Local Search, Part 1

Lecture 8

Chapter 4, Sections 4.1-4.2

Jim Rehg

College of Computing

Georgia Tech

February 1, 2016

Dominance

Reminder: Heuristic $h(n)$ is *admissible* if $h(n) <= h^*(n)$

If h1 and h2 *admissible* heuristics, and

$h2(n) >= h1(n)$ for all n,

Then h2 *dominates* h1, and

h2 is better for search

Why?

Note: $h'(n) = \max\{ h1(n), h2(n) \}$ is admissible and dominates h1 and h2

Dominant Heuristic is Better

In A* every node n with

$f(n) < C^*$ will be expanded

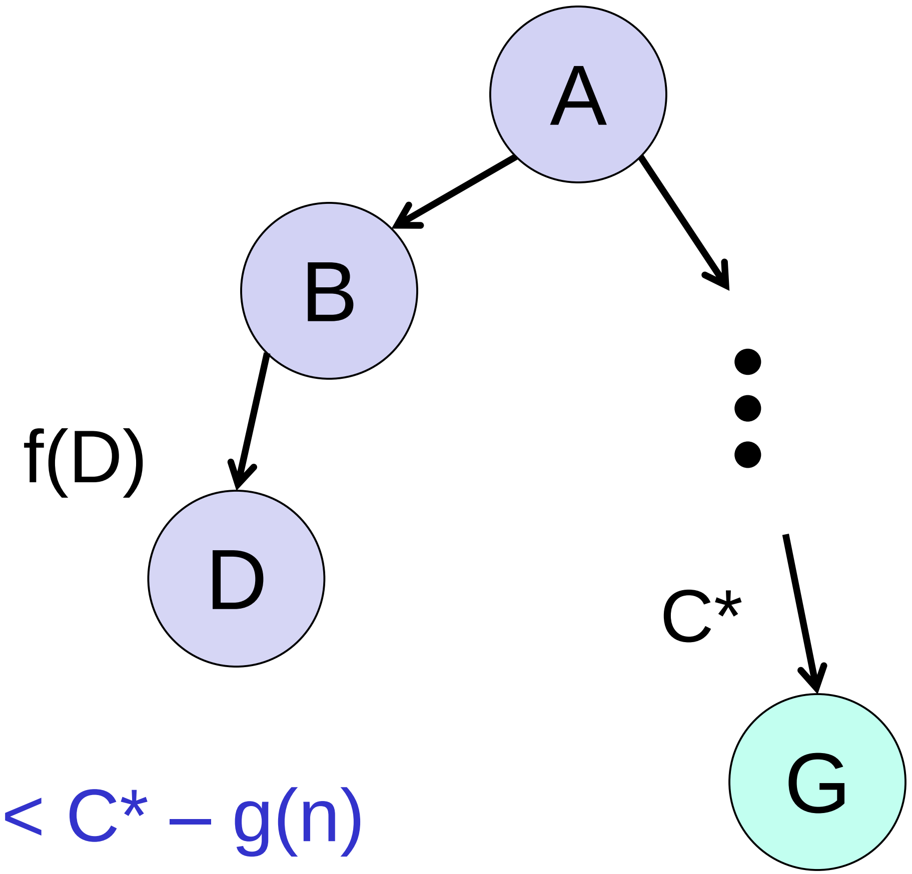
$h(n) < C^* - g(n)$ will be expanded

If $h2(n) > h1(n)$ for all n, then

Set of n for which $h1(n) < C^* - g(n)$

Will be *larger* than set of n for which $h2(n) < C^* - g(n)$

Thus h1 will expand more nodes



Local Search

In previous search problems

Solution = Path to goal state

In Local Search

Solution = Goal state itself

The path taken to the goal doesn't matter

Local Search Problems

What are some examples of local search problems?

Local Search Problems

What are some examples of local search problems?

- Circuit Design
- Class Scheduling
- Routing Planes/Ships
- Web Search

Optimization Problems in General

Local Search Formulation

Current state s

Evaluation (cost) function H(s)

Neighborhood of possible successors of s

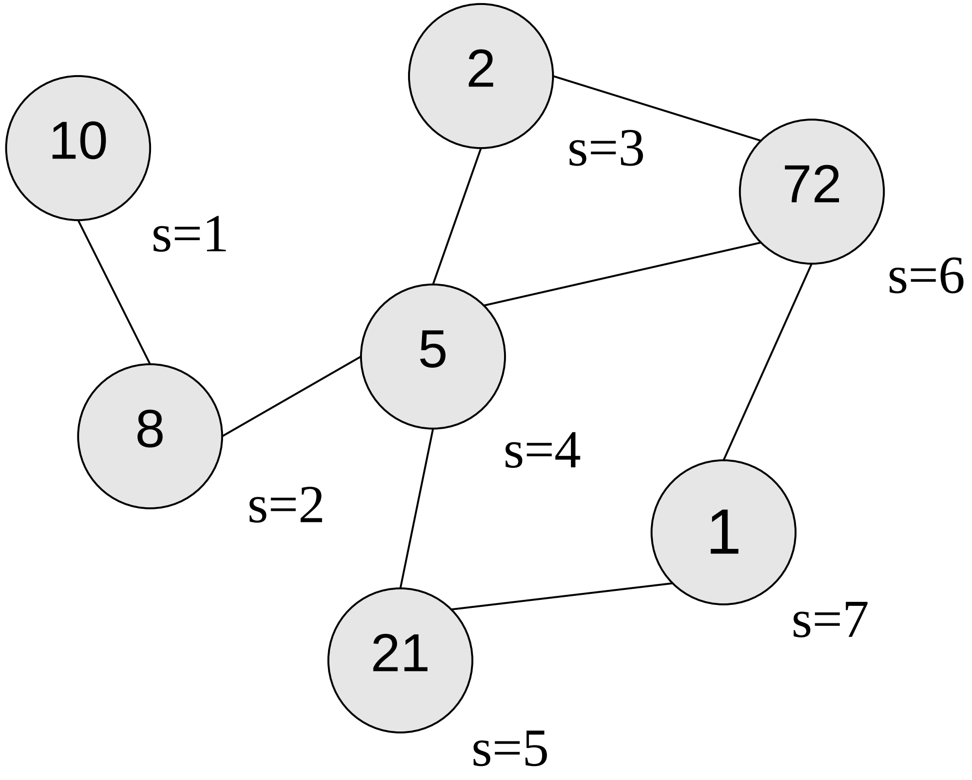
Goal:

Select s in S such that H(s*) is a minimum of H(s)*

Mathematically: $s^* = \arg\min_{s \in S} H(s)$

Local and Global Minimum

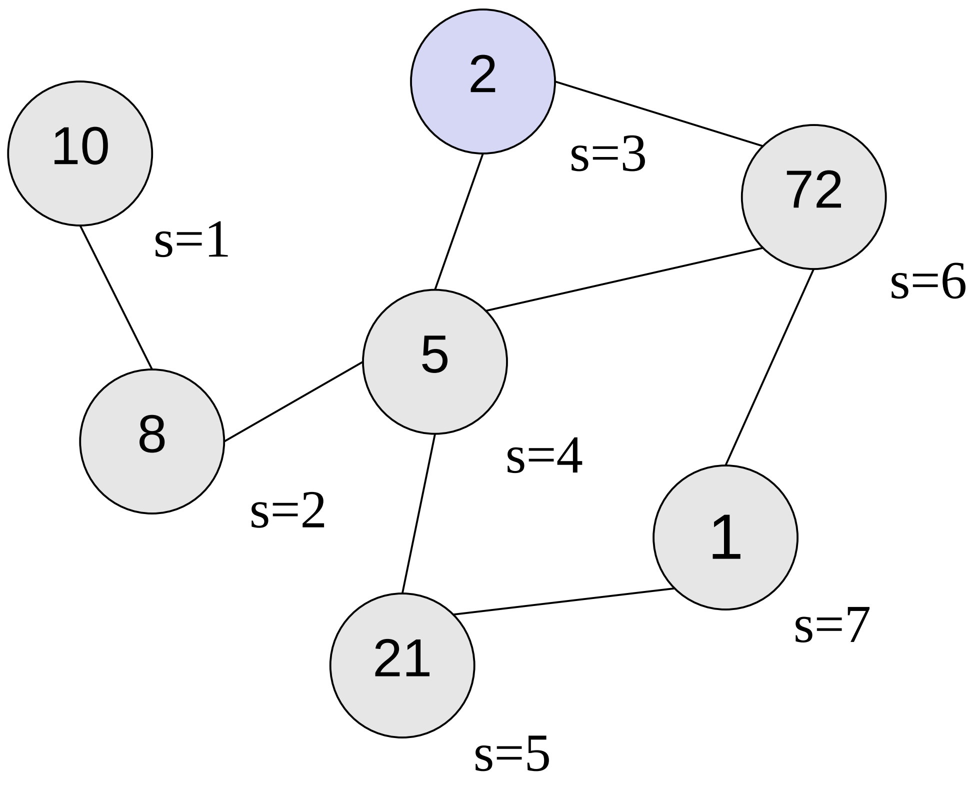
Discrete State



What are the minima?

Local and Global Minimum

Discrete State



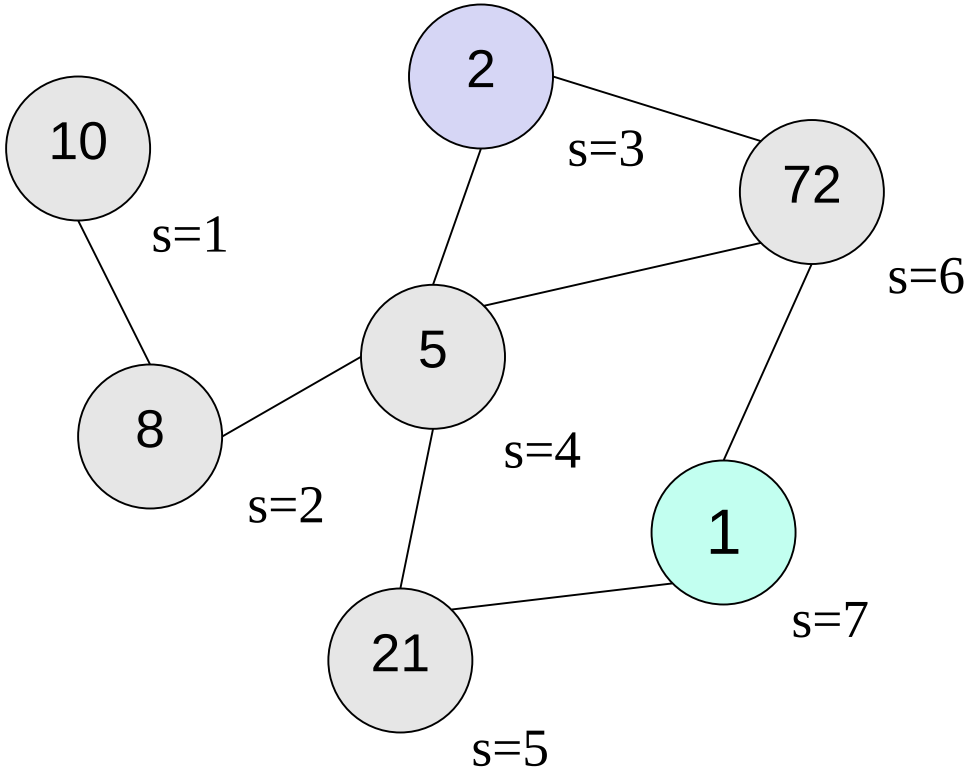
What are the minima?

Local Minimum

s=3   H(3)=2

Local and Global Minimum

Discrete State



What are the minima?

Local Minimum

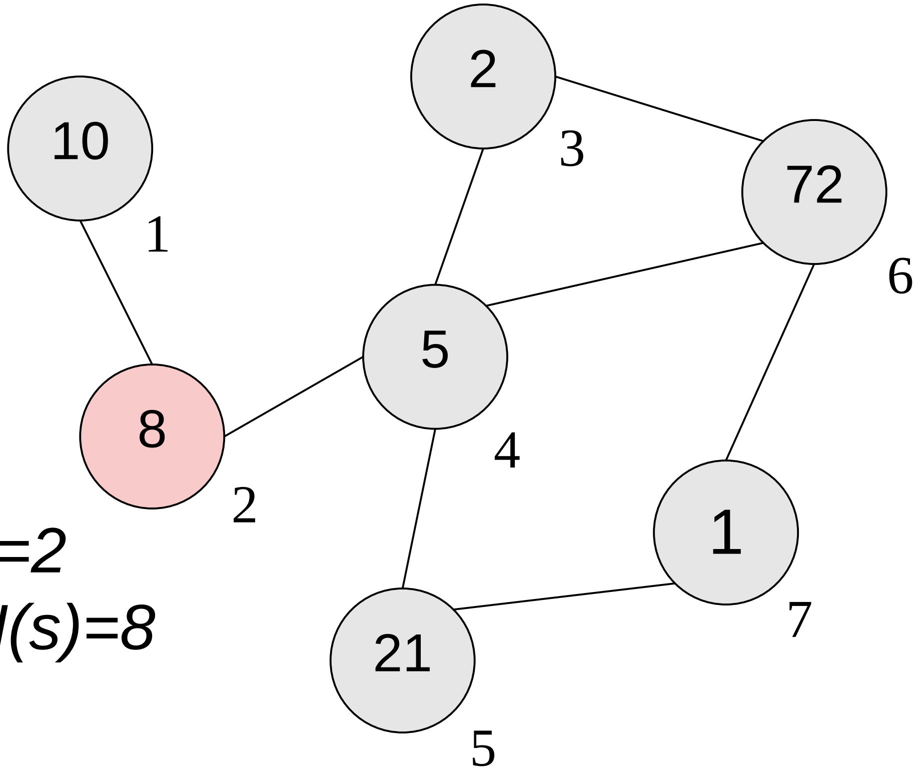
s=3 H(3)=2

Global Minimum

s=7 H(7)=1

Hill "Climbing" Search

Discrete State



s=2
H(s)=8

We will address minima but the principles apply to maxima also

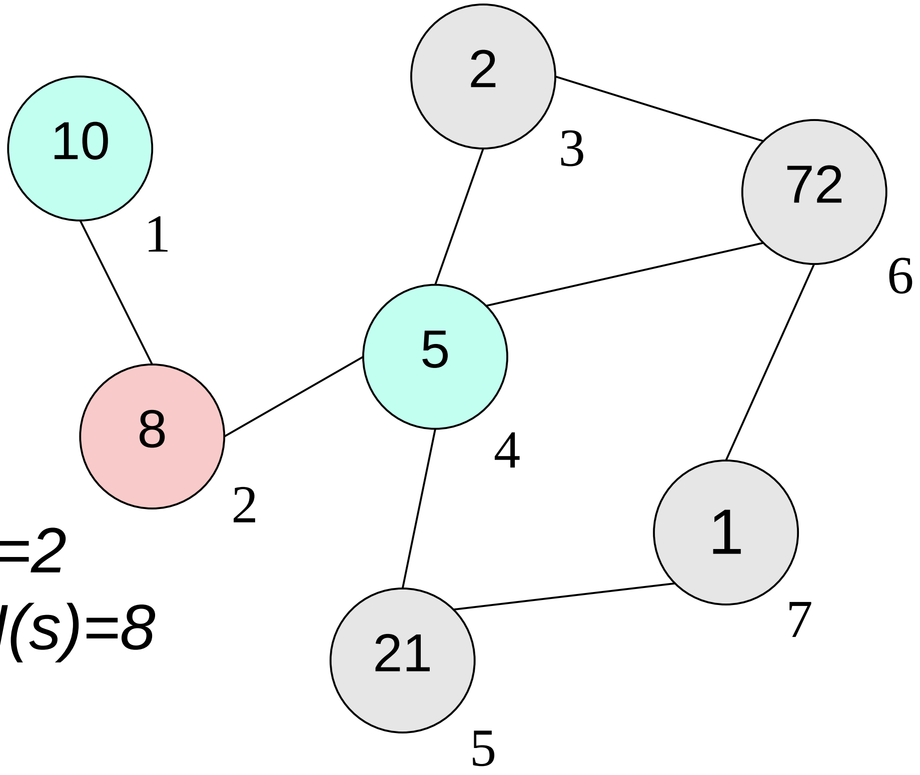
Initialize current state s

At each iteration:

- Expand s to obtain neighbors
- Select minimum cost neighbor s'
- If H(s') >= H(s) then return (s, H(s))
- s = s'

Hill “Climbing” Search

Discrete State



s=2
H(s)=8

We will address minima but the principles apply to maxima also

Initialize current state s

At each iteration:

Expand s to obtain neighbors

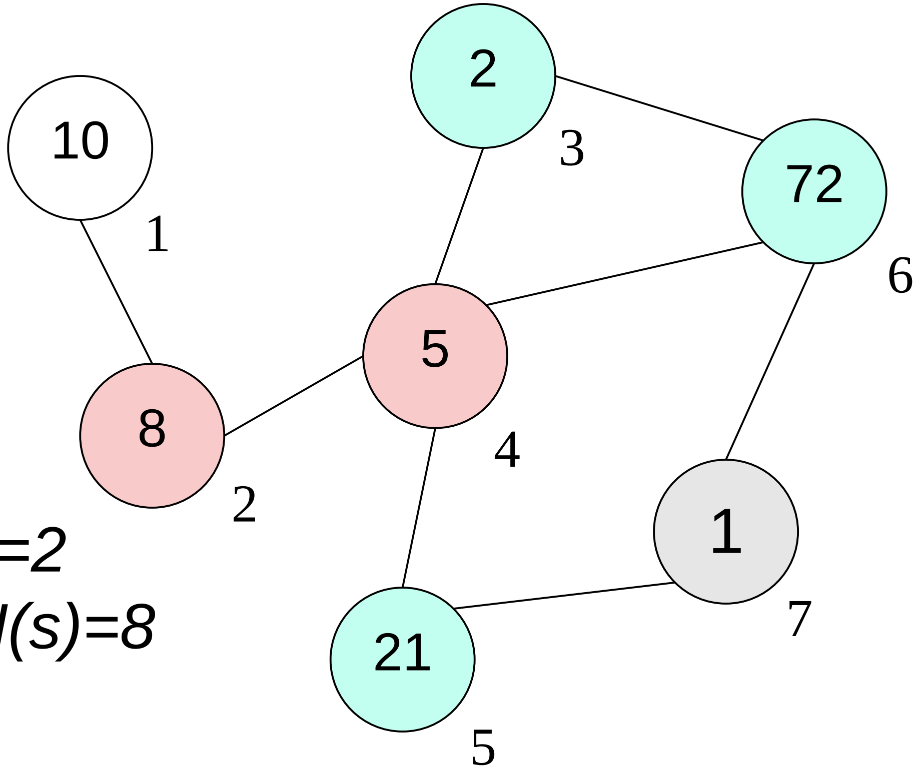
Select minimum cost neighbor s’

If H(s’) >= H(s) then return (s, H(s))

s = s’

Hill "Climbing" Search

Discrete State



s=2
H(s)=8

We will address minima but the principles apply to maxima also

Initialize current state s

At each iteration:

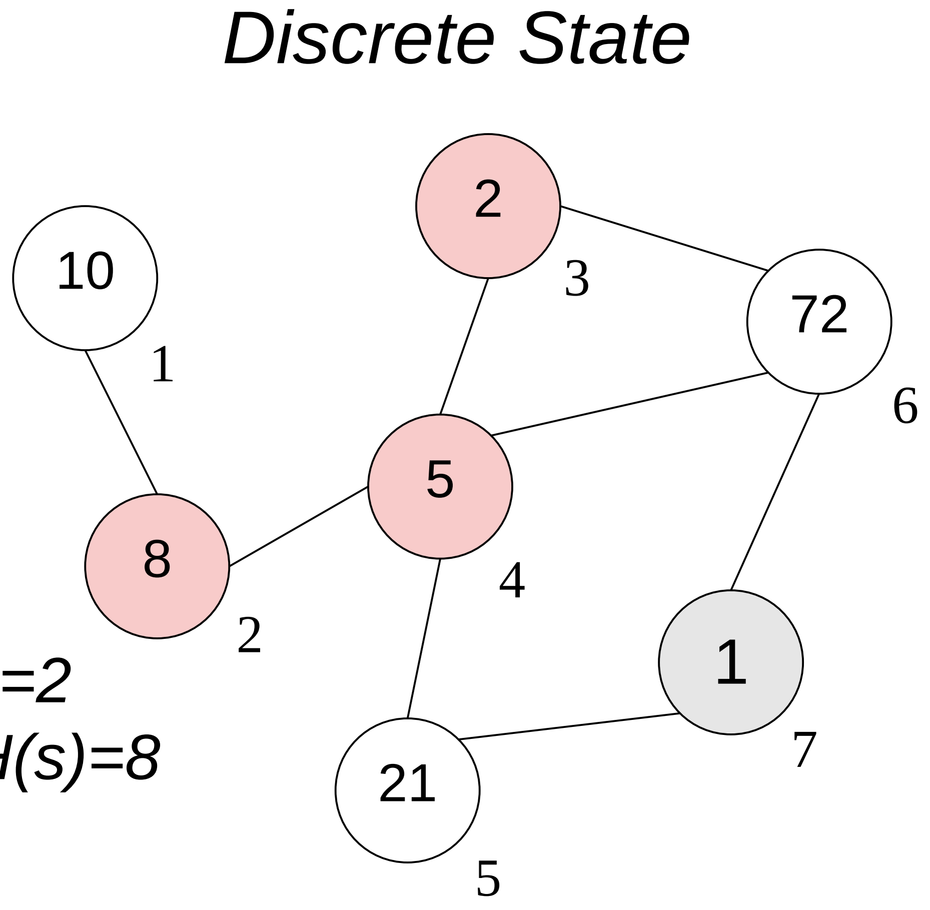
Expand s to obtain neighbors

Select minimum cost neighbor s'

If H(s') >= H(s) then return (s, H(s))

s = s'

Hill “Climbing” Search



s=2
H(s)=8

Terminate in *local minimum s=3, H(s)=2*

We will address minima but the principles apply to maxima also

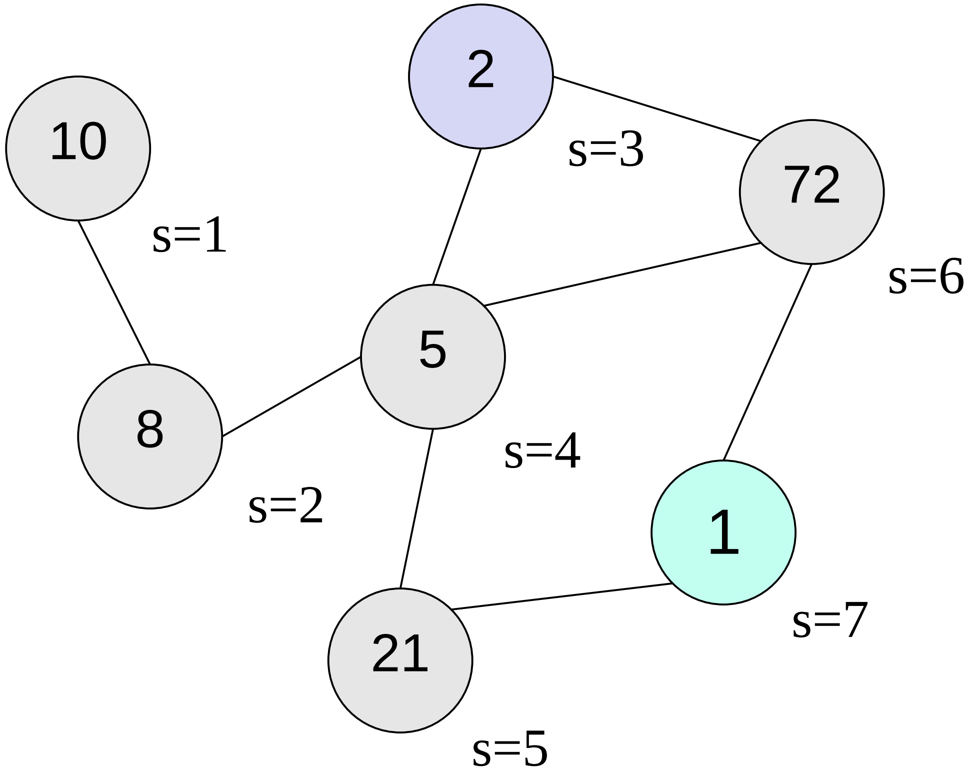
Initialize current state s

At each iteration:

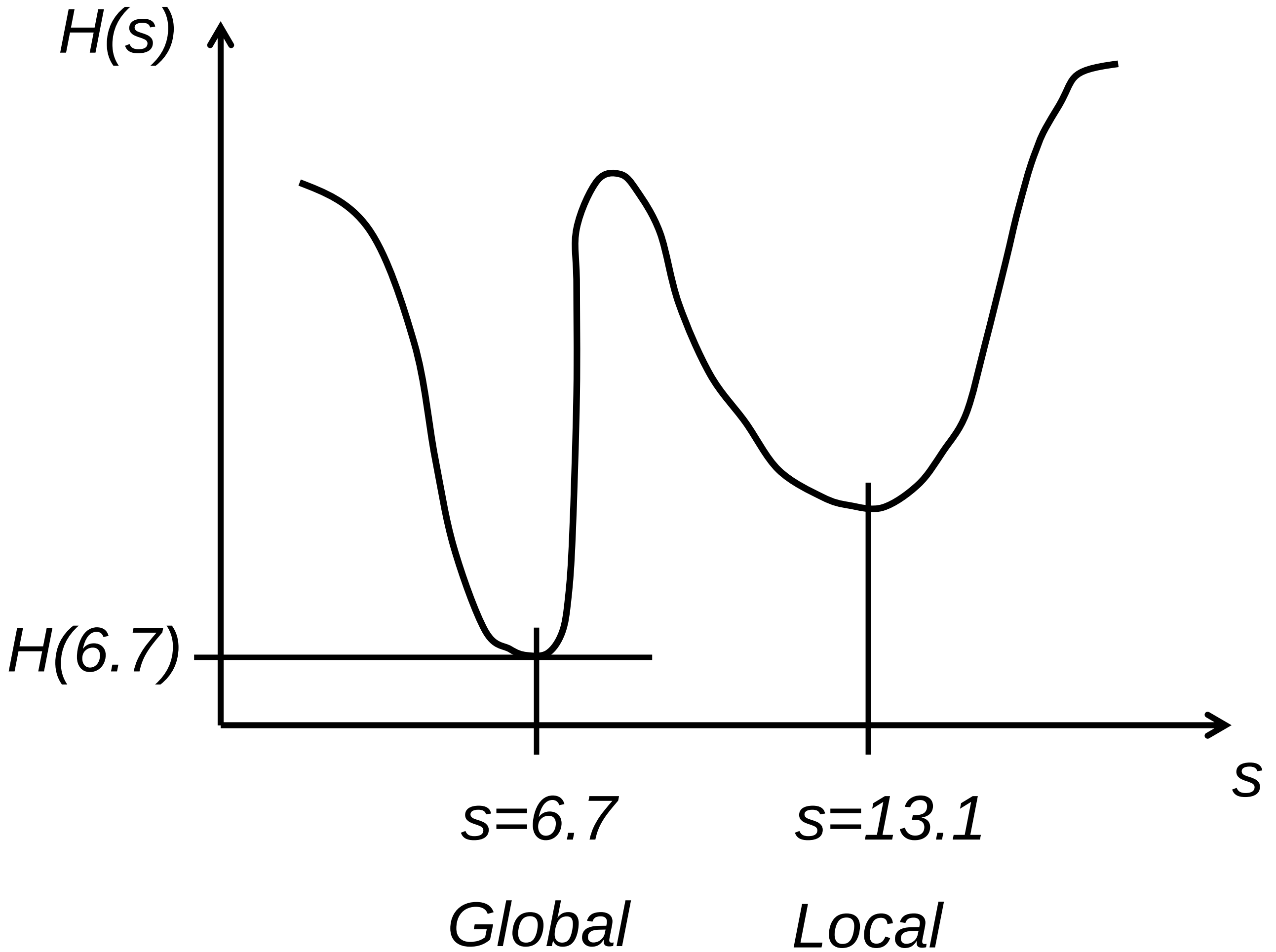
- Expand s to obtain neighbors
- Select minimum cost neighbor s’
- If H(s’) >= H(s) then return (s, H(s))
- s = s’

Local and Global Minimum

Discrete State



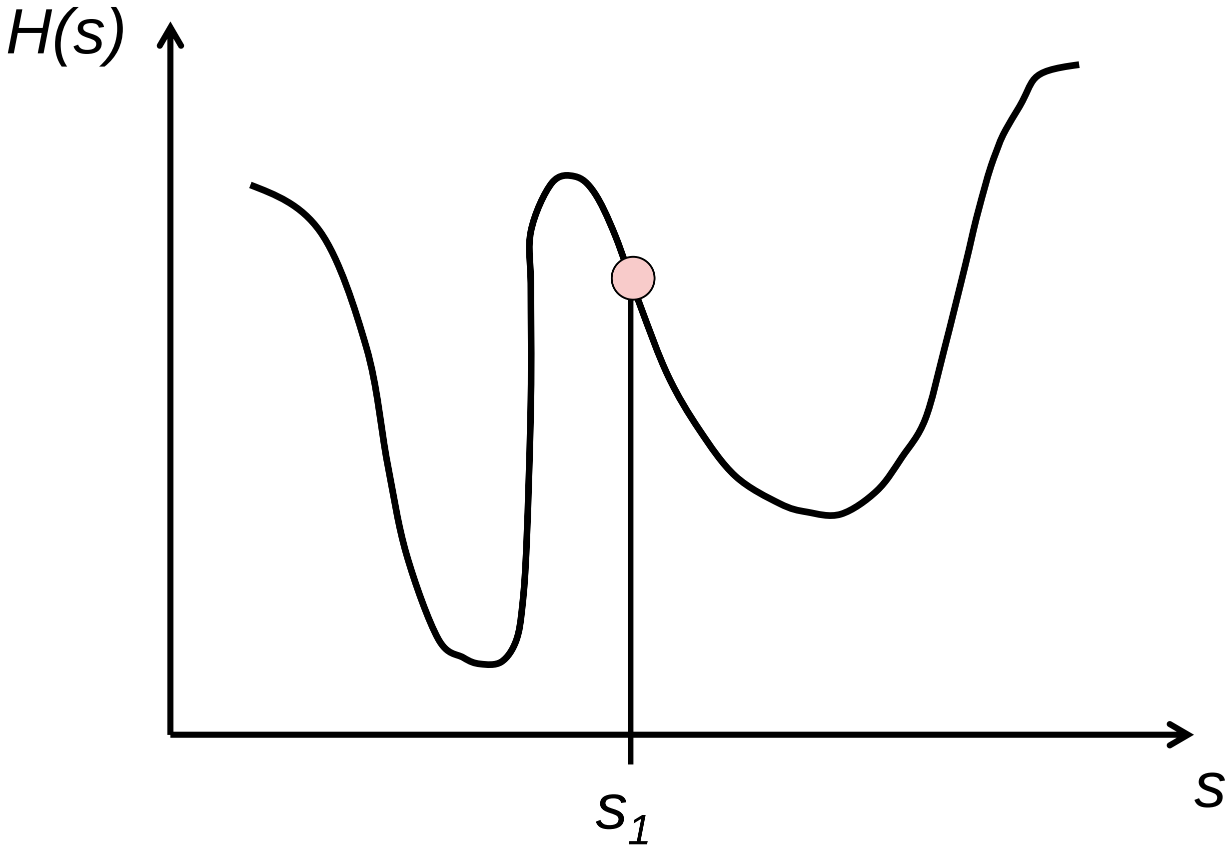
Continuous State



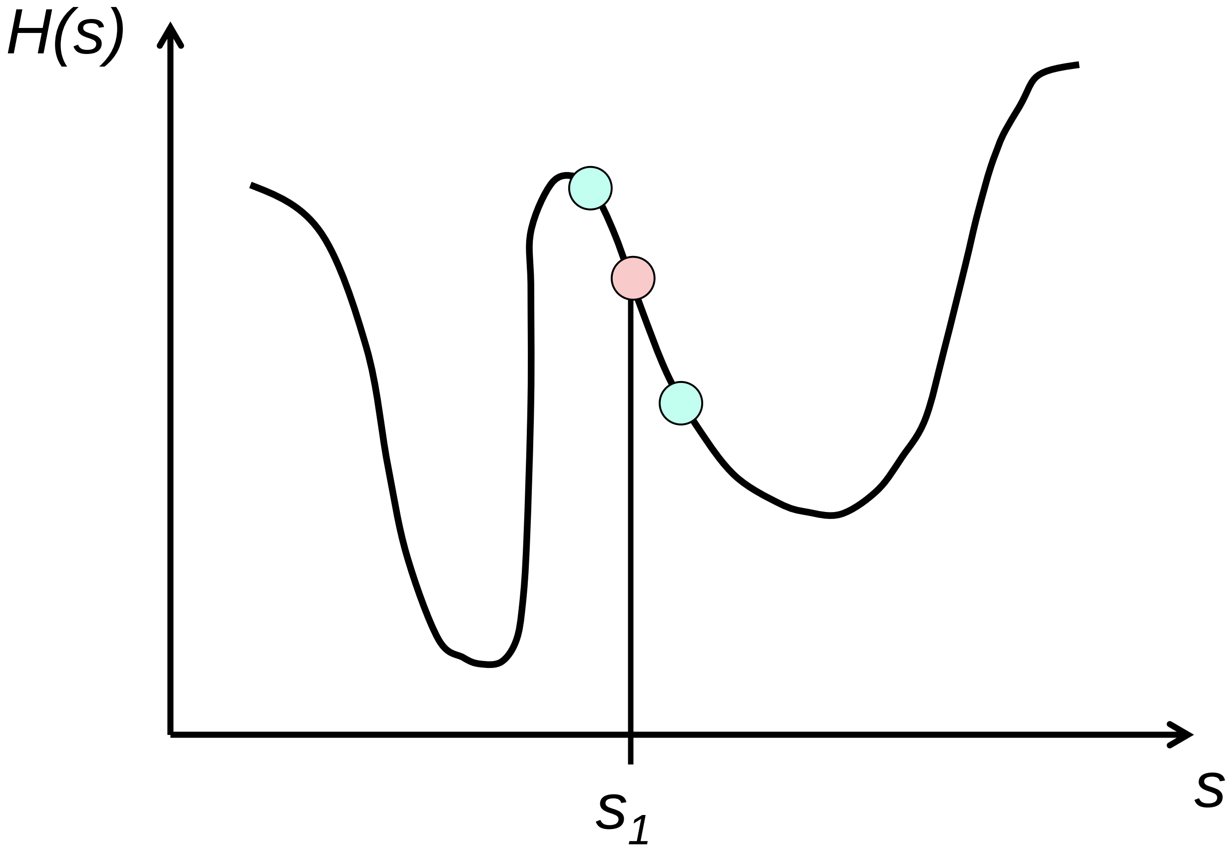
Local minimum: s=3

Global minimum: s=7

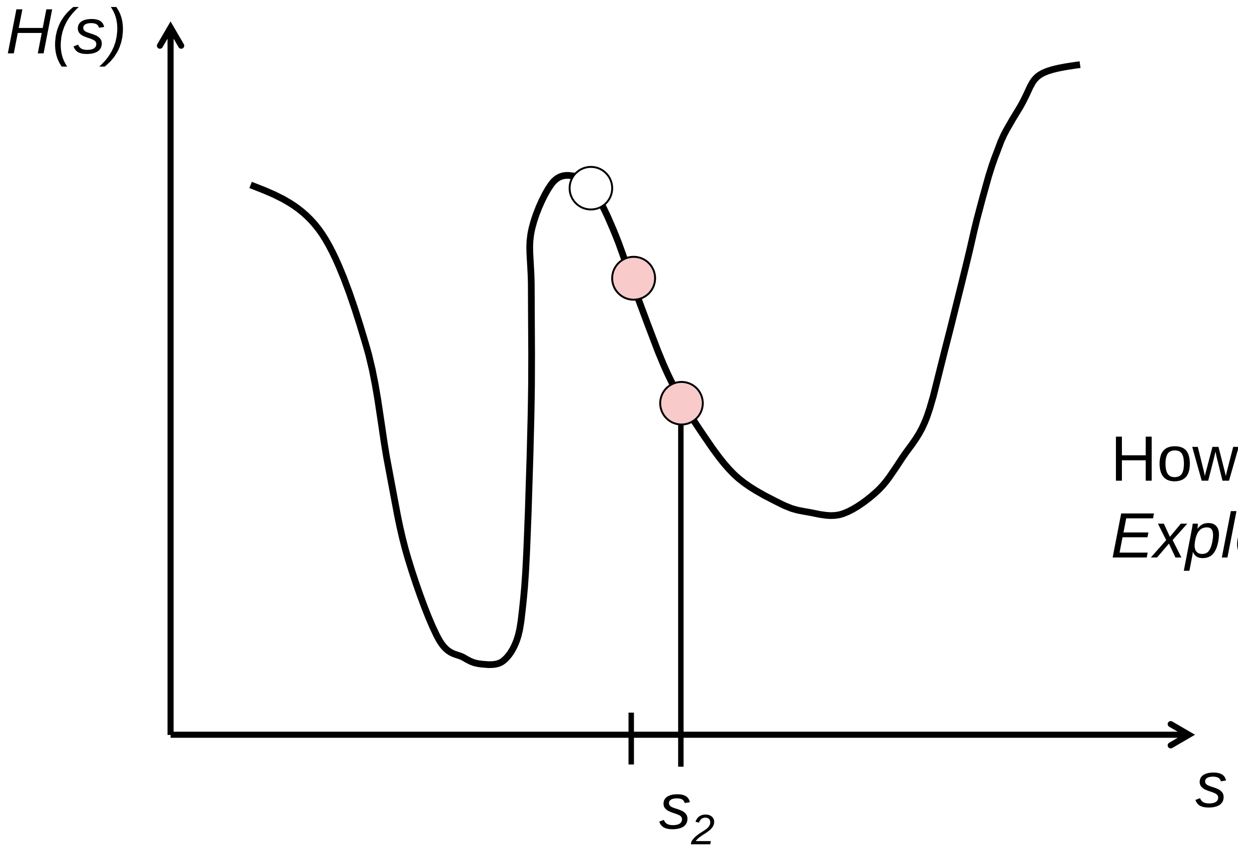
Local Search with Continuous States



Local Search with Continuous States

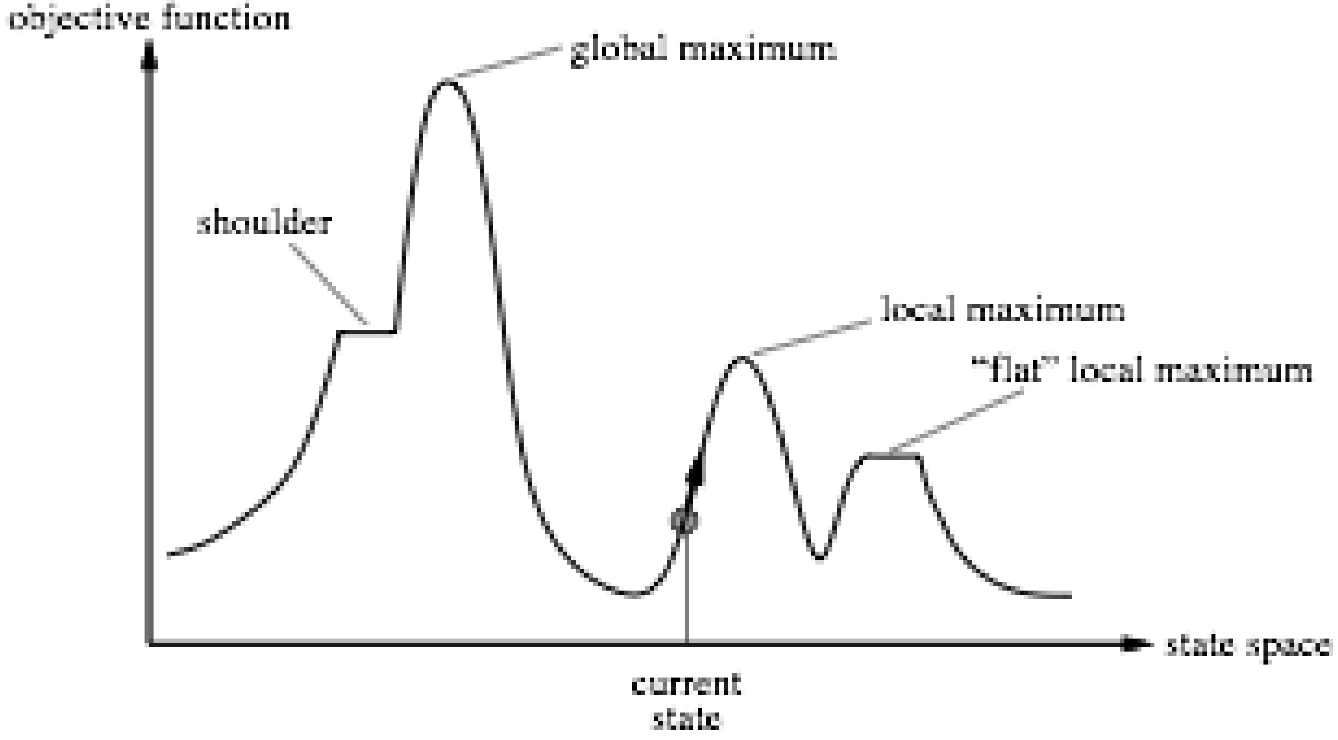


Local Search with Continuous States



How can we best identify the next state?
Exploit the gradient of the cost function

Issues with Local Minima



Summary

The goal of local search is to find a state which minimizes (or maximizes) a given objective function (state cost)

Search begins at a starting point and proceeds iteratively so as to improve the objective

Local extrema (minima or maxima) are states for which local search cannot improve the objective function

Questions?