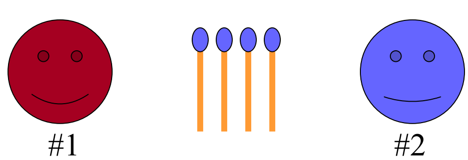
Graphs & Games

EECS 477

Lecture 17, 11/12/2002

Nim: the rules

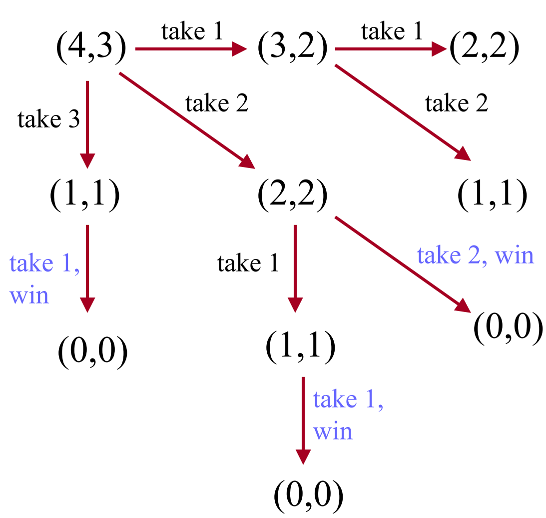
- Two players, heap of N matches
- Player #1 must take A, 0<A<N
- After player #K took A matches, player #(3-K) must take B, 0<B<=2*A
- Player taking the last match wins



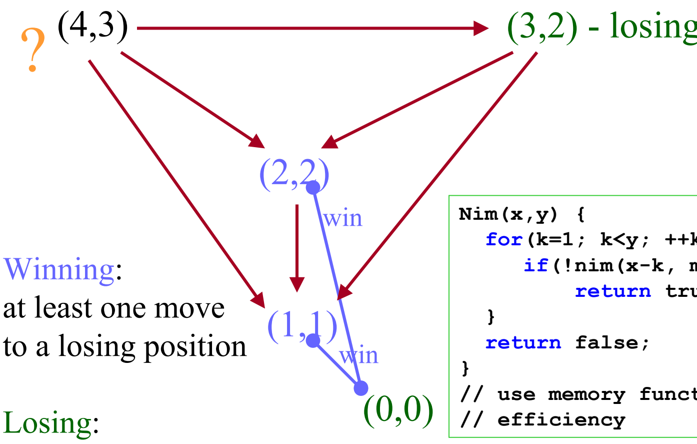
Nim: example

- Initially: 4 matches
- Represent state as a pair
  (num of matches, max number can take)
- Init: (4,3)
  – "four matches, can take at most three"
- Graph
  – Nodes: states
  – Arrows: moves (half-moves)

Nim: tree?



Nim: graph



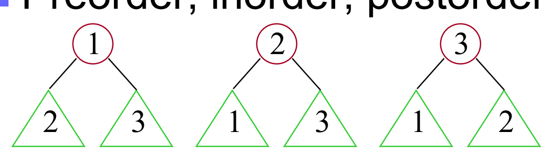
Winning: at least one move to a losing position

Losing: all moves lead to a winning position or no moves

```
Nim(x,y) {
  for(k=1; k<y; ++k) {
    if(!nim(x-k, min(2k,x-k)))
       return true;
  }
  return false;
}
// use memory function for
// efficiency
```

Tree traversals

- Preorder, inorder, postorder



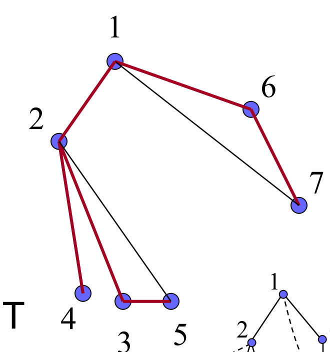
- $T(n) \leq \max_k \{ T(n-1-k) + T(k) + c\}$
  - Constructive induction: $T(n) \leq a*n+b$

  $T(n) = \Theta(n)$
    - Pre- & post-order for ancestor preconditioning p.293

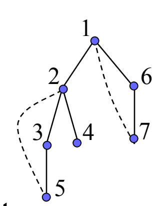
DFS

- Depth-first search: graph traversal
  - Unmark every vertex
  - Explore:
    - Push unmarked neighbors unto the stack, marking them
      - If instead of the stack we have queue then breadth-first search (BFS) is performed
    - You've seen them before, we use DFS for:
      - Finding articulation points
      - Topological sorting

DFS properties

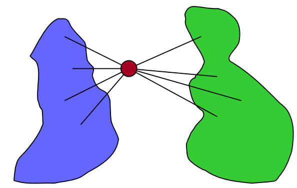


- Undirected graphs
  - Takes $\Theta(|E|+|V|)$ time
  - Builds a spanning tree T
    - Example

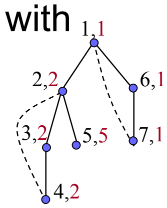


  - If not connected get a forest
  - Edges not in T connect a node to its ancestor (cannot cross to another branch)
  - Nodes of T indexed in pre-order (*prenum*)
    - Of course, depends on the starting node

Articulation points



- A node *v* of a connected graph
  - is an articulation point if deleting it with adjacent edges makes the graph disconnected



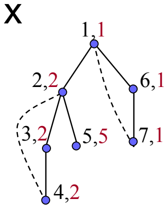
- Find them
- Define highest[v] = prenum of a highest node that can be reached going down the tree and at most one dashed link up

Articulation points

- Node v is an articulation point

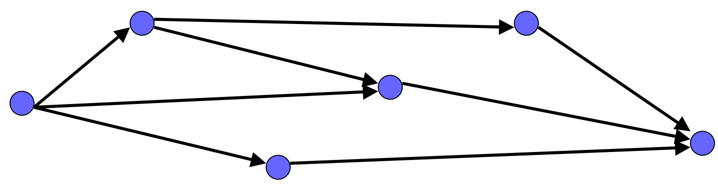
  if and only if it has at least one child x such that `highest[x]>=prenum[v]`

  - Indeed then subtree rooted at x will be separated from the rest of the graph
  - Root is articulated if it has more than one child
  - highest[v] = min(prenum[v], prenum[w], highest[u]) over all w's connected to v by dashed line and all children u
    - » (this is how we compute highest values)



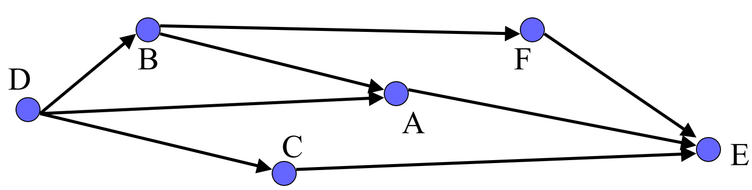
Topological sorting

- DAGs: directed acyclic graphs
  - More general than trees
  - Represent partial orderings
    - Set inclusion
    - Project dependencies
      - Nodes = stages, edges = activities



Topological sorting

- Topological ordering
  - Nodes indexed such that if there is an edge from x to y then x<y



  - Do DFS and post-order gives the reverse of what we need (e.g.DCBFAE) proof?

Breadth-first search (BFS)

- Queue instead of stack
- Not naturally recursive
- Trees and dashed edges look different
  - No links within branches, links across
- Useful when we have infinite search trees (e.g. implicitly specified)
- Useful when we wanna find the shortest path (solution)

Backtracking

- Exploring implicit graph
  - similar to DFS in directed graph
- Solution consists of parts
  - Choice which to add
  - Knapsack: N types of objects
    - e.g. {(2oz, $3) (3oz, $4), (5oz, $10)}
    - W = 10
    - [{},0] – root of the tree
    - [{2,5},$13], etc.

Eight queens problem

- No threatening
- Solutions:
  - C(64,8) approx. 4 billions
  - Vector of 8 numbers 8^8 approx 16 millions
  - Permutations 8! = 40,320
  - Backtracking
    - DFS: tree of k-promising vectors (size 2057)
    - One queen at a time
    - Check right away – only the lastly added queen

Branch and bound

- Looking for an optimal solution
  - Use bounds to prune the search tree
  - DFS or BFS
  - Example: assignment
    - Matrix Cost[x,y]
    - Minimize Σ_x Cost[x, a[x]] where a[x] is the assignment and a[x]!=a[y] when x!=y
    - Assign jobs with least costs one per worker