Exercises: Artificial Intelligence

The farmer, fox, goose and grain

Representation

- States of the form $[\mathcal{L}|\mathcal{R}]$, where:
  - $\mathcal{L}$: *Items on left bank*
  - $\mathcal{R}$: *Items on right bank*
- $\mathcal{L}$ and $\mathcal{R}$ contain:
  - Fa: *Farmer*
  - Fo: *Fox*
  - Go: *Goose*
  - Gr: *Grain*

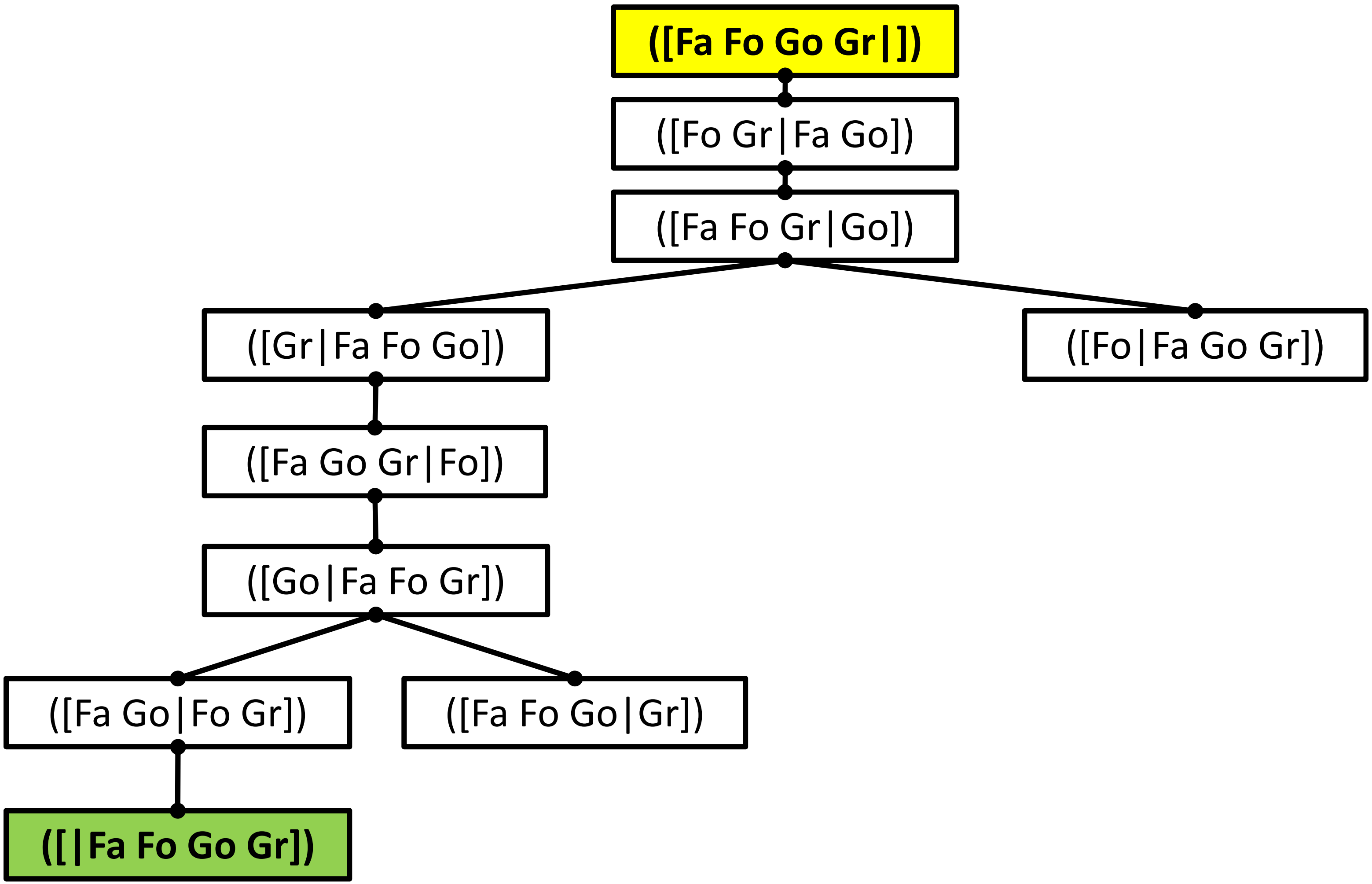
Representation

- Start: [Fa Fo Go Gr||]
- Goal: [|Fa Fo Go Gr]
- Rules:
  - R_1: [Fa $\mathcal{X}$|$\mathcal{Y}$] $\longrightarrow$ [$\mathcal{X}$|Fa $\mathcal{Y}$]
  - R_2: [$\mathcal{X}$|Fa $\mathcal{Y}$] $\longrightarrow$ [Fa $\mathcal{X}$|$\mathcal{Y}$]
  - R_3: [Fa z $\mathcal{X}$|$\mathcal{Y}$] $\longrightarrow$ [$\mathcal{X}$|Fa z $\mathcal{Y}$]
  - R_4: [$\mathcal{X}$|Fa z $\mathcal{Y}$] $\longrightarrow$ [Fa z $\mathcal{X}$|$\mathcal{Y}$]
  - No combination (Fo,Go) or (Go,Gr) on either bank, without the farmer.

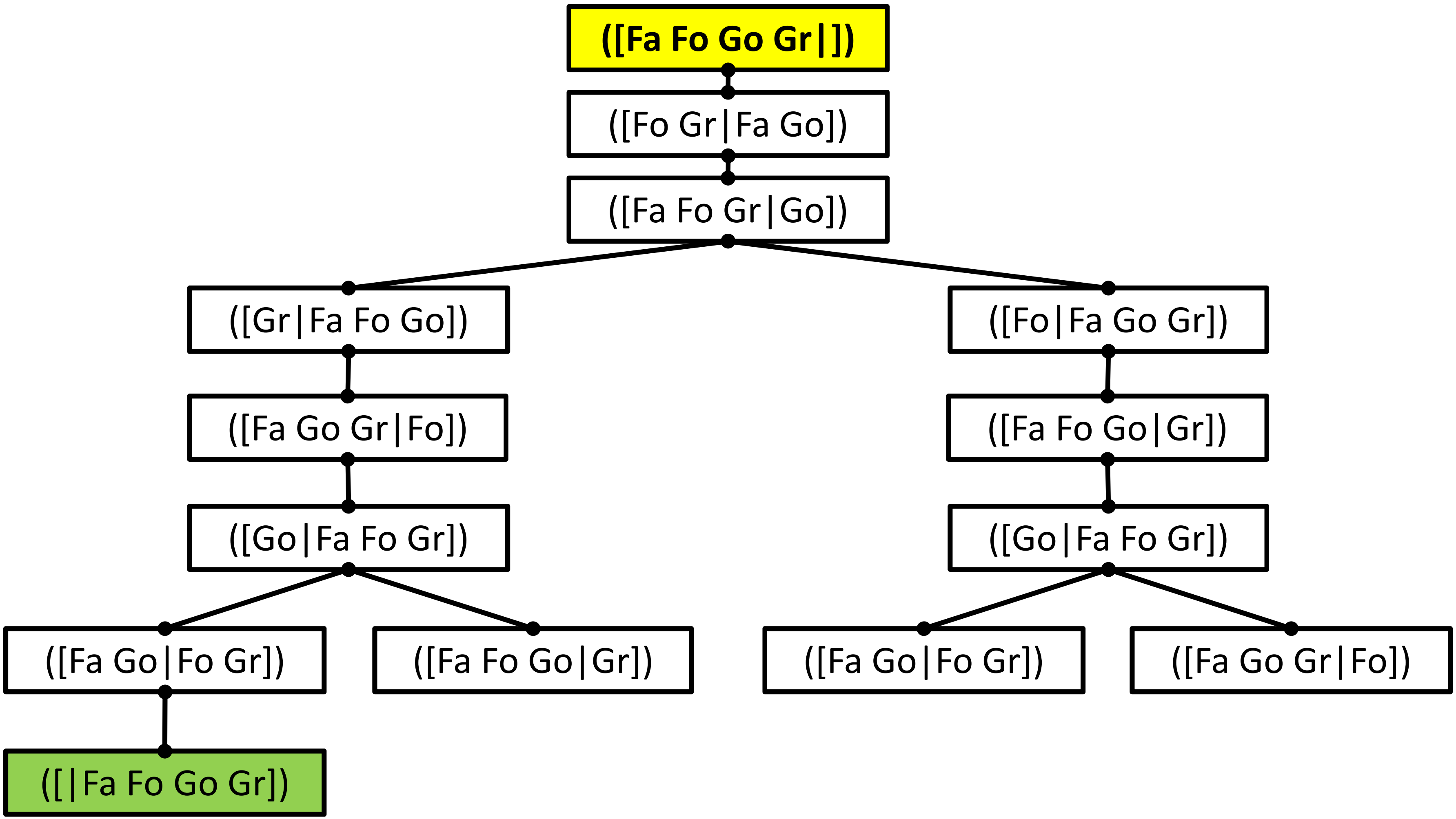
Depth-first search (queues)

- S = (<[Fa Fo Go Gr|]>)
- Q_1 = (<[Fa Fo Go Gr|][Fo Gr|Fa Go]>)
- Q_2 = (<[Fa Fo Go Gr|][Fo Gr|Fa Go][Fa Fo Gr|Go]>)
- Q_3 = (<[Fa Fo Go Gr|][Fo Gr|Fa Go][Fa Fo Gr|Go][Gr|Fa Fo Go]>,<[Fa Fo Go Gr|][Fo Gr|Fa Go][Fa Fo Gr|Go][Fo|Fa Go Gr]>)
- Q_4 = (<[Fa Fo Go Gr|][Fo Gr|Fa Go][Fa Fo Gr|Go][Gr|Fa Fo Go][Fa Go Gr|Fo]>,<[Fa Fo Go Gr|][Fo Gr|Fa Go][Fa Fo Gr|Go][Fo|Fa Go Gr]>)
- Q_5 = (<[Fa Fo Go Gr|][Fo Gr|Fa Go][Fa Fo Gr|Go][Gr|Fa Fo Go][Fa Go Gr|Fo][Go|Fa Fo Gr]>,<[Fa Fo Go Gr|][Fo Gr|Fa Go][Fa Fo Gr|Go][Fo|Fa Go Gr]>)
- Q_6 = (<[Fa Fo Go Gr|][Fo Gr|Fa Go][Fa Fo Gr|Go][Gr|Fa Fo Go][Fa Go Gr|Fo][Go|Fa Fo Gr][Fa Go|Fo Gr]>,<[Fa Fo Go Gr|][Fo Gr|Fa Go][Fa Fo Gr|Go][Gr|Fa Fo Go][Fa Go Gr|Fo][Go|Fa Fo Gr][Fa Fo Go|Gr]>,<[Fa Fo Go Gr|][Fo Gr|Fa Go][Fa Fo Gr|Go][Fo|Fa Go Gr]>)
- G = (<[Fa Fo Go Gr|][Fo Gr|Fa Go][Fa Fo Gr|Go][Gr|Fa Fo Go][Fa Go Gr|Fo][Go|Fa Fo Gr][Fa Go|Fo Gr][|Fa Fo Go Gr]>,<[Fa Fo Go Gr|][Fo Gr|Fa Go][Fa Fo Gr|Go][Gr|Fa Fo Go][Fa Go Gr|Fo][Go|Fa Fo Gr][Fa Fo Go|Gr]>,<[Fa Fo Go Gr|][Fo Gr|Fa Go][Fa Fo Gr|Go][Fo|Fa Go Gr]>)

Depth-first search (search tree)



Breadth-first search (search tree)



Exercises: Artificial Intelligence

Bidiretional Search

Bidirectional Search

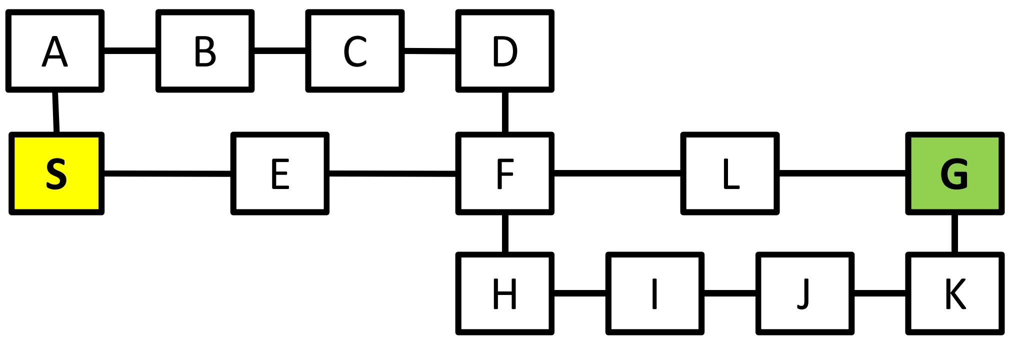
PROBLEM 1: BREADTH-FIRST?

Other methods than 2 x breadth-first

- Bidirectional search is complete for each combination with at least one complete search-strategy.
  - 2 x Breadth-first
  - 2 x Depth-first
  - Breadth-first and Depth-first
- Not each combination benefits from searching at both ends.

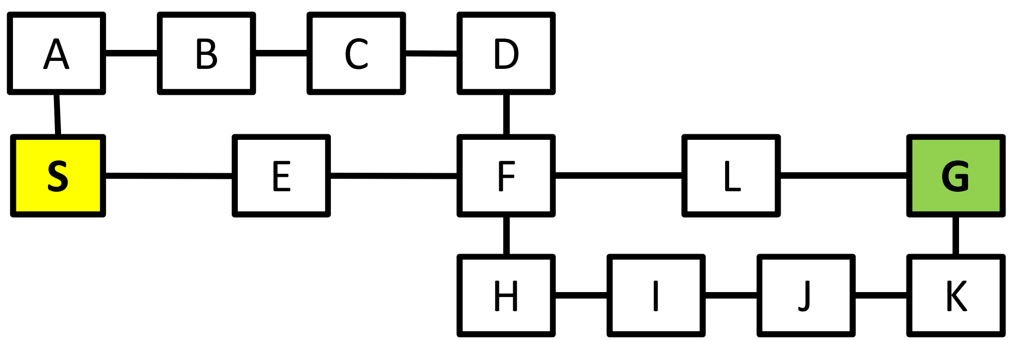
2 x Depth-first

- Forward:
  - (<S>) →(<SA>,<SE>) →(<SAB>,<SE>) →(<SABC>,<SE>) →(<SABCD>,<SE>) →**(<SABCDF>,<SE>)**
- Backward:
  - (<G>) →(<GK>,<GL>) →(<GKJ>,<GL>) →(<GKJI>,<GL>) →(<GKJIH>,<GL>) →**(<GKJIHF>,<GL>)**



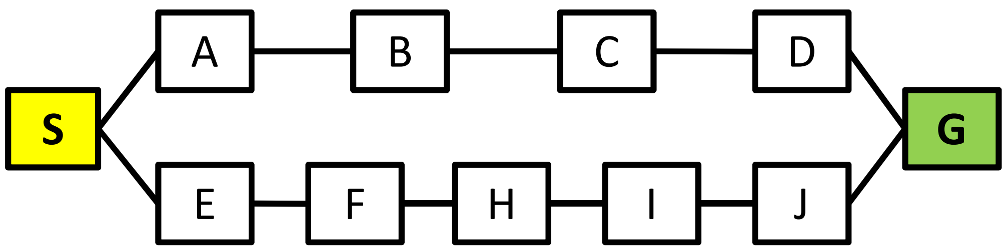
2 x Breadth-first

- Forward:
  - (<S>)→(<SA>,<SE>)→(<SE>,<SAB>)→**(<SAB>,<SEF>)**
- Backward:
  - (<G>)→(<GK>,<GL>)→(<GL>,<GKJ>)→**(<GKJ>,<GLF>)**



Breadth-first and Depth-first

- Forward (Breadth-first):
  - (<S>)→(<SA>,<SE>)→(<SE>,<SAB>)→(<SAB>,<SEF>) →**(<SEF>,<SABC>)**
- Backward (Depth-first):
  - (<G>)→(<GJ>,<GD>)→(<GJI>,<GD>)→(<GJIH>,<GD>) →**(<GJIHF>,<GD>)**

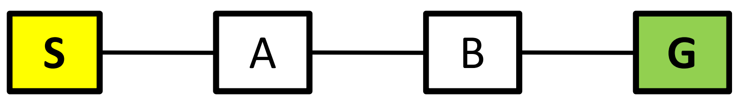


Bidirectional Search

PROBLEM 2: SHARED-STATE CHECK?

Replace shared-state check

- When only checking identical end-states, paths can cross each other unnoticed.
- Forward:
  - (<S>)→(<SA>)→(<SAB>)→(<SABG>)
- Backward:
  - (<G>)→(<GB>)→(<GBA>)→(<GBAS>)



Exercises: Artificial Intelligence

Beam Search

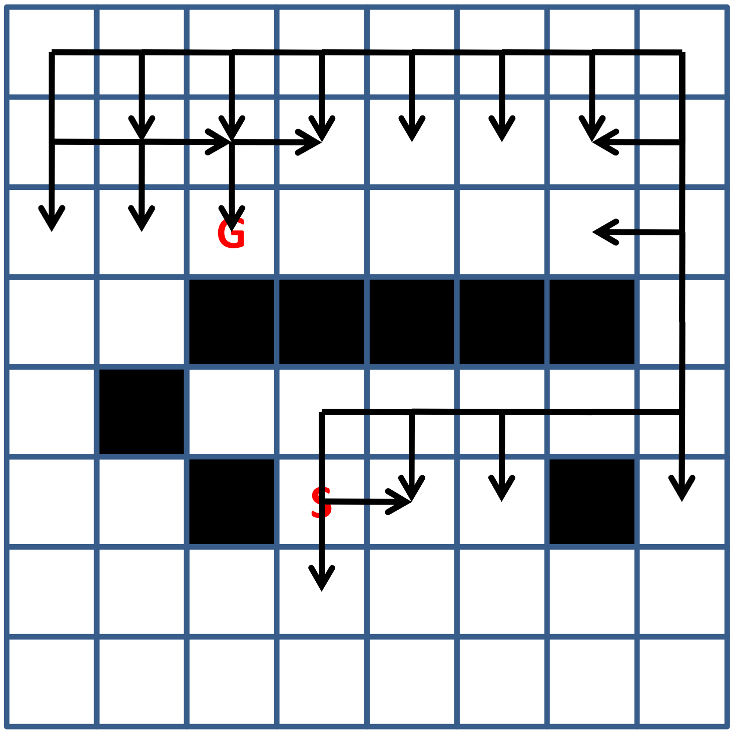
Beam Search

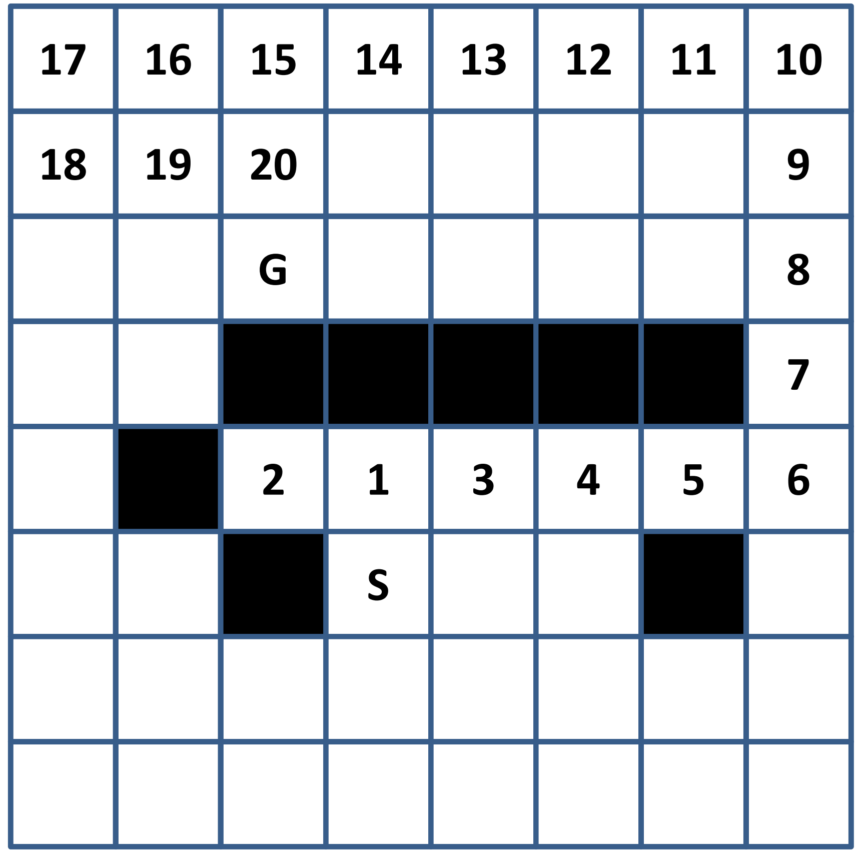
- ***Input:***
  - **QUEUE**: Path only containing root
  - **WIDTH**: Number
- ***Algorithm:***
  - **WHILE** (QUEUE not empty && goal not reached) **DO**
    - Remove ***all paths*** from QUEUE
    - Create paths to all children (of all paths)
    - Reject paths with loops
    - ***Sort new paths (according to heuristic)***
    - ***(Optimization: Remove paths without successor)***
    - Add WIDTH ***best paths*** to QUEUE
  - **IF** goal reached
    - **THEN** success
    - **ELSE** failure

Exercises: Artificial Intelligence

Path Search

Depth-first Search

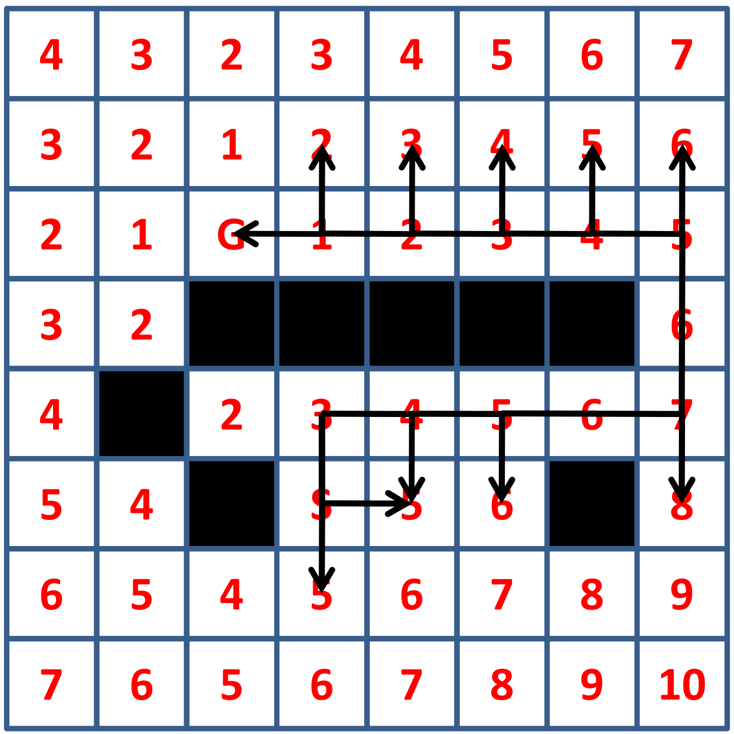


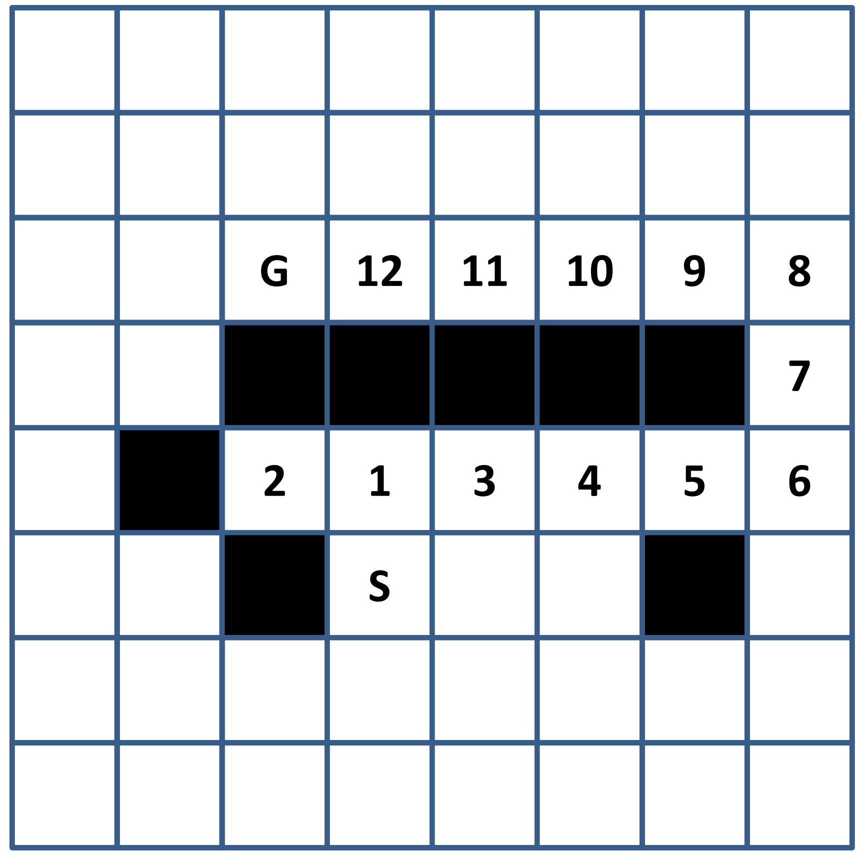


Heuristic: Manhattan Distance

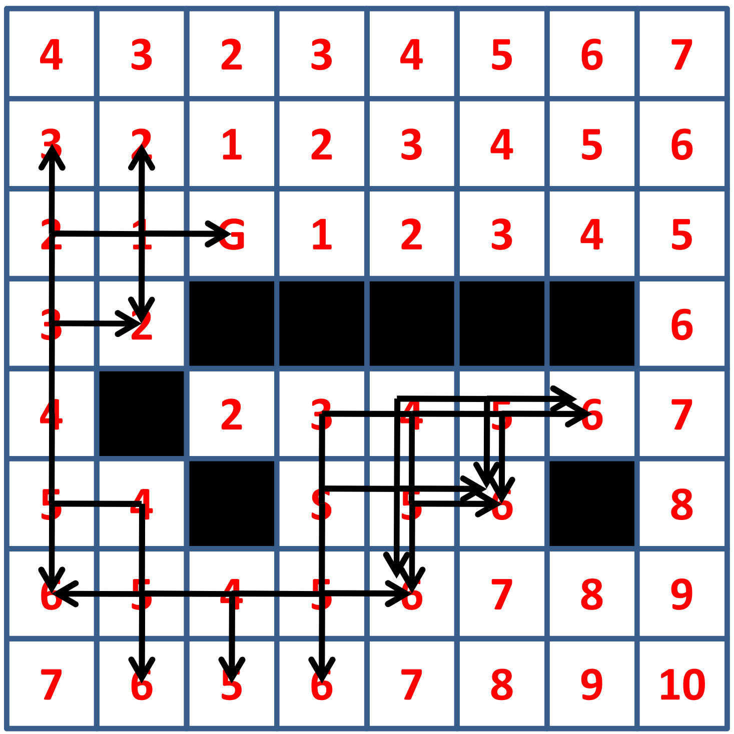
4	3	2	3	4	5	6	7
3	2	1	2	3	4	5	6
2	1	0	1	2	3	4	5
3	2						6
4		2	3	4	5	6	7
5	4		4	5	6		8
6	5	4	5	6	7	8	9
7	6	5	6	7	8	9	10

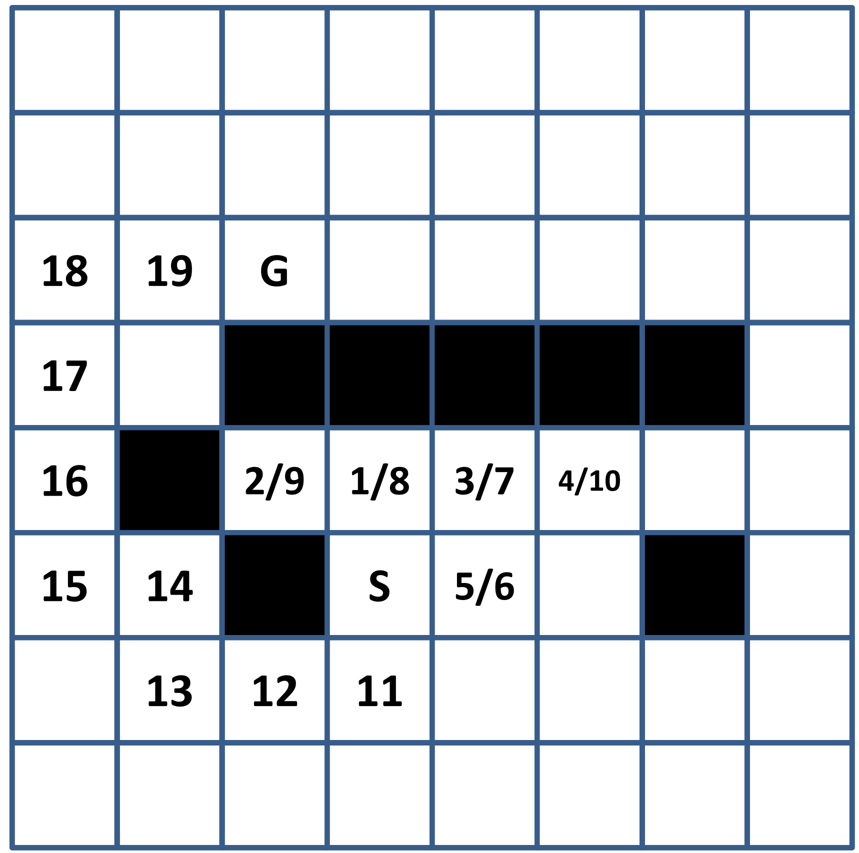
Hill-climbing I Search





Greedy Search





Exercises: Artificial Intelligence

Water Jugs

Representation

- States of the form [x,y], where:
  - x: *contents of 4 liter jug*
  - y: *contents of 3 liter jug*
- Start: [0,0]
- Goal: [2,0]

Representation

- Rules:
  - Fill x: $[x,y] \wedge x < 4 \longrightarrow [4,y]$
  - Fill y: $[x,y] \wedge y < 3 \longrightarrow [x,3]$
  - Empty x: $[x,y] \wedge x > 0 \longrightarrow [0,y]$
  - Empty y: $[x,y] \wedge y > 0 \longrightarrow [x,0]$
  - Fill x with y: $[x,y] \wedge x+y > 4 \wedge y > 0 \longrightarrow [4,(x+y-4)]$
  - Fill x with y: $[x,y] \wedge x+y \leq 4 \wedge y > 0 \longrightarrow [(x+y),0]$
  - Fill y with x: $[x,y] \wedge x+y > 3 \wedge x > 0 \longrightarrow [(x+y-3),3]$
  - Fill y with x: $[x,y] \wedge x+y \leq 3 \wedge x > 0 \longrightarrow [0,(x+y)]$

Heuristic

- $H([x,y]) = f(x) + f(y)$
- $f(x)$ is defined as follows:

x	0	1	2	3	4
f(x)	2	1	0	1	3

  - We need a jug filled with 2 liter.
  - To obtain a jug filled with 2 liter we need a jug filled with either 1 or 3 liter.
  - We consider an empty jug better than a jug filled with 4 liter.

Hill-climbing II Search