Local Search

Russell and Norvig: Chapter 5

CSE 240: Winter 2023

Lecture 8

Announcements

- Go over feedback today (thank you!)
- Next week: Prof. Marinescu will lecture (AAAI)
- Check Slack before posting
  - (Try to keep threads with replies).
  - ~24 hour return

Agenda and Topics

- Local search and optimization algorithms.
  - Hill climbing
  - Stochastic Hill Climbing
  - Simulated Annealing
  - Local Beam Search

Hill Climbing Search

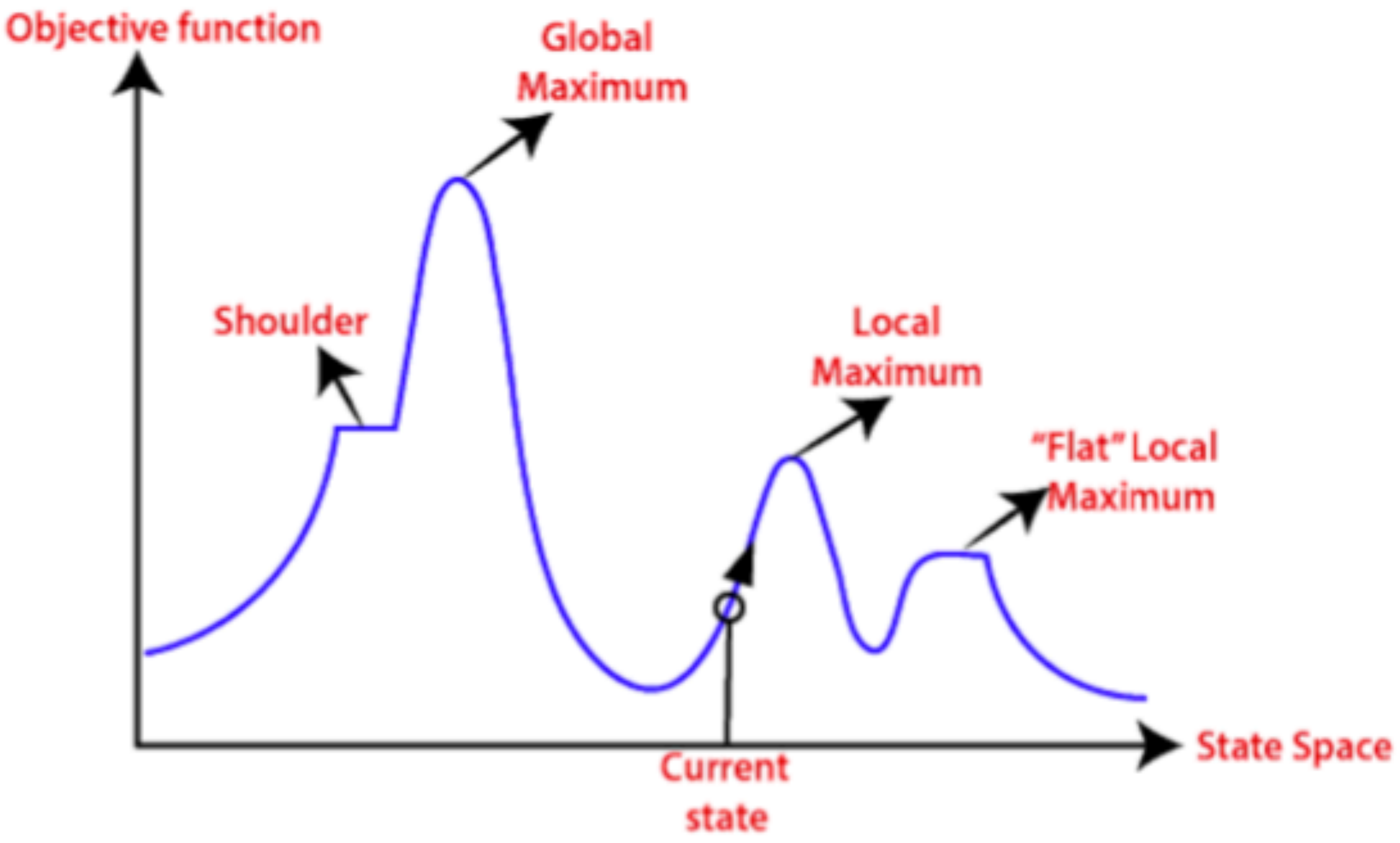
```
function HILL-CLIMBING(problem) returns a state that is a local maximum
   inputs: problem, a problem
   local variables: current, a node
                    neighbor, a node

   current ← MAKE-NODE(INITIAL-STATE[problem])
   loop do
      neighbor ← a highest-valued successor of current
      if VALUE[neighbor] ≤ VALUE[current] then return STATE[current]
      current ← neighbor
```

- Search strategy: steepest ascent among immediate neighbors until reaching a peak.
- “...like trying to find the top of Mount Everest in a thick fog while suffering from amnesia”

Hill Climbing Difficulties

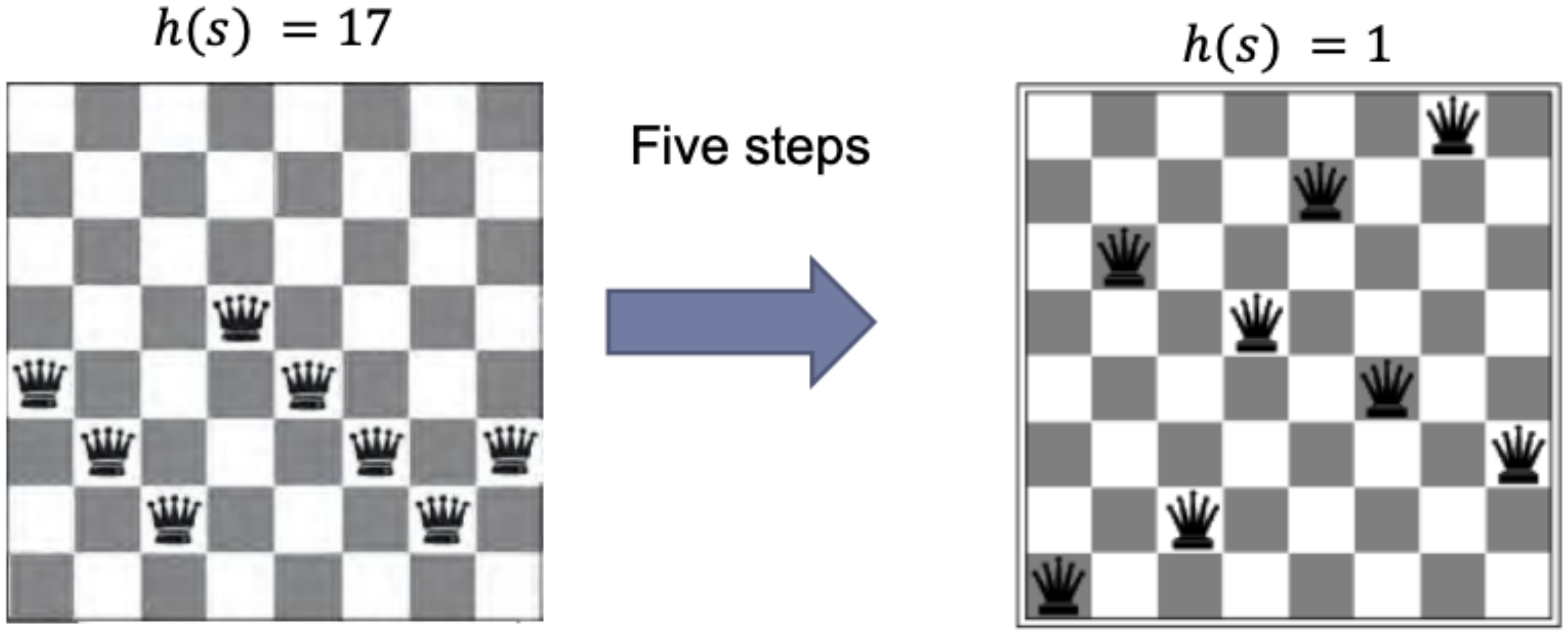
- Depending on initial state, algorithm can get stuck in local optimum
- Local maxima: a peak that's not a global maximum
- Plateau: a flat area (shoulders or flat local maximum).



A one-dimensional state-space landscape in which elevation corresponds to the objective function

Hill Climbing Search Problem: 8-Queen

- From a random initial state, the algorithm will get stuck in 86% of the cases
- On average, 4 steps for succeeding and 3 steps for failure

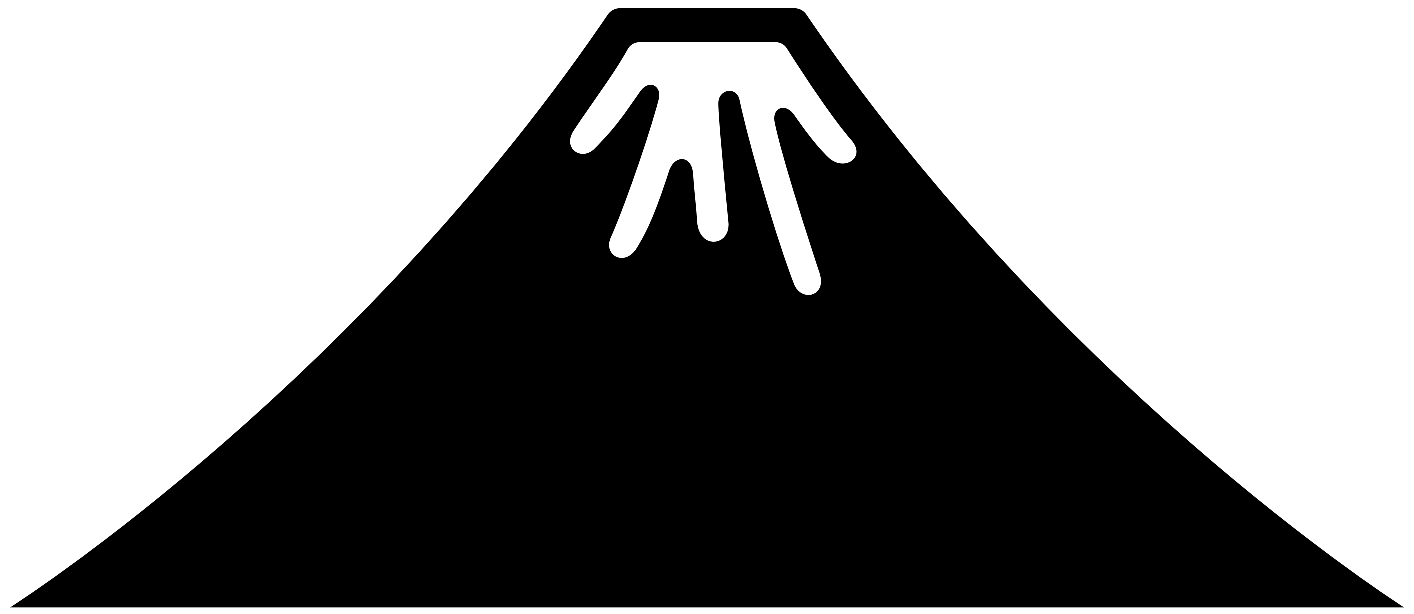


Sideways Move

- **Sideways move**: plateau may be a shoulder so keep going sideways move when there is no uphill move
- Problem: infinite loop for late local maximum
    - Solution: an upper-bound on the number of consecutive sideways moves
- Results on n-queen:
    - Limit 100 sideways moves
    - 94% success rate instead of 14%
    - On average, 21 steps when succeeding, and 64 steps when failing

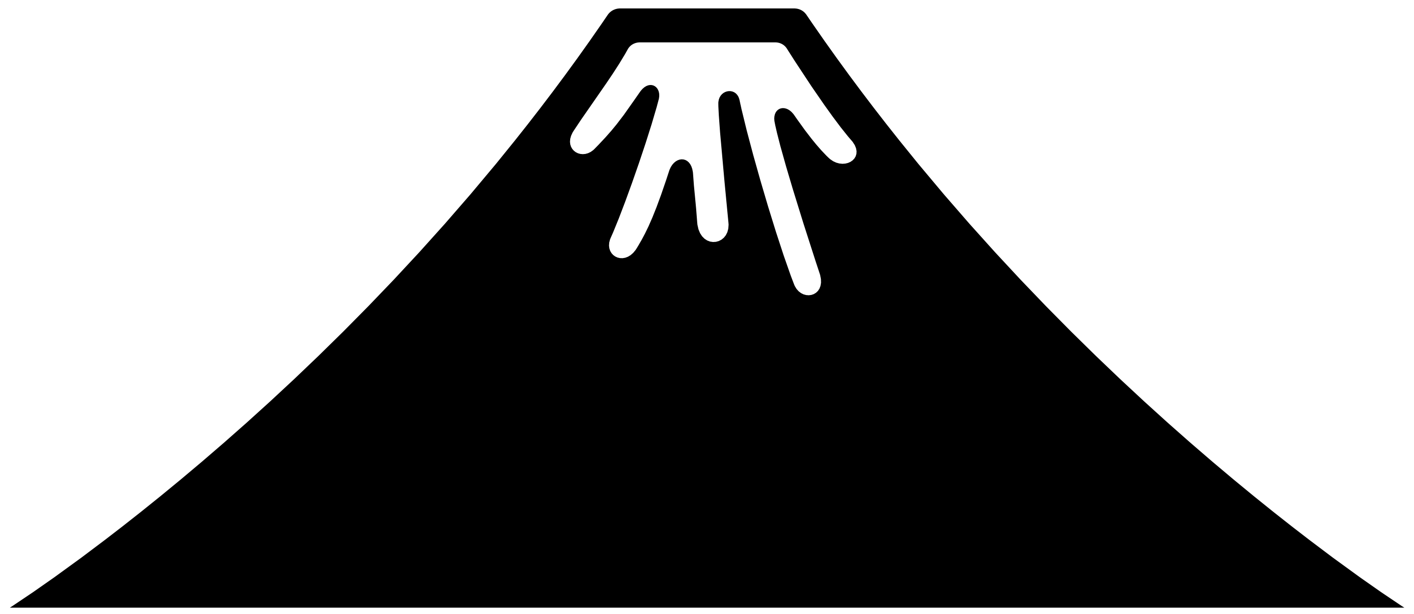
Variants of Hill-Climbing

- Trying to solve problem of hill- climbing search
  - Sideways moves
  - Stochastic hill-climbing
    - Or first-choice hill-climbing
  - Random restart hill-climbing



Stochastic Hill Climbing

- Randomly chooses among available uphill moves according to the steepness of these moves
- P(s') is an increasing function of h(s') – h(s)
- First-choice hill-climbing: generating successor randomly until one better than the current state is found
    - A good choice when the number of successors is high



Random Restart Hill-Climbing

- Adopts the idea of "if at first you don't succeed, try again."
- It conducts a series of hill-climbing searches from randomly generated initial states until a goal is found.
- The algorithm is asymptotically complete with probability approaching 1, because it will eventually generate a goal state as the initial state.

```
while state != goal
     run hill-climbing search from a random initial state
```

Random Restart Hill-climbing

- Assuming that p is the probability of success
- Then the expected number of required restarts is $\frac{1}{p}$
- Expected number of steps = $n_s + n_f\left(\frac{1}{p} - 1\right)$
  - n_s: average number of steps for success
  - n_f: average number of steps for failure
- Results on 8-queen:
  - p = 14%, $\frac{1}{p}$ = 7 (iterations)→ # of steps = 6$*$3+4=22 steps
  - Using sideways moves: p = 94%, $\frac{1}{p}$ = 1.06 (iterations) → # of steps = (1.06 - 1) * 64 + 21 = 26 steps
  - Sideways moves will require similar number of steps overall, but it doesn't need restarting

Restarts	0	2	4	8	16	32	64
Success?	14%	36%	53%	74%	92%	99%	99.994%

CE 8: Pros and Cons

- Compare and contrast the hill-climbing augmentation strategies we've seen so far:
  - Randomly chooses among available uphill moves according to the steepness of these moves
  - P(s') is an increasing function of h(s') – h(s)
  - First-choice hill-climbing: generating successor randomly until one better than the current state is found
    - A good choice when the number of successors is high
  - Random restart

Effect Of Landscape Shape On Hill-climbing Algorithm

Shape of the landscape is important

Few local maximum and plateau areas, then random restart hill-climbing is fast

Landscape of real problems is usual an unknown

NP-Hard problems typically have exponential number of local maxima

Simulated Annealing Search

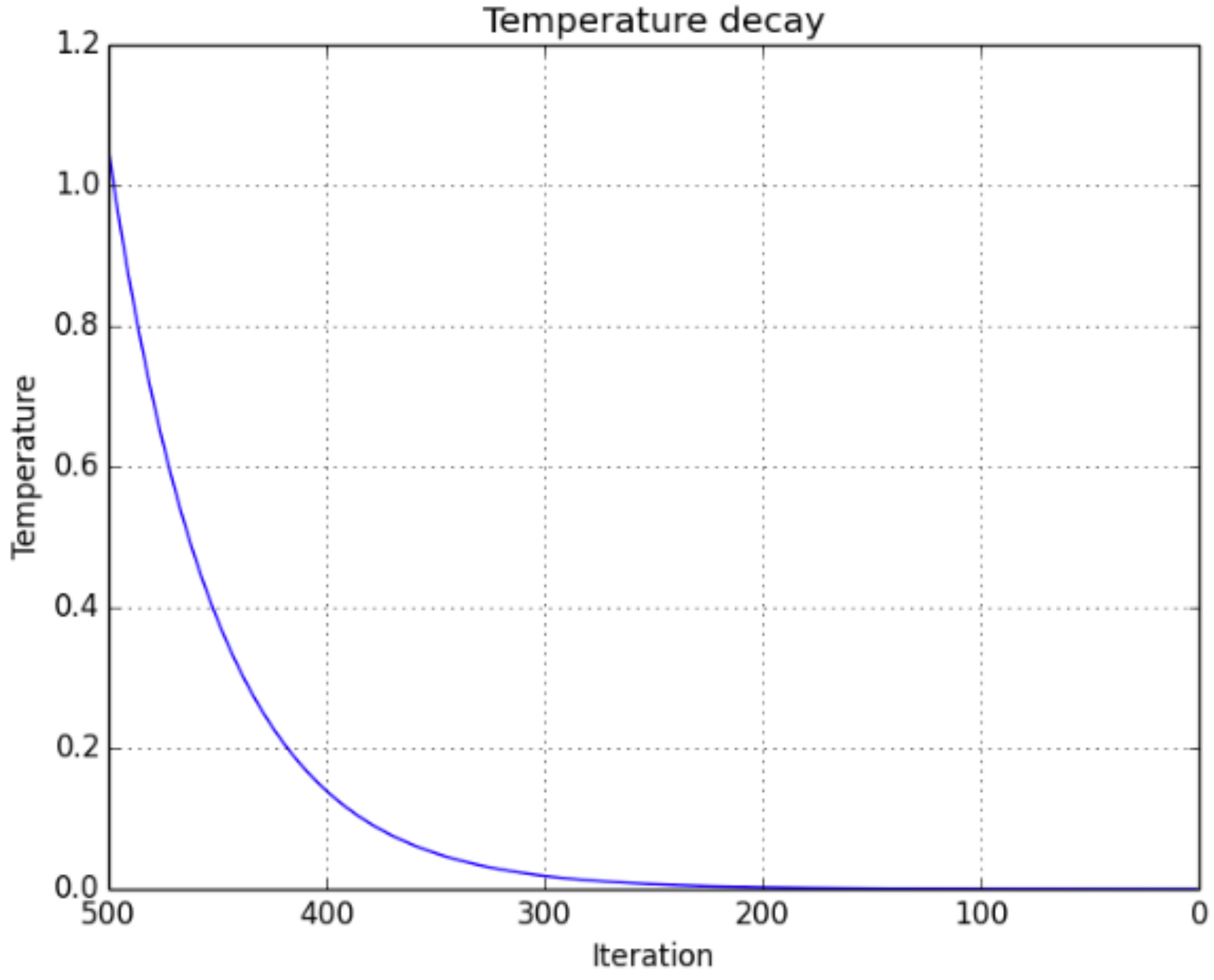
Idea: escape local maxima by allowing some "bad" moves but gradually decrease their frequency

```
function SIMULATED-ANNEALING( problem, schedule) returns a solution state
   inputs: problem, a problem
           schedule, a mapping from time to "temperature"
   local variables: current, a node
                    next, a node
                    T, a "temperature" controlling prob. of downward steps

   current ← MAKE-NODE(INITIAL-STATE[problem])
   for t ← 1 to ∞ do
       T ← schedule[t]
       if T = 0 then return current
       next ← a randomly selected successor of current
       ΔE ← VALUE[next] – VALUE[current]
       if ΔE > 0 then current ← next
       else current ← next only with probability e^(Δ E/T)
```

Typical Annealing Schedule

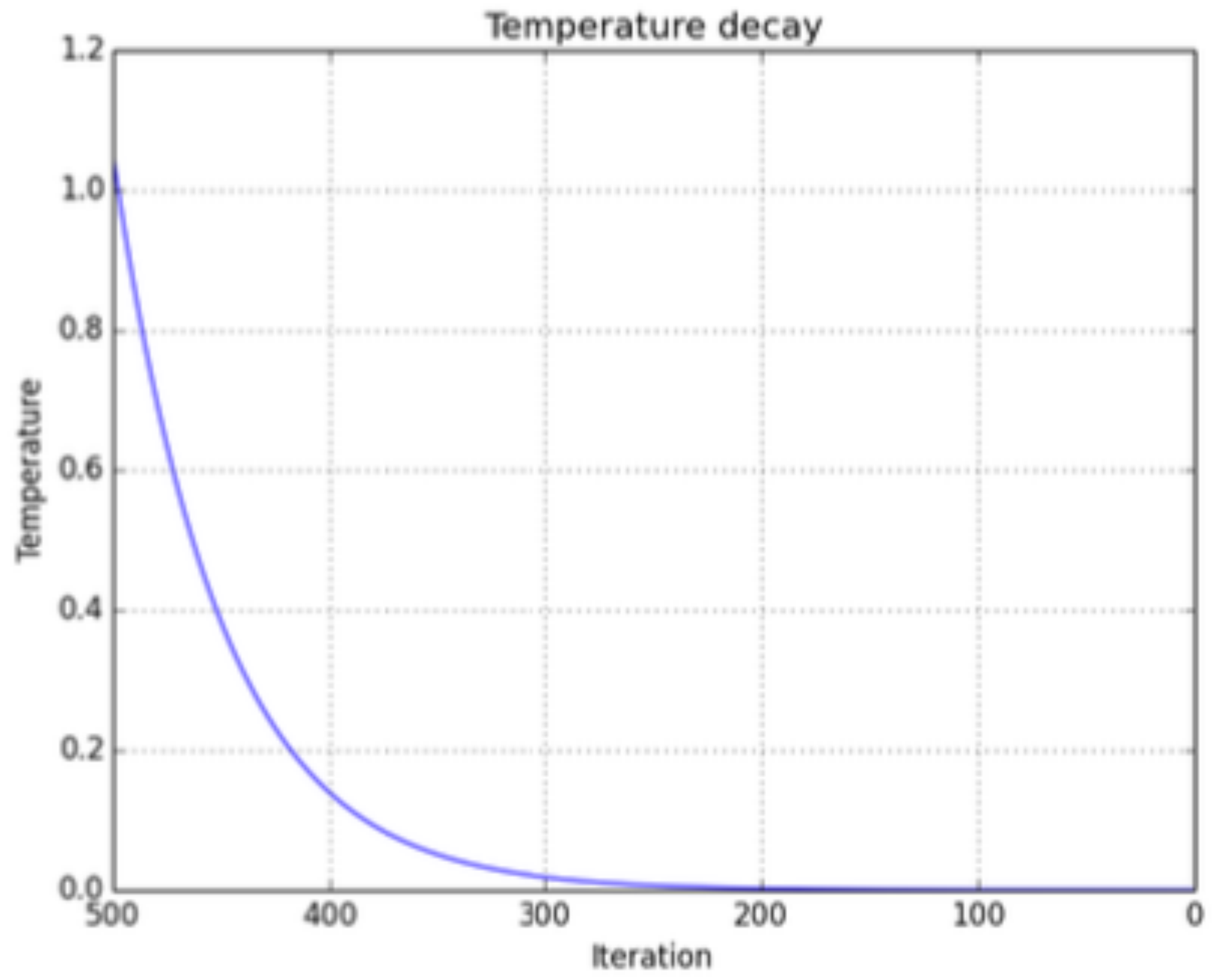
- Usually use a decaying exponential
- Temperature schedule should be decided based on problem characteristics



P(accepting a worse successor)

- Decreases as temperature T decreases
  - Accept bad moves early on
- Increases as $|\Delta E|$ decreases
  - Accept not much worse

$e^{\Delta E / T}$		Temperature T	
		High	**Low**
$\|\Delta E\|$	**High**	Medium	Low
	Low	High	Medium



next ← a randomly selected successor of *current*
$\Delta E \leftarrow$ VALUE[*next*] – VALUE[*current*]
if $\Delta E > 0$ **then** *current* ← *next*
else *current* ← *next* only with probability $\boxed{e^{\Delta E/T}}$

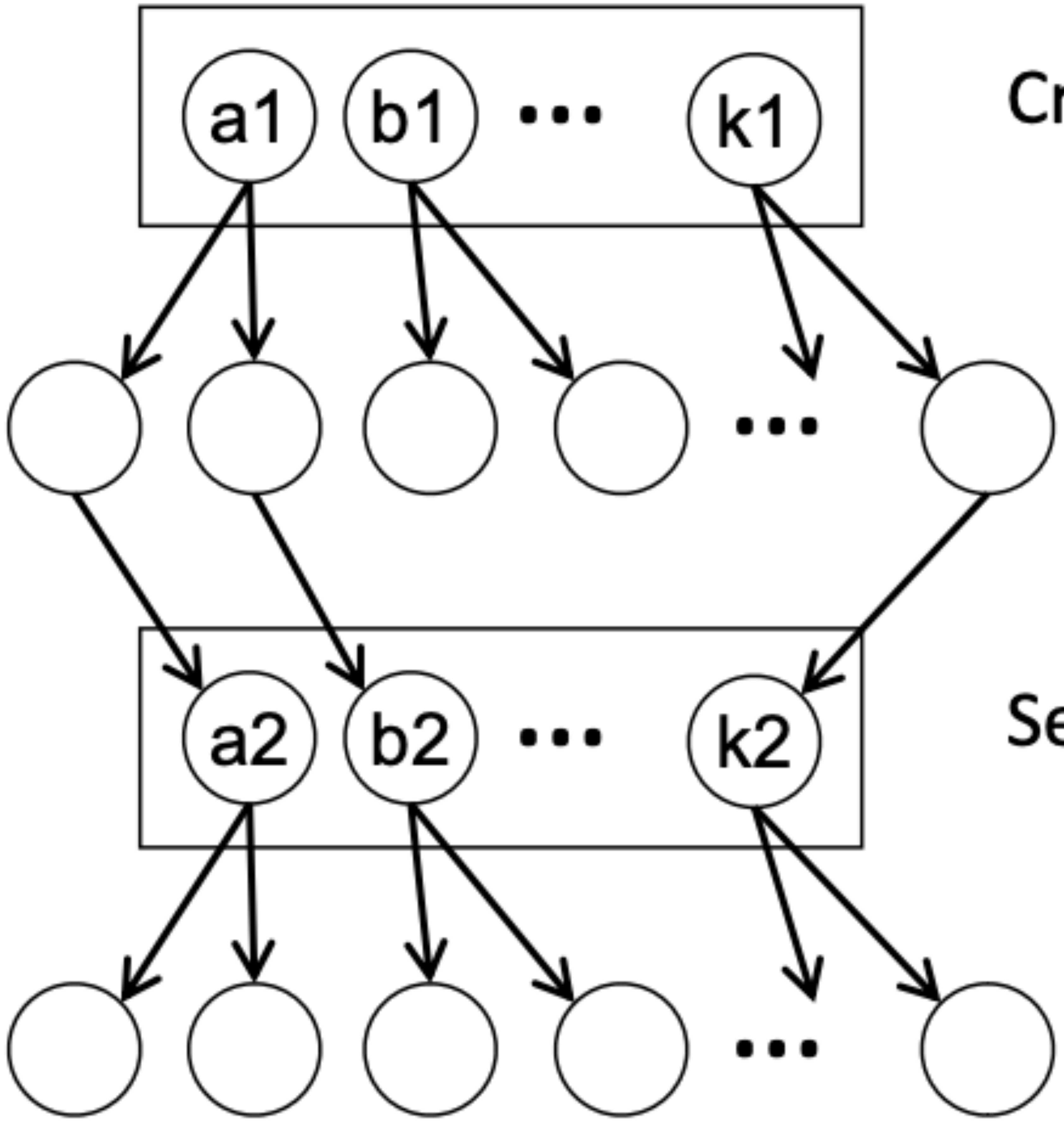
Properties of Simulated Annealing

- One can prove:
  - If T decreases slowly enough, then simulated annealing search will find a global optimum with probability approaching 1
  - Unfortunately this can take a VERY VERY long time
  - Note: in any finite search space, <u>random walk</u> also will find a global optimum with probability approaching 1 (asymptotically complete algorithm). So, ultimately this is a very weak claim
- Often works very well in practice
  - But usually VERY VERY slow
- Widely used in VLSI layout, airline scheduling, etc.

Local Beam Search

- Keep track of *k* states rather than just one
- Start with *k* randomly generated states
- At each iteration, all the successors of all *k* states are generated
- If any one is a goal state, stop; else select the *k* best successors from the complete list and repeat.

Local Beam Search



Create *k* random initial states

Generate their children

Select the *k* best children

Repeat indefinitely...

Is it better than simply running *k* searches?

Local Beam Search Problem

- Concentrates search effort in areas believed to be fruitful
  - May lose diversity as search progresses, resulting in wasted effort
- Solution:
  - **Stochastic beam search**
  - Choose k successors at random with probability that is an increasing function of their objective value

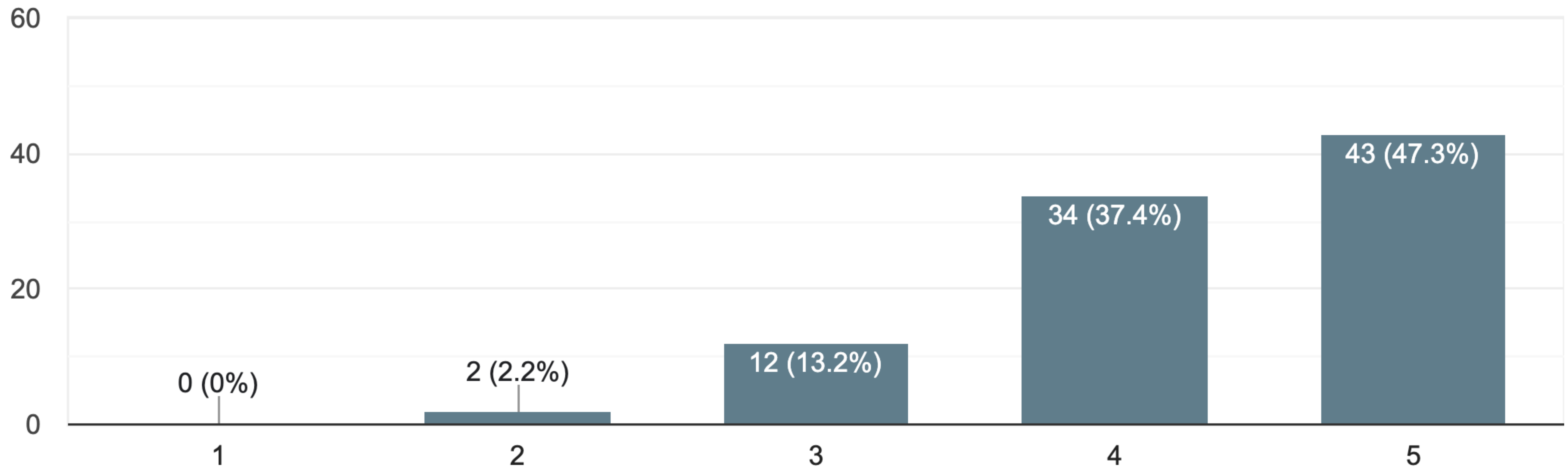
Survey

First of all; thanks!

- The teaching team and I are very open to feedback
- We appreciate you taking the time to help make the class better!
- Disclaimer
  - Large graduate class.
  - Different abilities

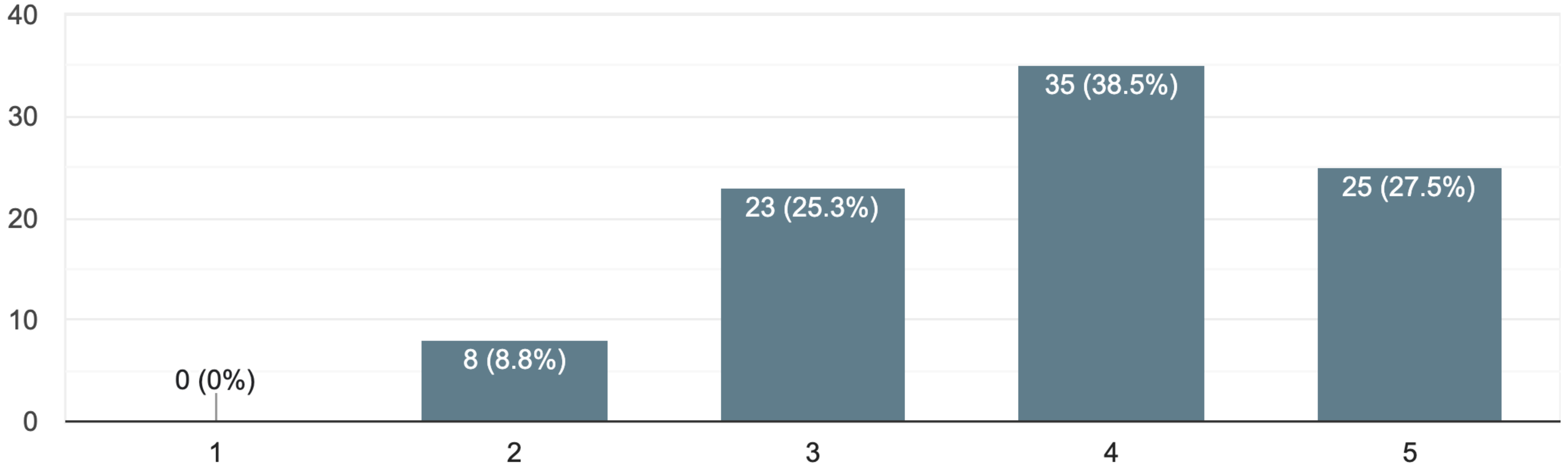
1. Choose the appropriate representation for an AI problem or domain model (e.g., BFS versus DFS)

91 responses



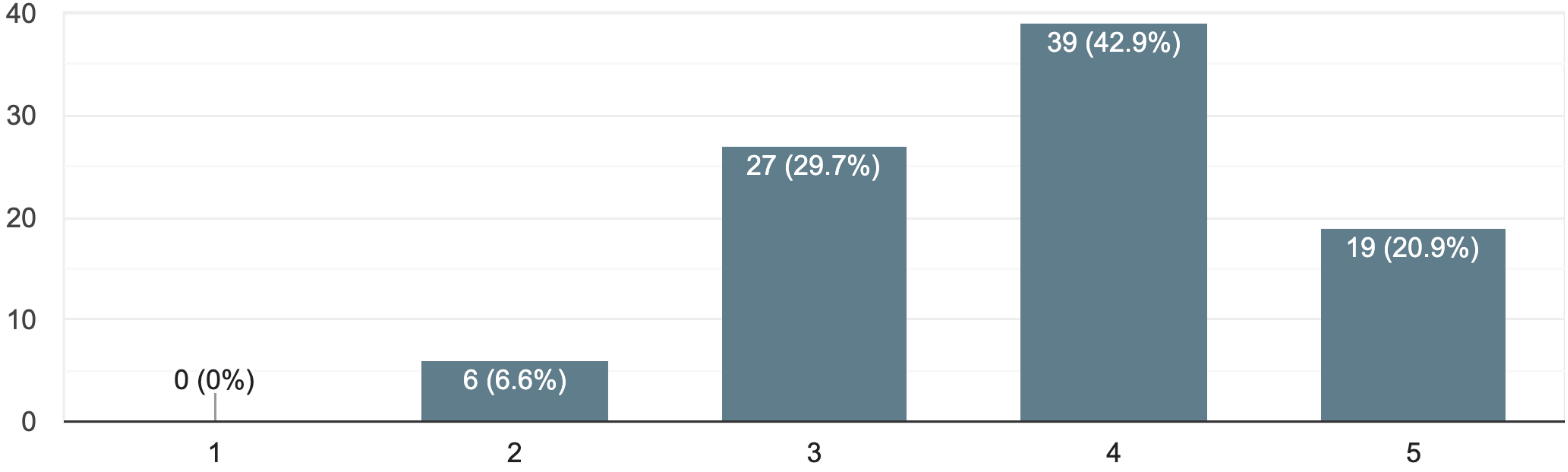
2. Implement and debug core AI algorithms in a clean and structured manner (e.g., alpha-beta pruning, A*, etc.)

91 responses



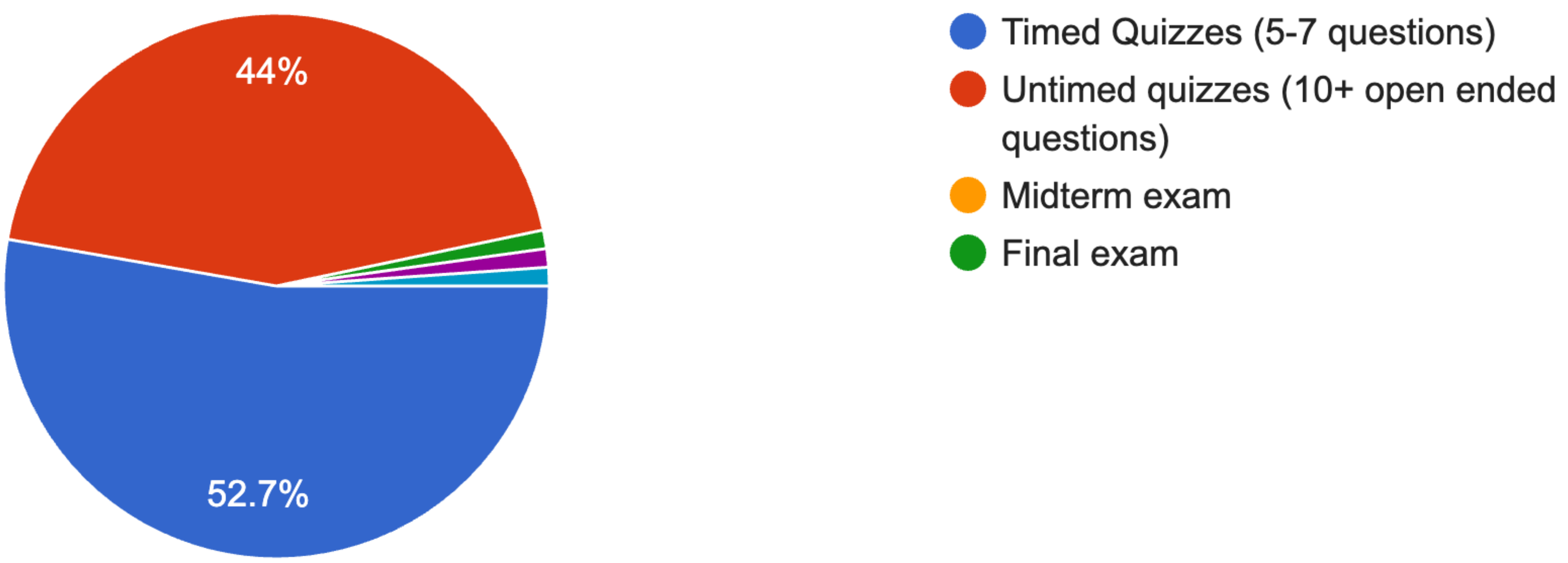
3. Design and analyze the performance of an AI system or component

91 responses



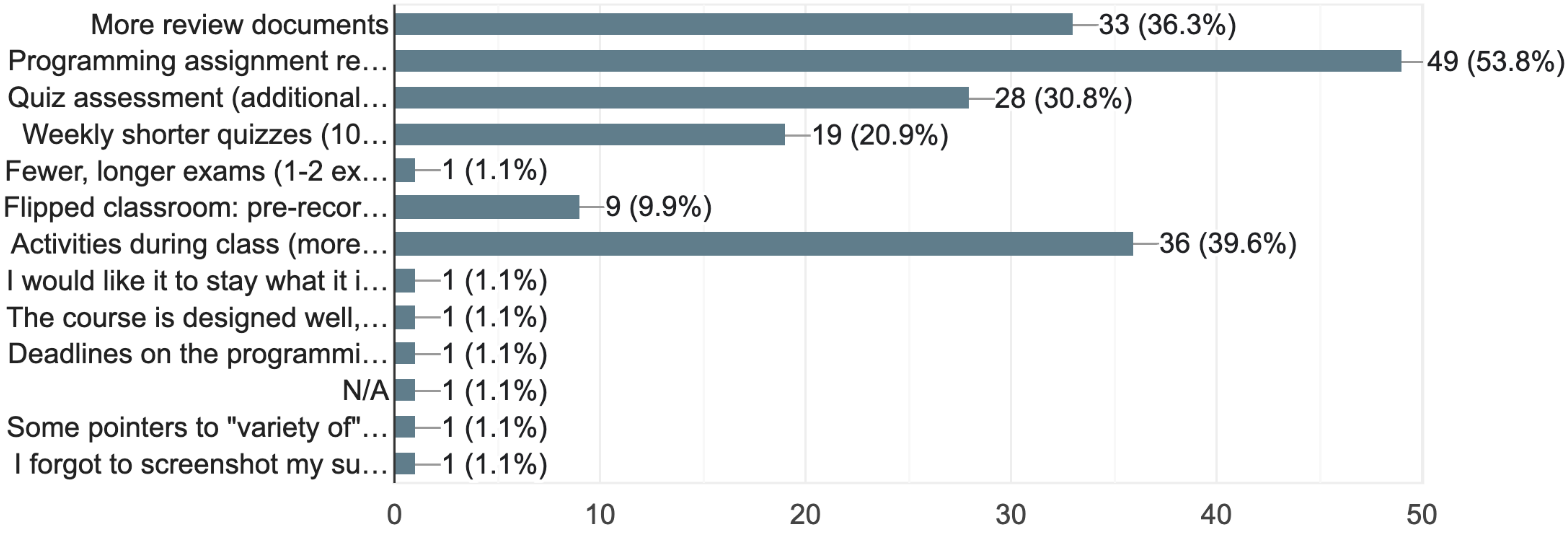
4. What modality of examination would be most supportive to test your conceptual knowledge of core concepts?

91 responses



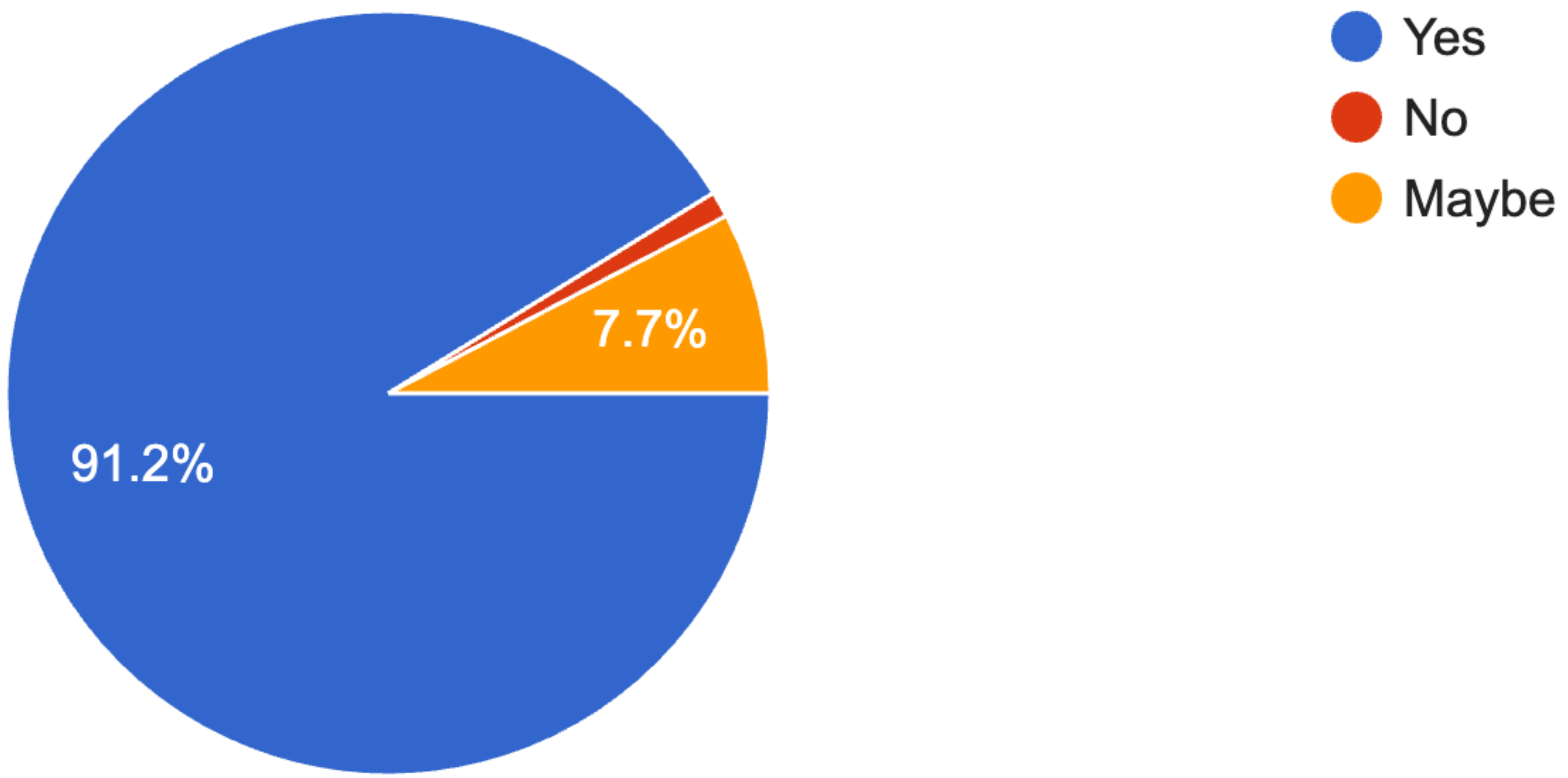
6. What changes could be made in the first weeks of the course to support learning? (Choose up to 2):

91 responses



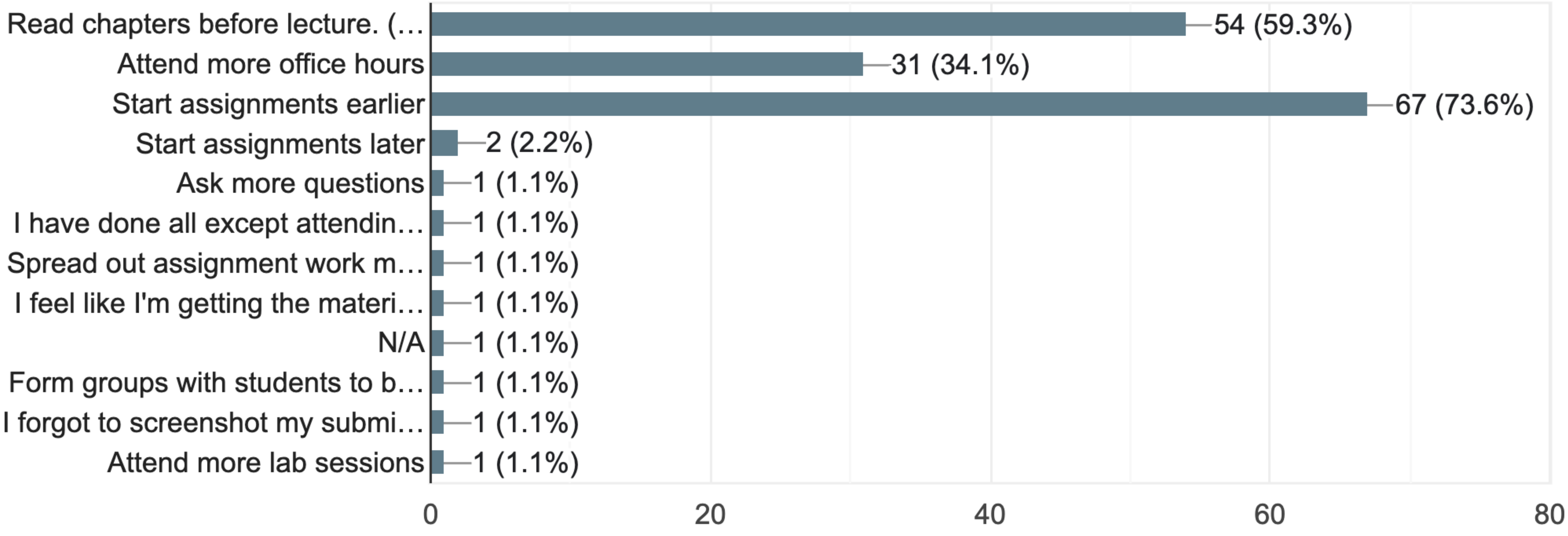
8. Do you feel that you have enough support and accessibility from the teaching team?

91 responses



10. To improve your learning in this course in the second half of the term, how might you modify your approach to studying, working on the course materials, or managing your time?

91 responses



Changes

- Discuss quiz modality with the teaching team.
- More advanced material and examples
- Try to save the annotations on slides

Summary

- Local search and optimization algorithms
  - Hill climbing
  - Stochastic Hill Climbing
  - Simulated Annealing
  - Local Beam Search

Next time:

- Genetic Algorithms
- Constraint Satisfaction Problems (CSP)