CS 4649/7649
Robot Intelligence: Planning

Heuristics & Search

Sungmoon Joo

School of Interactive Computing
College of Computing
Georgia Institute of Technology

*Slides based on Dr. Mike Stilman's lecture slides

Course Info.

- **Course Website: joosm.github.io/RIP2014**
- **Course Wiki: github.com/RIP2014/RIP2014Wiki/wiki**
  - add your contact info, start grouping/filling in project ideas, etc.
  - github invitation sent (if you didn't get one, let me know)
  - S/W tutorial

- **RIM seminar :** http://www.robotics.gatech.edu/hg/item/318301
  - Steven M. LaValle – Planning expert, Virtual Reality
  - Friday, September 5, 2014
    12:00~13:00
  - Marcus Nanotechnology Building

State Space vs. Plan Space vs. Graph Space

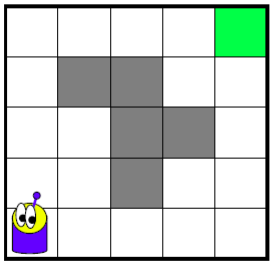
	State Space (60s –)	Plan Space (70s –)	Graph Space (90s -)
Algorithm	Progression Regression	Partial Order Planning	Iterates Graph Building Regression Search
Nodes	World States	Partial Plans	Graph Levels
Transitions	Actions -Move(x,y,z) -Load(x,y) -Open(r)	Plan Refinement Operations -Adding Steps -Promotion -Demotion	Sets of Actions -Constraint Coding -Mutual Exclusion

Search from Initial State to Goal

Nodes and Cells Represent States

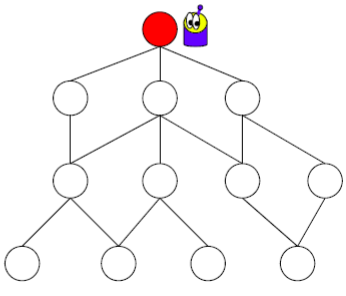
Grid

Actions – Move to **Neighbor Cell**



Graph

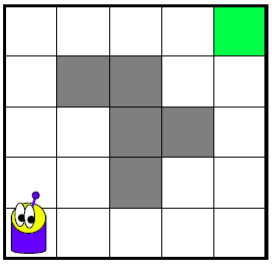
Actions – Move to **Adjacent Node**

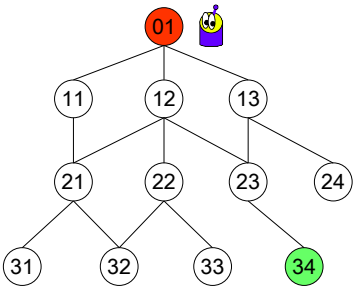


. . .

Uninformed Search

Subtitle: **We don't know anything** about the domain.

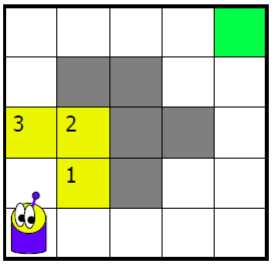


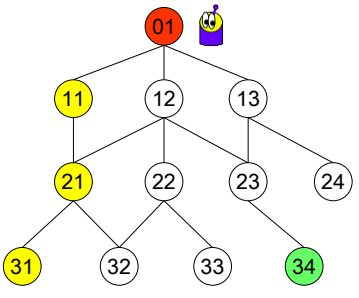


. . .

Uninformed Search: Depth First (DFS)

Moving along one branch

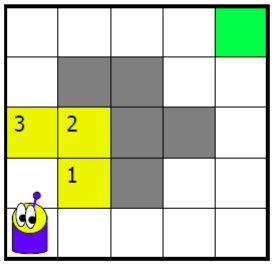


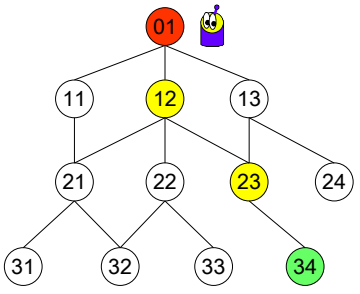


. . .

Uninformed Search: Depth First (DFS)

Moving along one branch

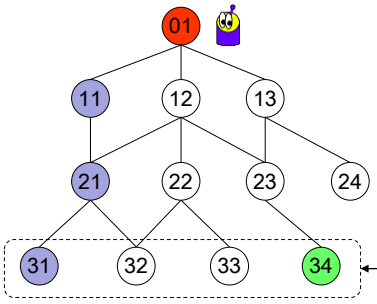


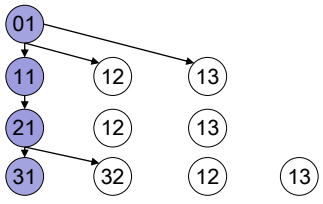


. . .

Uninformed Search: Depth First (DFS)

Take a look at the QUEUE

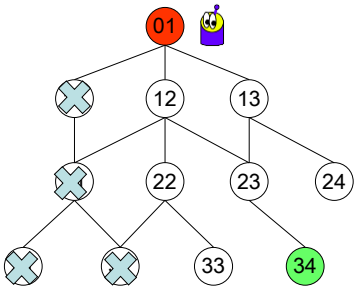


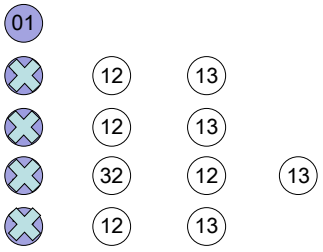


Expand until a node on the terminal level appears

Uninformed Search: Depth First (DFS)

Take a look at the QUEUE

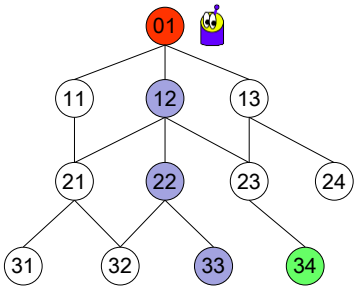


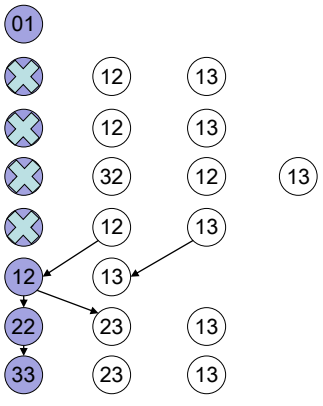


Check the terminal nodes
If not the goal, delete them and their parent nodes

Uninformed Search: Depth First (DFS)

Take a look at the QUEUE

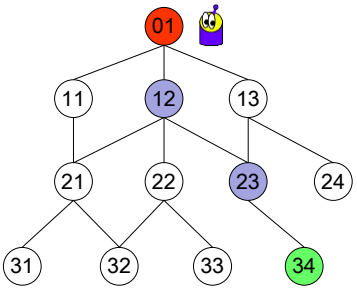




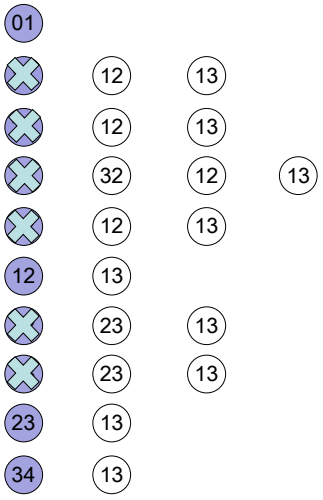
Explore another branch

Uninformed Search: Depth First (DFS)

Take a look at the QUEUE



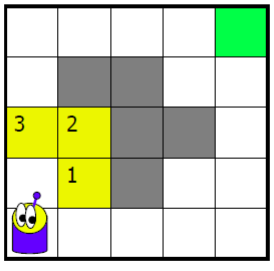
Repeat until you reach the goal

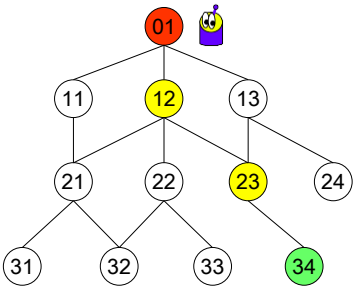


Uninformed Search: Depth First (DFS)

Advantages?

Disadvantages?

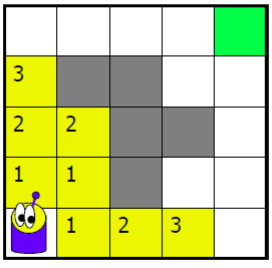


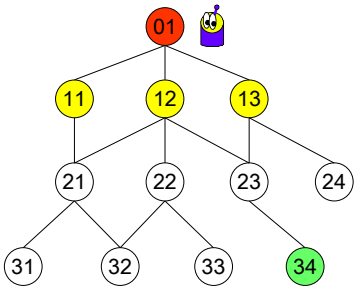


. . .

Uninformed Search: Breadth First (BFS)

Visiting every branch

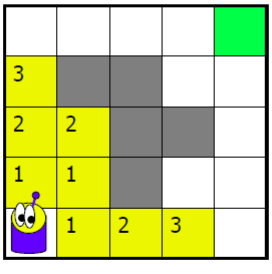


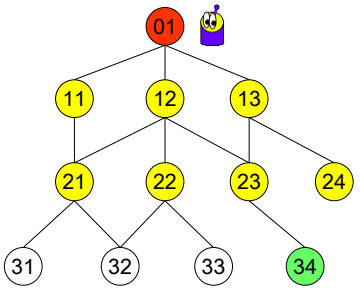


. . .

Uninformed Search: Breadth First (BFS)

Visiting every branch

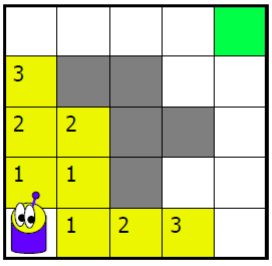


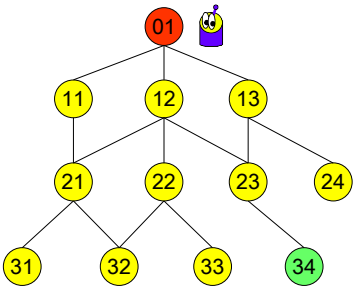


. . .

Uninformed Search: Breadth First (BFS)

Visiting every branch

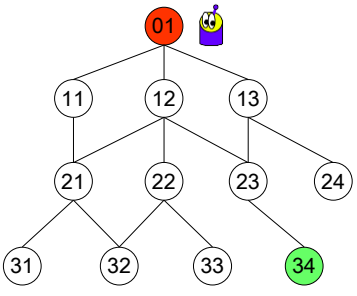




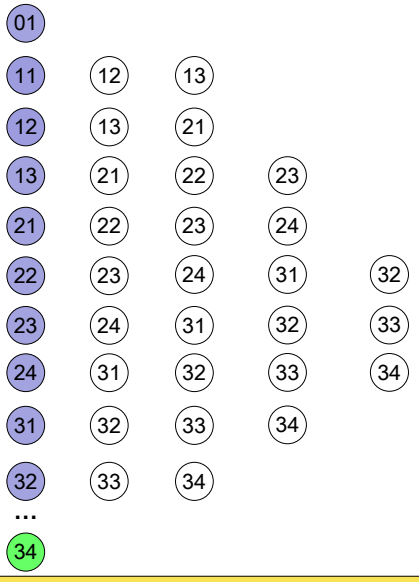
. . .

Uninformed Search: Breadth First (BFS)

Take a look at the QUEUE

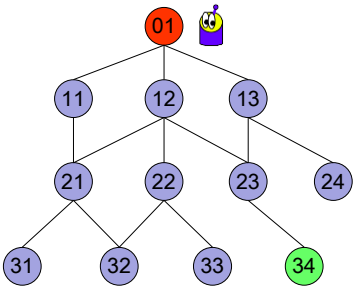


Explore every branch, level by level

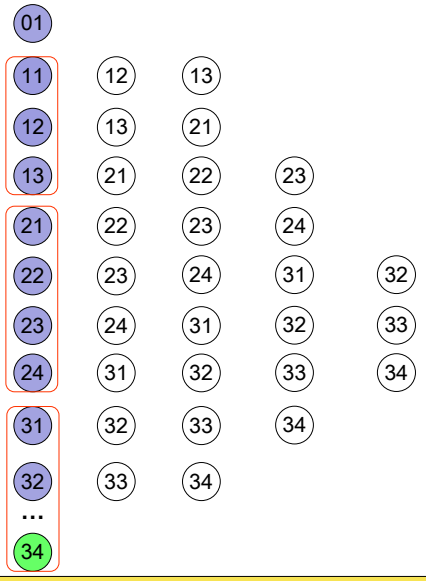


Uninformed Search: Breadth First (BFS)

Take a look at the QUEUE

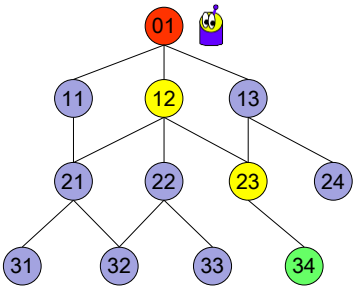


Every node on the given level is checked before any node on a later level is checked!

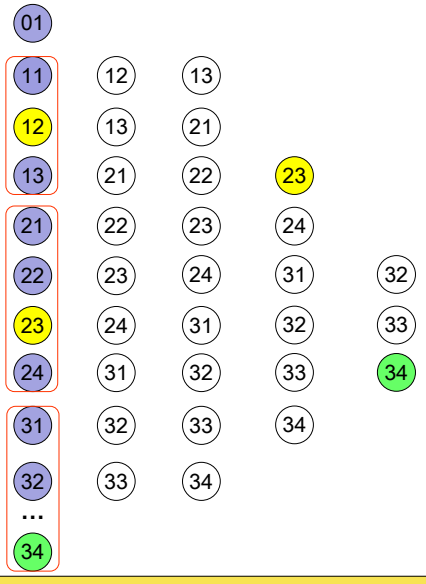


Uninformed Search: Breadth First (BFS)

Take a look at the QUEUE



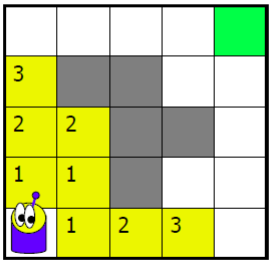
Every node on the given level is checked before any node on a later level is checked!

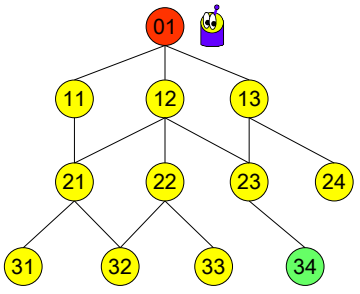


Uninformed Search: Breadth First (BFS)

Advantages?

Disadvantages?

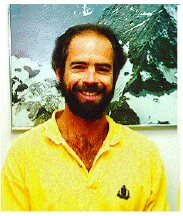




. . .

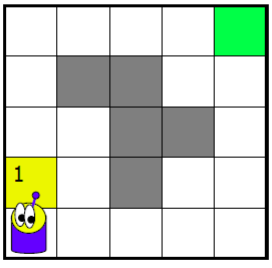
Uninformed Search: Iterative Deepening (IDS)

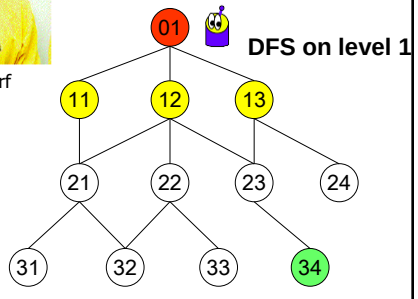
Korf '85
Stickel & Tyson '85



Richard Korf

DFS, level by level

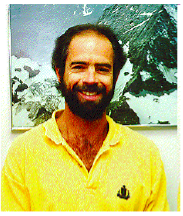




. . .

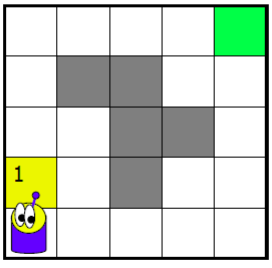
Uninformed Search: Iterative Deepening (IDS)

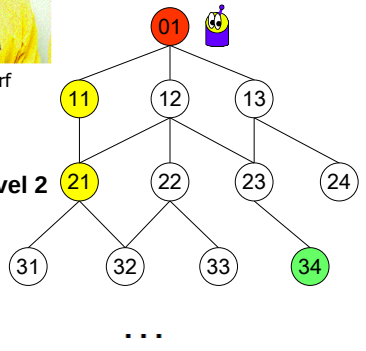
Korf '85
Stickel & Tyson '85



Richard Korf

DFS, level by level



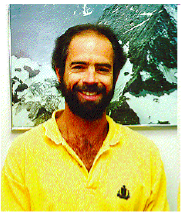


DFS on level 2

. . .

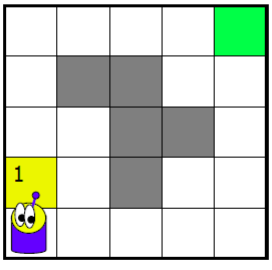
Uninformed Search: Iterative Deepening (IDS)

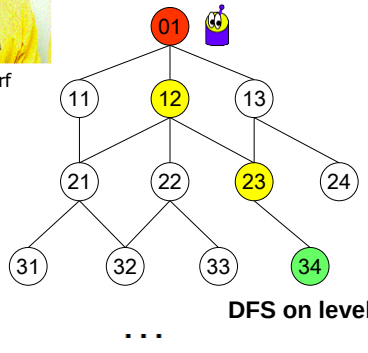
Korf '85
Stickel & Tyson '85



Richard Korf

DFS, level by level





DFS on level 3

. . .

Uninformed Search: Iterative Deepening (IDS)

What is the difference between IDS and BFS?

Much less memory at any given time
- first checked in the same order they would be checked in a breadth-first-search
- nodes are deleted as the search progresses

Usually better than plain DFS

when memory is not an issue

Main drawback

Redundancy

Efficiency in Planning

- Planning Efficiency: **speed** of a planner

Smart robots make **good** decisions.

Smarter robots make **good** decisions **fast**!

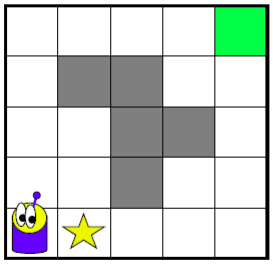
We like smart robots

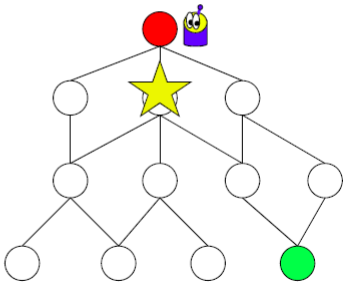
How can we make our robots smarter?

Informed or Heuristic Search

Subtitle: **We know which choices might be GOOD**

"Magic" Heuristic function: **h(s) = "cost to go"**

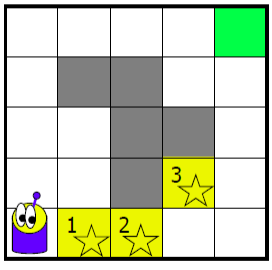


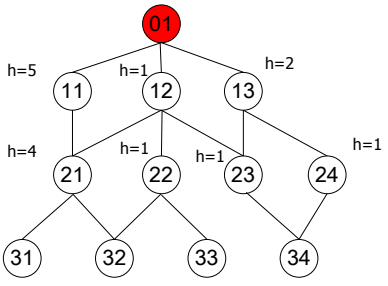


. . .

Informed Search: Best First Search

- h(s) - Tells us the expected cost of achieving the goal along a path
- Best First Search chooses the least heuristic cost node



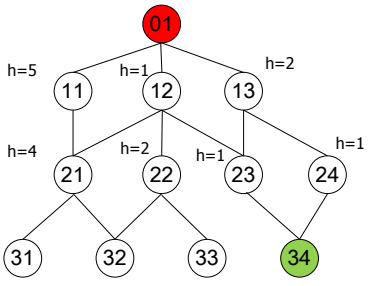


Informed Search: Best First Search

- OPEN = [initial state] CLOSED = []
- While OPEN is not empty
  do
  1. Remove the best node from OPEN, call it n, add it to CLOSED.
  2. If n is the goal state, backtrace path to n (through recorded parents) and return path.
  3. Create n's successors.
  4. For each successor do:
     a. If it is not in CLOSED and it is not in OPEN: evaluate it, add it to OPEN, and record its parent.
     b. Otherwise, if this new path is better than previous one, change its recorded parent.
        i. If it is not in OPEN add it to OPEN.
        ii. Otherwise, adjust its priority in OPEN using this new evaluation.

  done

Informed Search: Best First Search



Open	Successor	Closed	Parent
[01]		[]	
[11,12,13]	[11,12-n,13]	[01]	01
[11,13]	[21,22,23-n]	[12,01]	12
[21,22,11,13]	[34-n]	[23,12,01]	23

Properties of Heuristics: h(s)

Informed

- Does estimate lead to the goal?
- Accuracy of heuristic

Admissible

- h(s) $\leq$ true "cost to go"
- Is Best-First Search with Admissible h(s) optimal? NO! Why?

Informed Search: A* Search

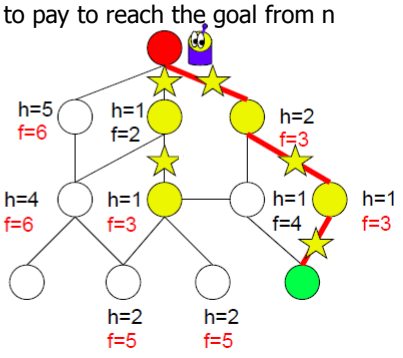
Value = Cost + Cost To Go

f(n) = g(n) + h(n)

Cost you already paid to reach node n

Cost you have to pay to reach the goal from n

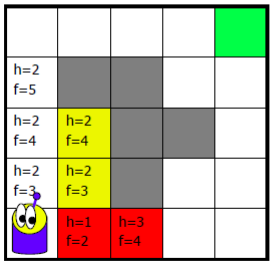
optimal path

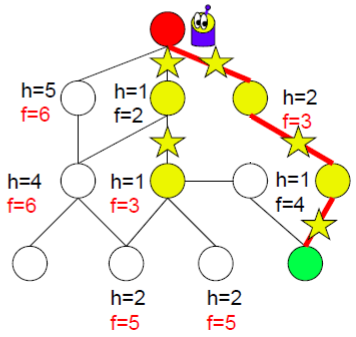


Informed Search: A* Search

Value = Cost + Cost To Go

f(n) = g(n) + h(n)





Variants of Heuristic Search

- Best-First
- A*
- Weighted A*
  - H(s) = Cost(s) + W × h(s)
  - Not admissible, but often works well
- Hill Climbing
  - Local Best-First Search
  - When stuck, randomly chooses new starting point
- "Enforced" Hill Climbing
  - Local Best-First Search
  - When stuck, perform breadth-first search until a better state is found