# Graph Search

Chirag Gosar

March 27, 2009

## Graph Search - Overview

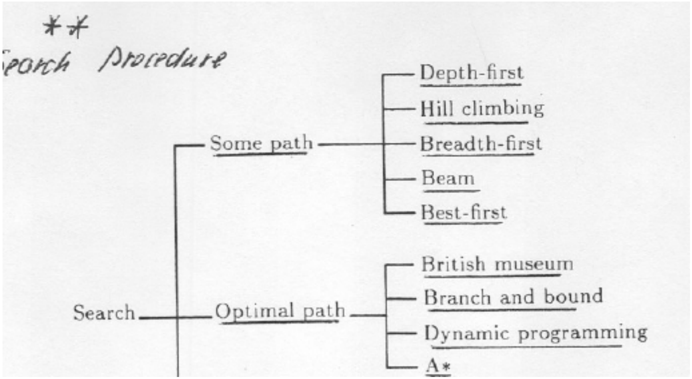
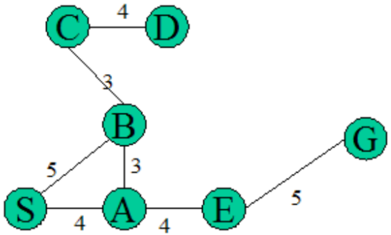
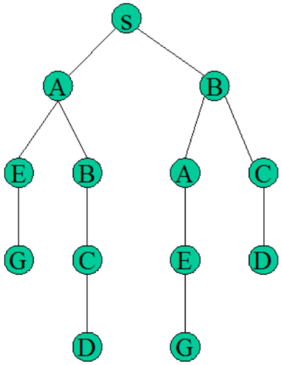


Figure: Search Overview

# Graph Search - Example





## Graph Search - DFS I

- Similar to a stack meaning Last In First Out (LIFO)
- Explores search space in a depth first manner
- Algorithm of Depth-First Search
    - Form a stack consisting of the root node.
    - Until the stack is empty or the goal has been reached, determine if the first element in the stack is the goal node.
        - If the first element is the goal node, do nothing
        - If the first element is not the goal node, remove the first element from the stack and add the first element's children, if any to the top of the stack
    - If the goal node has been found, annouce success; otherwise announce failure.

## Graph Search - DFS II

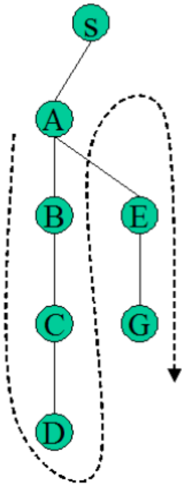


Figure: Depth First Search

## Graph Search - Hill Climbing I

- Uses Depth First Search with Estimated Remaining Distance Heuristics.
- The better the heuristic measure is, the better hill climbing will be relative to ordinary depth-first search.
- Algorithm of Hill Climbing
  - Form a stack consisting of the root node.
  - Until the stack is empty or the goal has been reached, determine if the first element in the stack is the goal node.
    - If the first element is the goal node, do nothing
    - If the first element is not the goal node, remove the first element from the stack, sort the first element's children if any by estimated remaining distance, and add the first element's children, if any to the top of the stack.
  - If the goal node has been found, annouce success; otherwise announce failure.

## Graph Search - Hill Climbing II

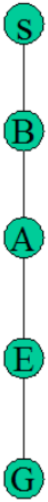


Figure: Hill Climbing

## Graph Search - Breadth-First Search I

- Uses Queue and follows a First in First Out (FIFO)
- Breadth-first search looks for the goal node among all nodes at a given level before using the children of those nodes to push on
- Algorithm of Breadth-First Search
  - Form a one-element queue consisting of the root node
  - Until the queue is empty or the goal has been reached, determine if the first element in the queue is the goal node.
    - If the first element is the goal node, do nothing.
    - If the first element is not the goal node, remove the first element from the queue and add the first element's children, if any, to the back of the queue.
  - If the goal node has been found, announce success; otherwise announce failure.

## Graph Search - Breadth-First Search II

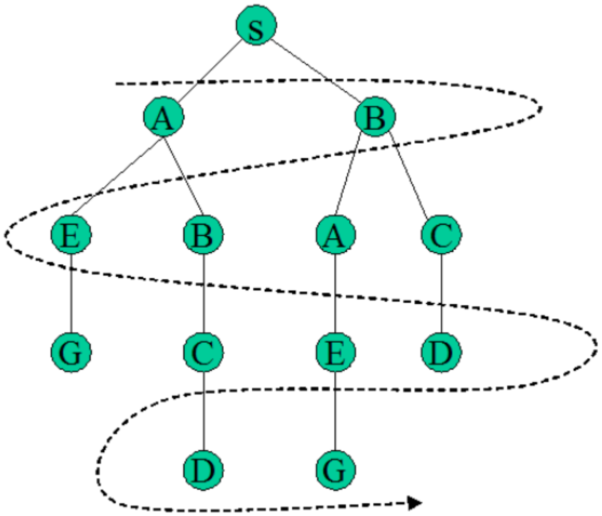


Figure: Breadth-First Search

## Graph Search - Beam Search I

- Beam Search uses Breadth First Search + the best N nodes heuristics
- This method prunes the search space.
- Algorithm
  - Beam search is like breadth-first search because beam search progresses level by level. Unlike breadth-first search, however, beam search only moves downward from the best N nodes at each level. The other nodes are ignored.

## Graph Search - Beam Search II

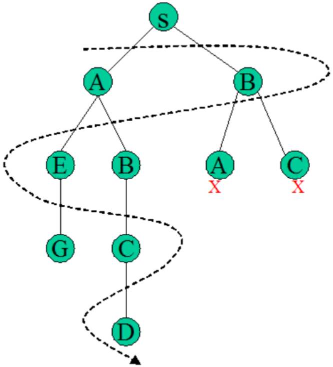


Figure: Beam Search

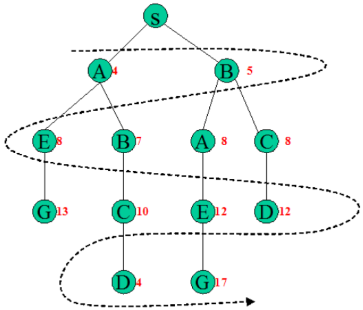
## Graph Search - Branch and Bound I

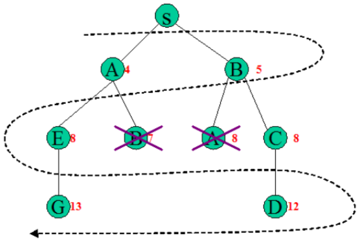
- Expands the least-cost partial path
- Algorithm
  - Form a queue of partial paths. Let the initial queue consist of the zero-length, zero-step path from the root node to nowhere.
  - Until the queue is empty or the goal has been reached, determine if the first path in the queue reaches the goal node.

    If the first path reaches the goal node do nothing.
    If the first path does not reach the goal node:

    - Remove the first path from the queue
    - Form new paths from the removed path by extending one step.
    - Add the new paths to the queue
    - Sort the queue by cost accumulated so far with least-cost paths in front.
  - If the goal node has been found, announce success; otherwise announce failure.

# Graph Search - Branch and Bound II





## Graph Search - A Star I

- Is a branch-and-bound search with an estimated remaining distance, combined with dynamic-programming principle.
- Add the total distance traveled for each path (as in branch & bound) but also add the remaining estimated distance, then pick the least lengthy path to explore.

## References I