Tufts UNIVERSITY

LOCAL SEARCH

ARTIFICIAL INTELLIGENCE | COMP 131

TODAY ON AI

- Hill climbing
- Simulated annealing
- Genetic algorithms
- Questions?

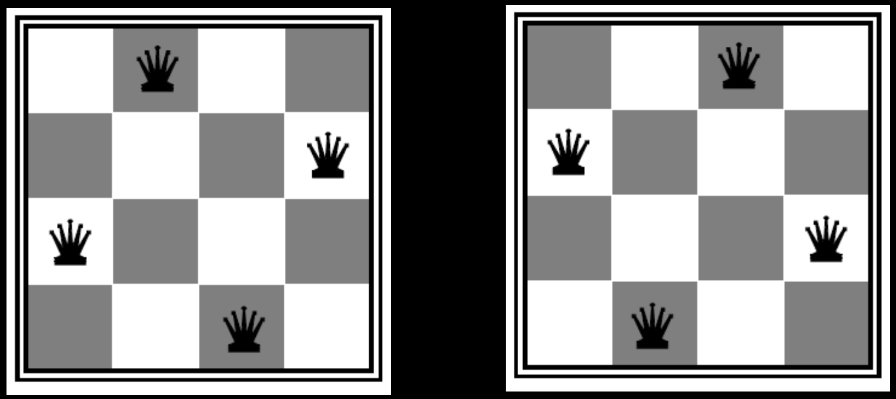
Constraint satisfaction problems (or **CSPs**) belong to a class of problems for which the goal itself is the most important part, not the path used to reach it.

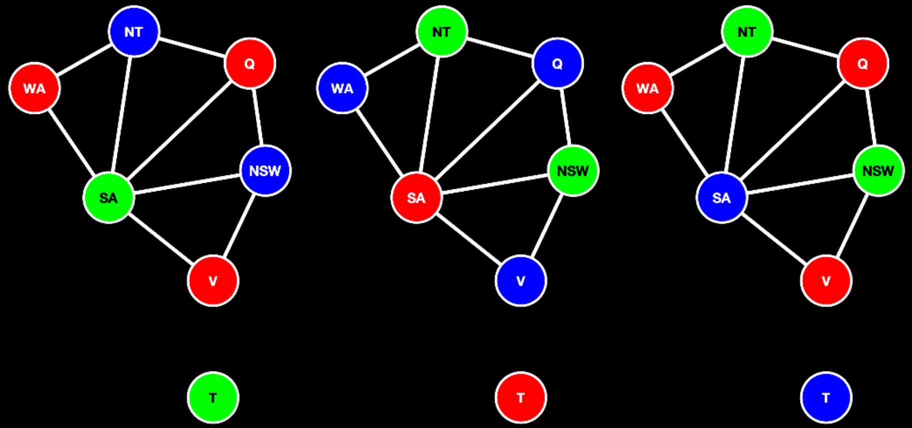
EXAMPLES

- Map coloring!
- Sudokus
- Crossword puzzles
- Job scheduling
- Cryptarithmetic puzzles
- N-Queens problems
- Hardware configuration
- Assignment problems
- Transportation scheduling
- Fault diagnosis
- More…

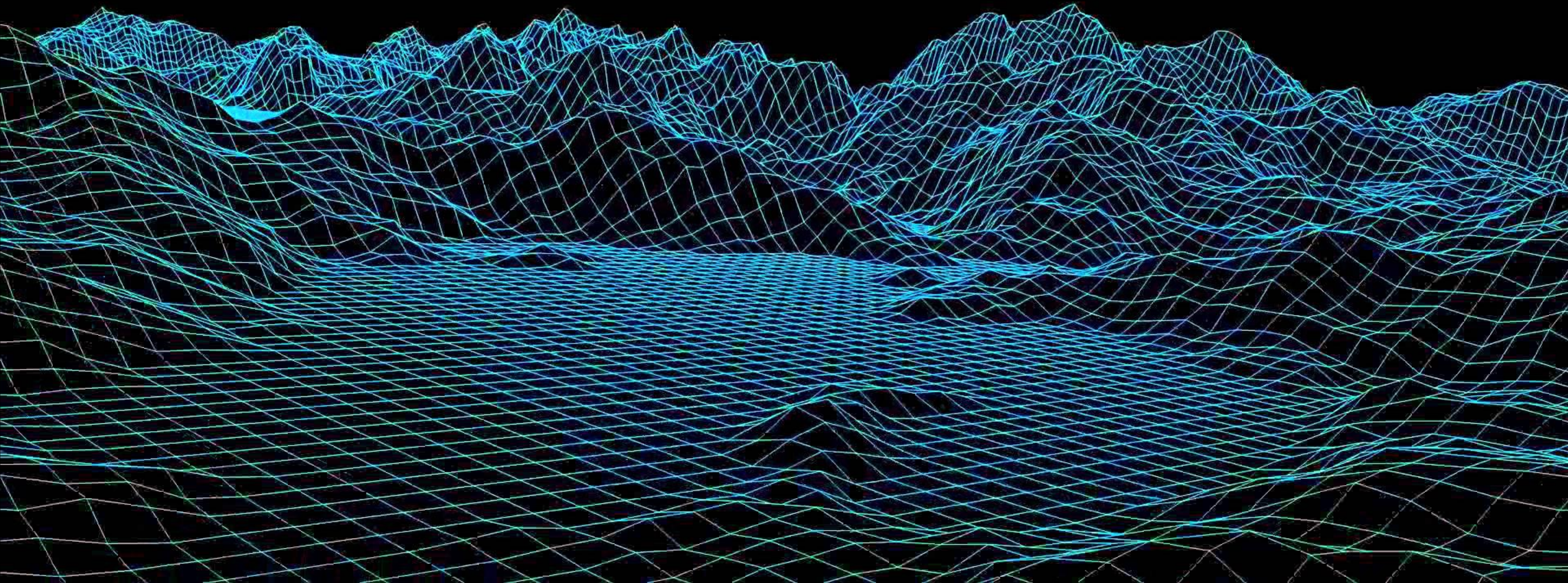
Local search algorithms aim to solve some Constraint Satisfaction Problems more efficiently. Specifically, they are tailored to find a solution to problems whose **search space is very big** or **infinite** without returning the actual path to the solution

They can **always** provide an answer to the problem, even if it is both not definitive nor correct.





Local search algorithms work best when a function that measure the **fitness** of the solution can be defined and the **fitness landscape** is continuous.



- This class of algorithms operate on **single current nodes** that represent the complete state of the search
- The **current state** is the only thing that matter
- The state is evaluated with an **objective function**
- They **generally tend to move** through neighborhoods

GOOD Generally much faster for large or infinite state space. More memory efficient

BAD Incomplete and suboptimal. Not systematic search

SECTION 02

Hill climbing algorithms

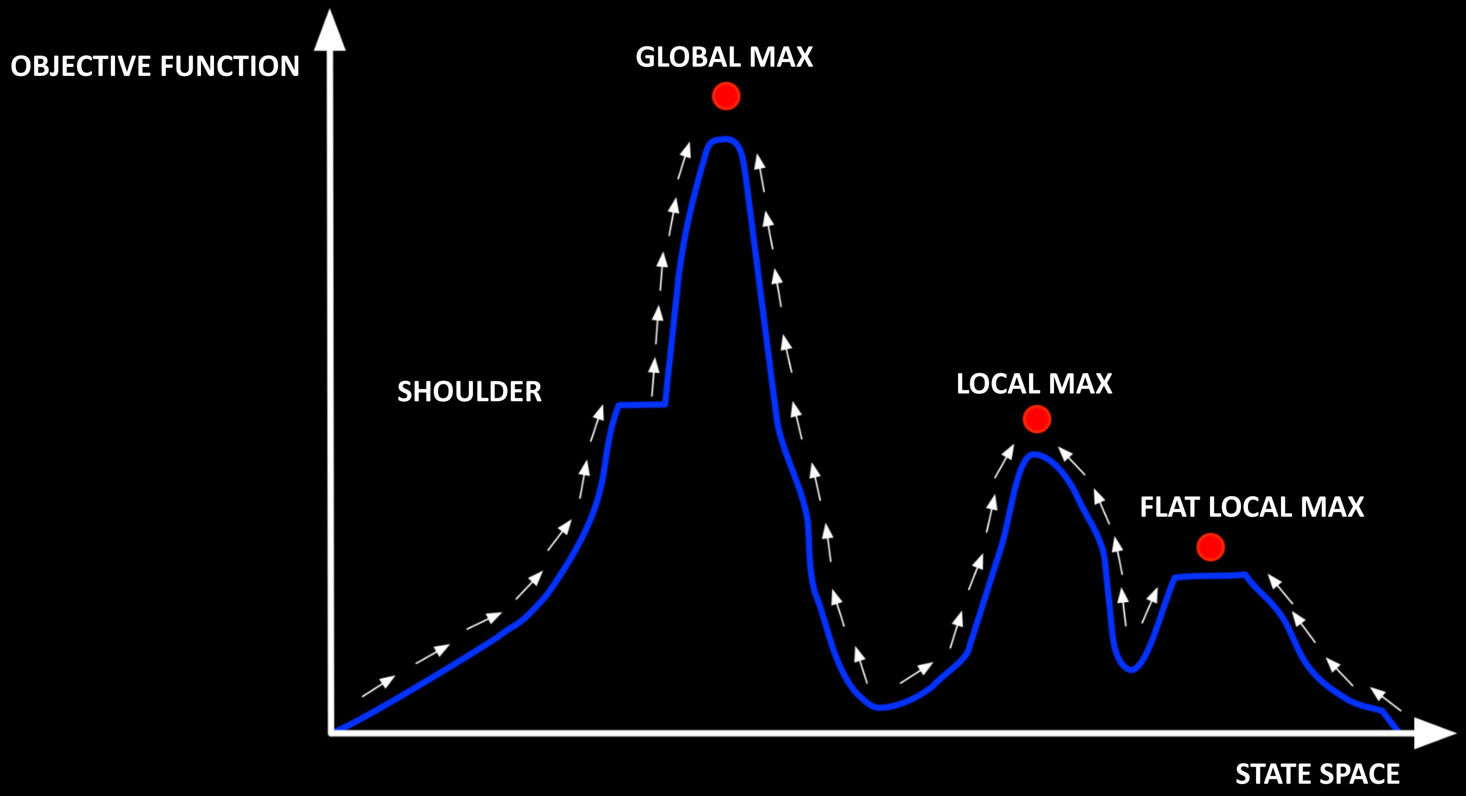
Hill climbing algorithm is the **most basic local search** technique. It is **greedy** in nature. At each step, the current node is **replaced** by the best neighbor.

```
function Hill-climbing(PROBLEM) return SOLUTION, or FAILURE

current = Make-node(PROBLEM initial state)

loop do
  neighbor = a highest-valued successor of current
  if neighbor.value ≤ current value then
    return current state
  current = neighbor
```

GOOD Very simple to implement

BAD It can easily get stack in local maxima, ridges and plateau

- **Randomly choose** to initialize several times
- Implement hill climbing for **each initialization** and find the optimal
- If each hill climbing search has a **probability p of success**, then the expected number of restarts required is $1/p$.

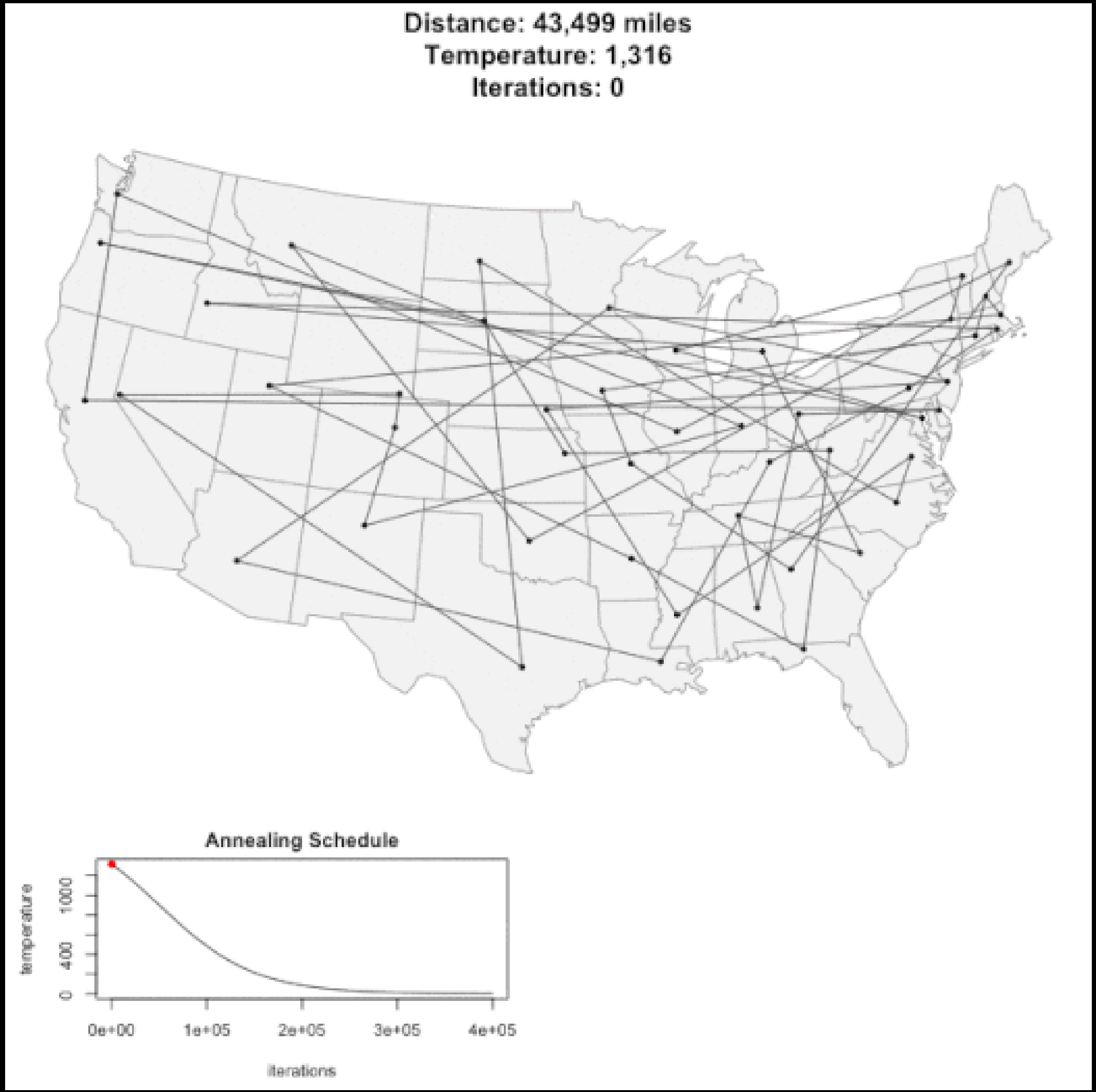


SECTION 02

Simulated annealing

Simulated annealing is a class of algorithms that is inspired by statistical physics:

- **Annealing** is used in metal forging and glass making to aid the formation of crystal structures in the material
- The process **slowly reduces the temperature** the material to allow initial more random arrangements of atoms. At **lower temperatures** the crystallin structure is more stable



The **Traveling Salesman Problem** is a mathematical problem, formulated by W. R. Hamilton in the 1800s, in which one has to find which is the **shortest route** which passes through each of a set of points once and only once.

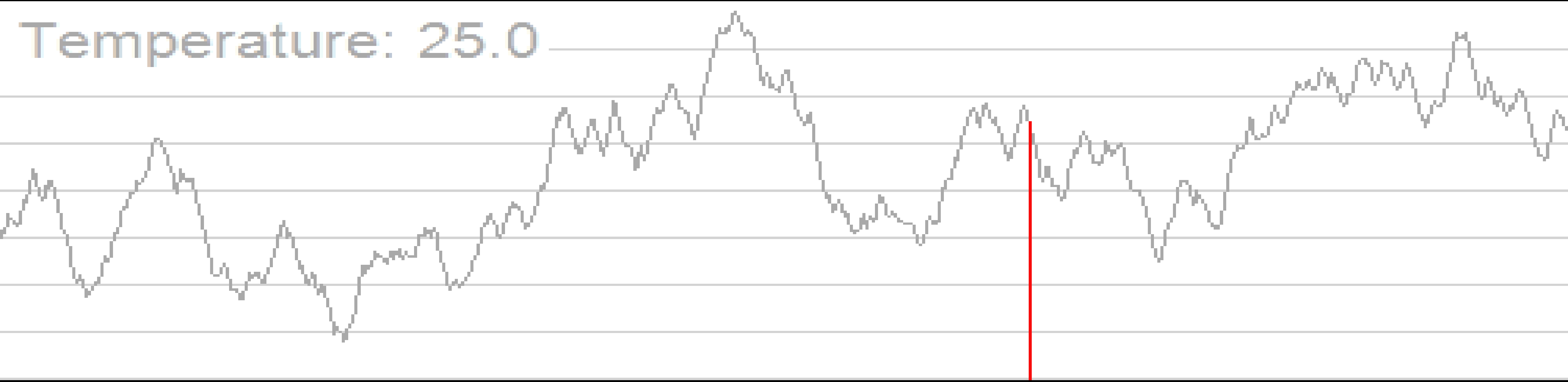
The Traveling Salesman Problem

Animation by shiny.rstudio.com

- The basic idea follows the **annealing physical metaphor**: select random successors with decreasing probability, also known as **temperature**.

- A gradient ΔE is calculated:
  - If $\Delta E > 0$ the new state is **accepted immediately** as an improvement
  - If $\Delta E < 0$ the new state is **accepted only with a probability** that depends on ΔE and T

- If T decreases slowly enough, the algorithm will converge




```
function Simulated-annealing(PROBLEM, SCHEDULE) return SOLUTION, or FAILURE
current = Make-node(PROBLEM initial state)
loop do
  T = SCHEDULE(t)
  if T = 0 then
    return current state
  next = a randomly selected successor of current
  ΔE = next value - current value
  if ΔE > 0 then
    current = next
  else
    current = next only with probability e^(ΔE/T)
```

$$e^{\frac{\Delta E}{T}}$$

Simulated annealing

Animation by Kingpin13, Wikimedia Commons.

SECTION 02

Genetic algorithm

Genetic algorithms are a randomized heuristic search strategy:

- They use a **natural selection metaphor** to find the best solution to a problem
- The selection process is applied to a population that is composed of **candidate solutions**
- The purpose is to **evolve a population** from which strong and diverse candidates can emerge via mutation and crossover, also known as mating

- An hypothesis is described by a **chromosome**
- Few successor functions are needed (also known as **fringe functions**):
  - Mutation
  - crossover
- A **fitness function** is used to implement a natural selection process
- A **solution test** is required if different from the fitness function
- Some general parameters guide the evolution of the population:
  - **Population size**
  - **Generation limit**

1. Start with a random population
2. Apply a **fitness function** to recognize the fittest individuals
3. Keep N hypotheses at each step that have a high value of a fitness function
4. Possibly **cull** the less fit individuals and remove them
5. Apply one or more **successor operations** to generate a new population
6. Apply the solution test to the best candidate
7. Start over

A **successor operation** changes the current state of the search into something new:

- A **Mutation** fringe operation: given a candidate, return a slightly different candidate
- A **Crossover** fringe operation: given two candidates, produce one that has elements of each

We don’t always generate a successor for each individual. Rather, we generate a successor **population** based on the individuals in the current population, weighted by fitness.

A new population can be generated by:

- Given a population P, generate P' by performing crossover $|P|$ times, each time selecting candidates with probability **proportional to their fitness**
- Get P'' by mutating each individual in P'
- Return P''

The previous approach doesn't explicitly allow individuals to survive more than one generation.

Crossover is **not necessary**, though it can be helpful in escaping local maxima. Mutation is **fundamental**.

GOOD

- Faster and with lower memory requirements
- It can explore a very large search space
- Easy to design

BAD

- Randomized – not optimal or even complete
- Can get stuck on local maxima, though mutation can help mitigate this
- It can be hard to design a chromosome

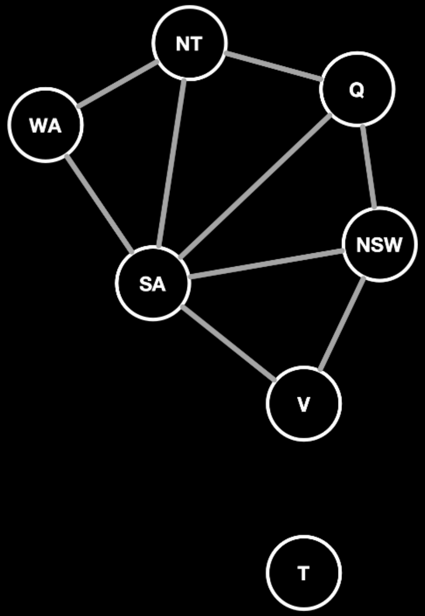
```
function GENETIC-ALGORITHM(population, FITNESS-FN) return an individual
  repeat
    new population ← empty set
    for i = 1 to SIZE(population) do
      x ← RANDOM-SELECTION(population, FITNESS-FN)
      y ← RANDOM-SELECTION(population, FITNESS-FN)
      child ← REPRODUCE(x, y)
      if (small random probability) then child ← MUTATE(child)
        add child to new population
        population ← new population
    until some individual is fit enough, or enough time has elapsed
    return the best individual in population, according to FITNESS-FN

function REPRODUCE(x, y) return an individual
  n ← LENGTH(x)
  c ← random number from 1 to n
  return APPEND SUBSTRING(x, 1, c), SUBSTRING(y, c + 1, n))
```

- **STATES**
  Color the Australia map so that neighboring regions do not match

- **CHROMOSOME**

WA	NT	Q	NSW	V	SA	T

- **FITNESS FUNCTION**
  Number of pairs of regions that do not violate the constraint (max value 10)



#	INDIVIDUAL							FITNES FUNCTION	NORMALIZED FITNES FUNCTION
1	R	G	R	B	R	B	B	9	14%
2	R	R	R	G	G	B	B	7	11%
3	G	G	R	B	B	R	R	7	11%
4	R	G	B	B	R	B	G	7	11%
5	G	G	R	R	R	B	B	7	11%
6	G	B	G	B	R	B	B	8	13%
7	B	G	B	G	R	R	R	9	15%
8	G	B	B	G	B	R	B	9	14%
								63	100%

	POPULATION	SELECTION			CROSS-OVER		MUTATION	
1	R G R B R B B	9 14%	7	B G B G R R R	1	B G B G B R B	1	B G B G B R B
2	R R R G G B B	7 11%	8	G B B G B R B	2	G B B G R R R	2	G B R G R R R
3	G G R B B R R	7 11%	1	R G R B R B B	3	R G G B R B B	3	R G G B R B B
4	R G B B R B G	7 11%	6	G B G B R B B	4	G B R B R B B	4	G B R B R B B
5	G G R R R B B	7 11%	2	R R R G G B B	5	G G B G R R R	5	G G B G G R R
6	G B G B R B B	8 13%	3	G G R B B R R	8	B B G B R B B	8	B B G B R B B
7	B G B G R R R	9 15%	4	R G B B R B G	7	G B G B R B B	7	G B G B R B B
8	G B B G B R B	9 14%	5	G G R R R B B	6	R G R G B R B	6	R G R B B R B

If **culling** is applied, the least fit individuals are eliminated

READINGS

Chapters 4.1 – 4.6

Chapter 5

QUESTIONS ?

ARTIFICIAL INTELLIGENCE
COMP 131

FABRIZIO SANTINI

VERSION 3.0