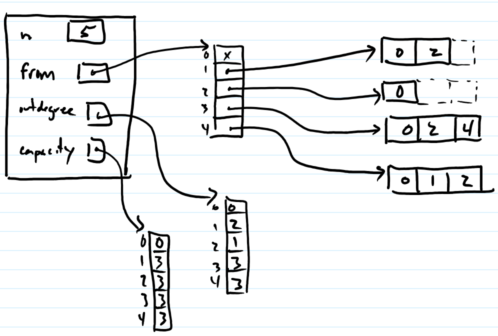
$\sum_{i=1}^{n} 1 + \underline{(n-1)}$ worst case
$= O(n^2)$ worst case

$\sum_{i=1}^{n} (1 + \text{outdegree}(v_i))$
↳ # of edges out
$= \sum_{i=1}^{n} 1 + \sum_{i=1}^{n} \text{outdegree}(v_i)$
$= n + m$
$O(n+m)$ total

Adj matrix

$O(n^2)$

for each entry
  if T
    do something



Graph Implementation Time/Space Complexity

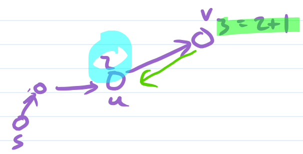
assuming vertices are referred to by integer index

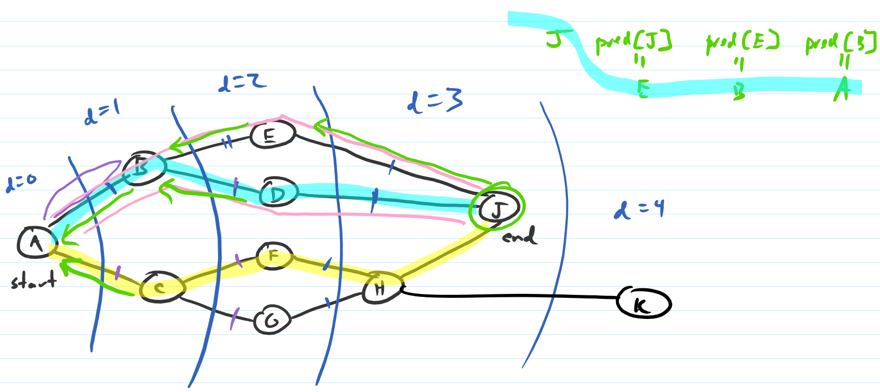
n = # vertices
m = # edges

	Adj Matrix	Adj List
space	$O(n^2)$	$O(n^2)$ worst $O(n+m)$ $0 \le m \le n^2$
has_edge	$O(1)$	$O(n)$ worst case $O(\text{outdegree}(n))$ directed $O(\text{degree}(n))$ undir
add_edge	$O(1)$	$O(n)$ if enforcing one edge/dir/pair (call has_edge first) $O(1)$ if allowing multiple edges between a single pair
for_each_neighbor	$O(n)$	$O(n)$ worst case $O(\text{degree}(n))$ undir $O(\text{outdegree}(n))$ dir
for each vertex v for each neighbor of v	$O(n^2)$	$O(n^2)$ worst case $O(n+m)$

Breadth-First Search

```
BFS(G,s)

for each vertex u in graph_vertices(G)
   color[u] <- WHITE      unseen
   d[u] <- infinity       distance from s (# of edges)
   pred[u] <- NIL

color[s] <- GRAY
d[s] <- 0
pred[s] <- NIL

Q <- [s]

while not queue_is_empty(Q)          for each vertex
   u <- queue_dequeue(Q)
   for each v in graph_adjacent(u)   for each neighbor
      if color[v] = WHITE            if still unseen
         color[v] <- GRAY            see it
         d[v] <- d[u] + 1
         pred[v] <- u                seen, but not explored
         queue_enqueue(Q, v)
   color[u] <- BLACK                 done w/u

                  — done
```





A   B C   E D F G   J H   K

FIFO queue   ~~A~~ ~~B~~ C E D

$\underbrace{B}_{d=1}$ $\underbrace{C\,E\,D}_{d=2}$

d+1

from G to F)

?