

Discrete Optimization

Local Search: Part VI

Goals of the lecture

- ▶ Escaping local minima
- ▶ Connectivity

If you want guarantees, buy a toaster (C. Eastwood)

► Local minima

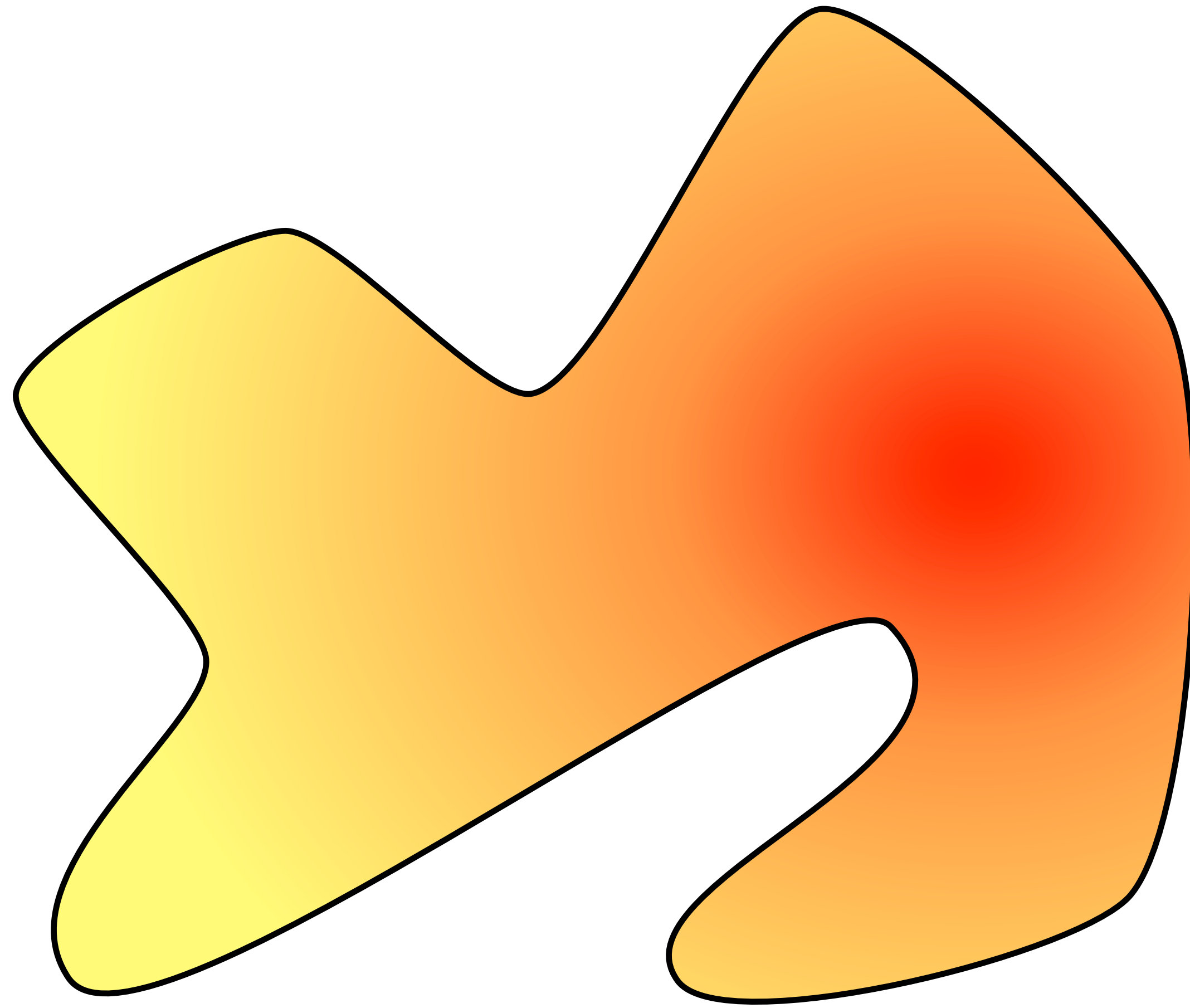
- a configuration c is a local minima with respect to neighborhood N if

$$\forall n \in N(c) : f(n) \geq f(c)$$

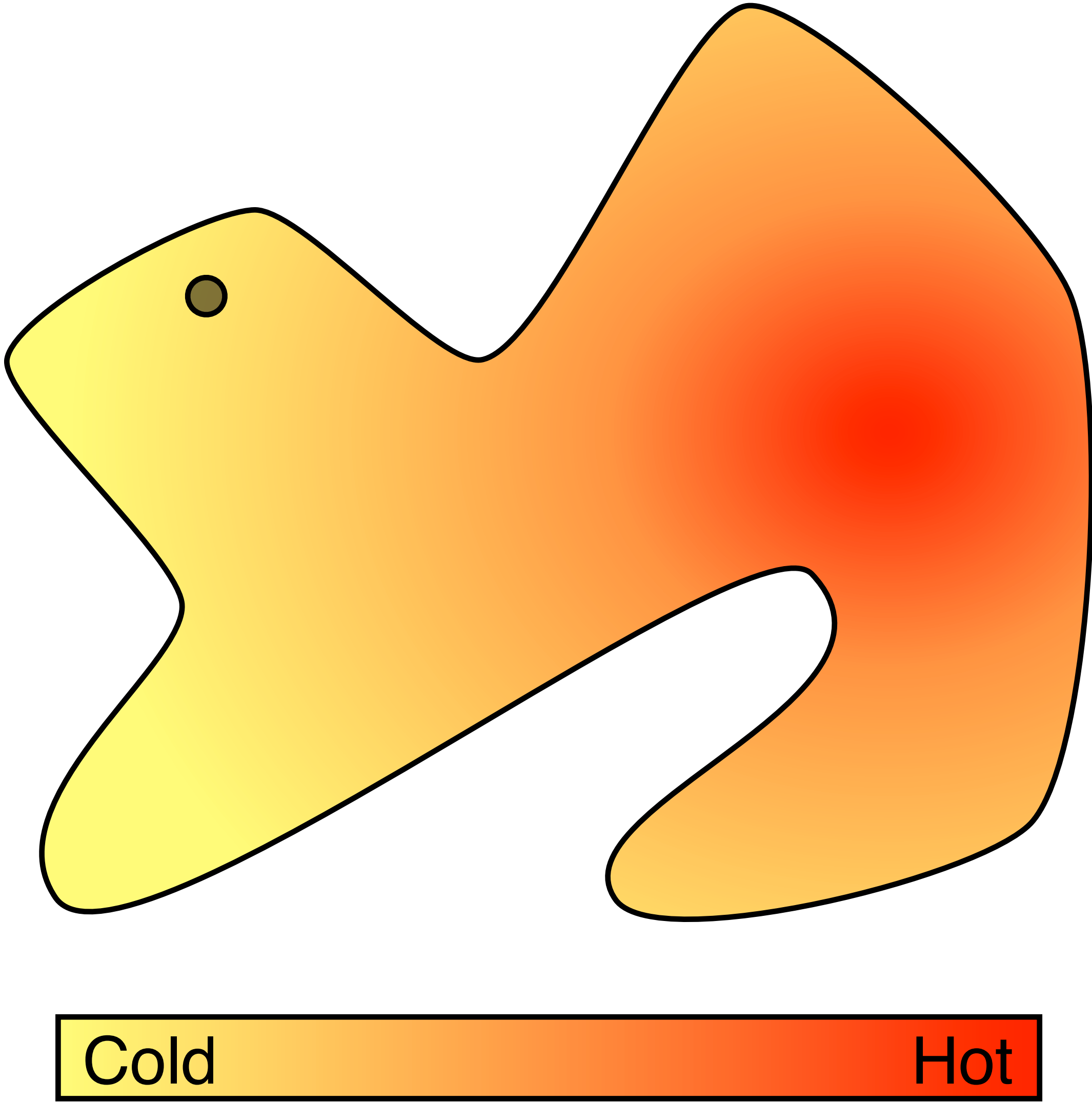
► No guarantees for global optimality

- escaping local optima is a critical issue in local search

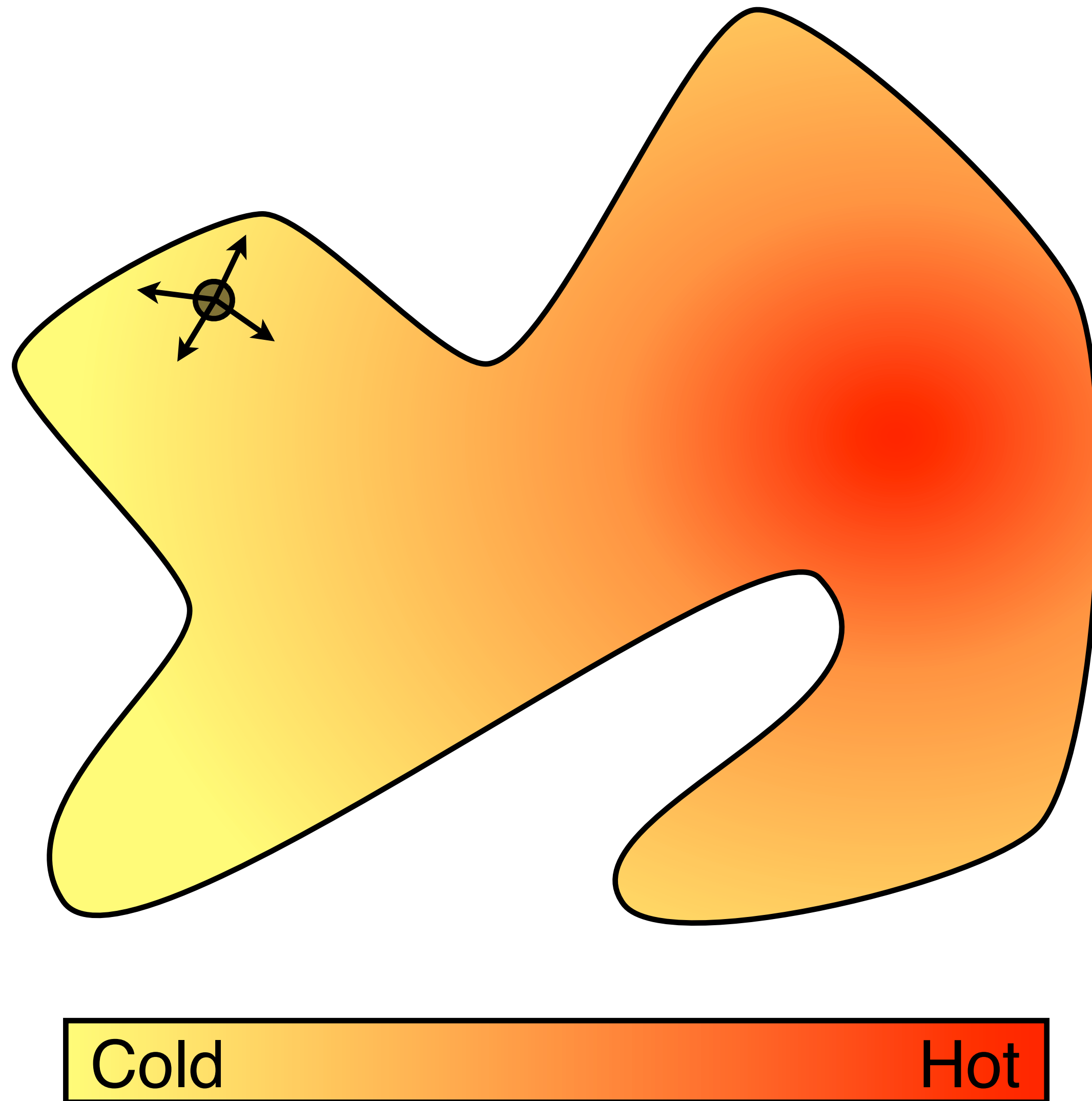
Local Search



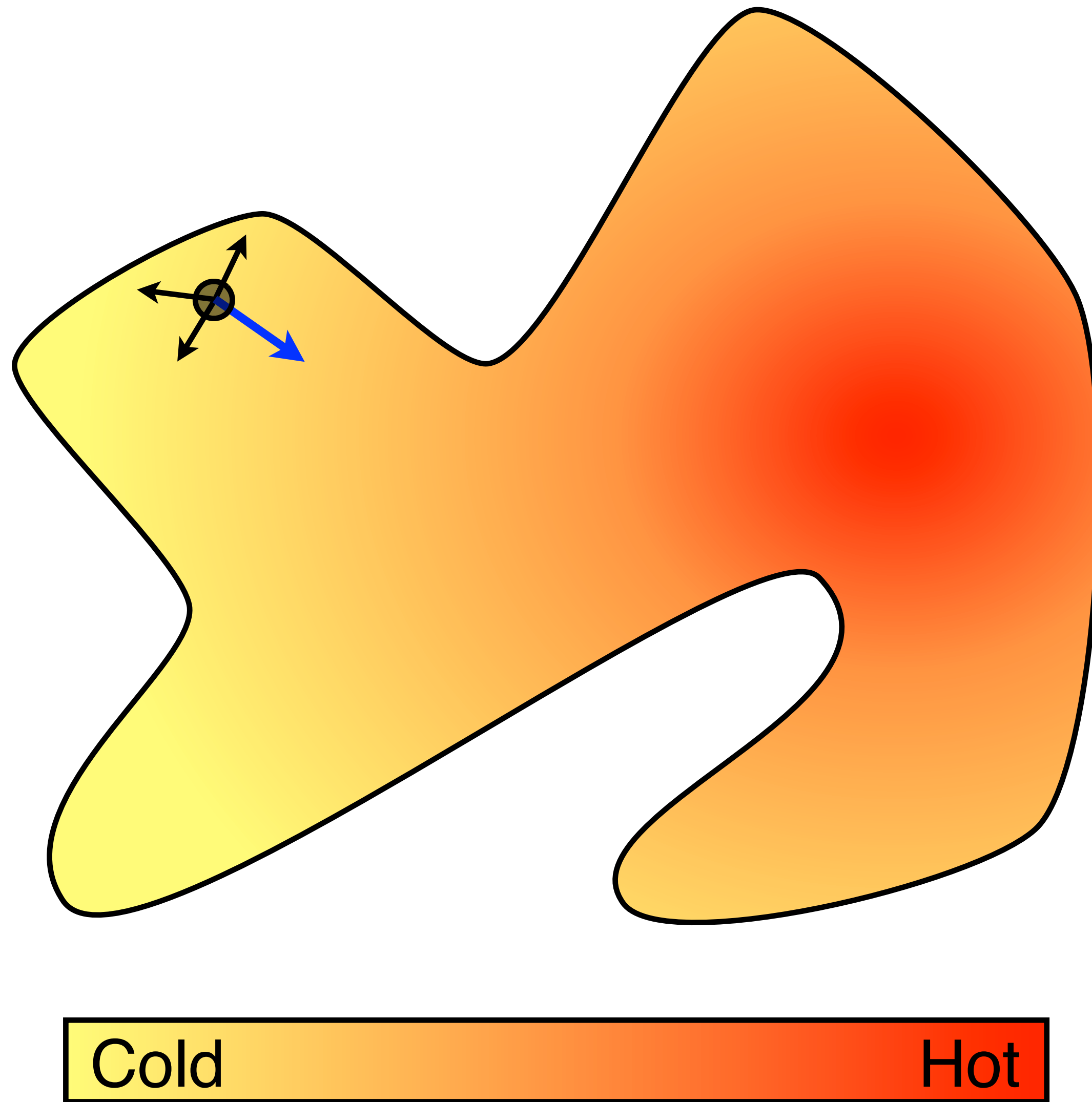
Local Search



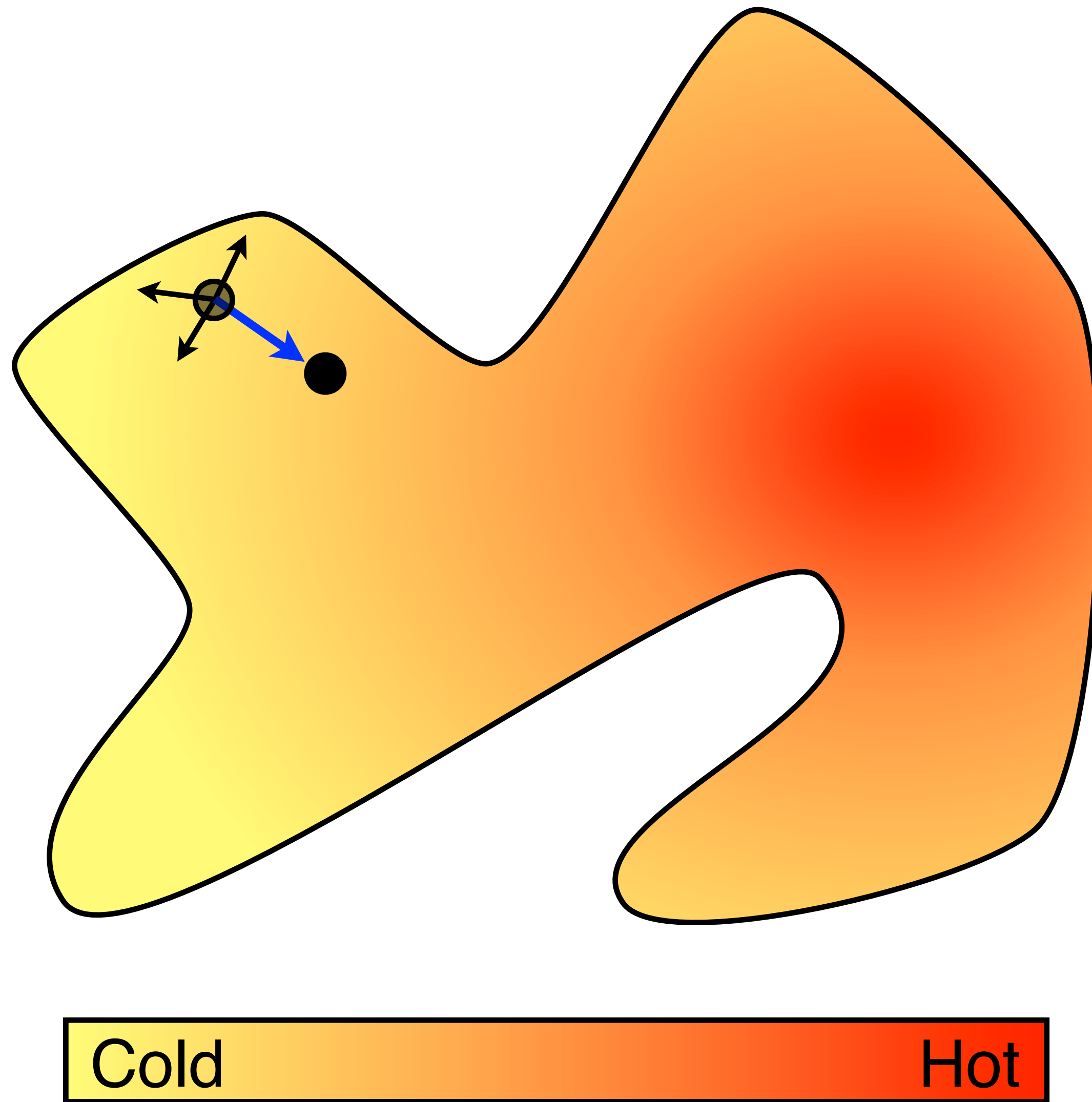
Local Search



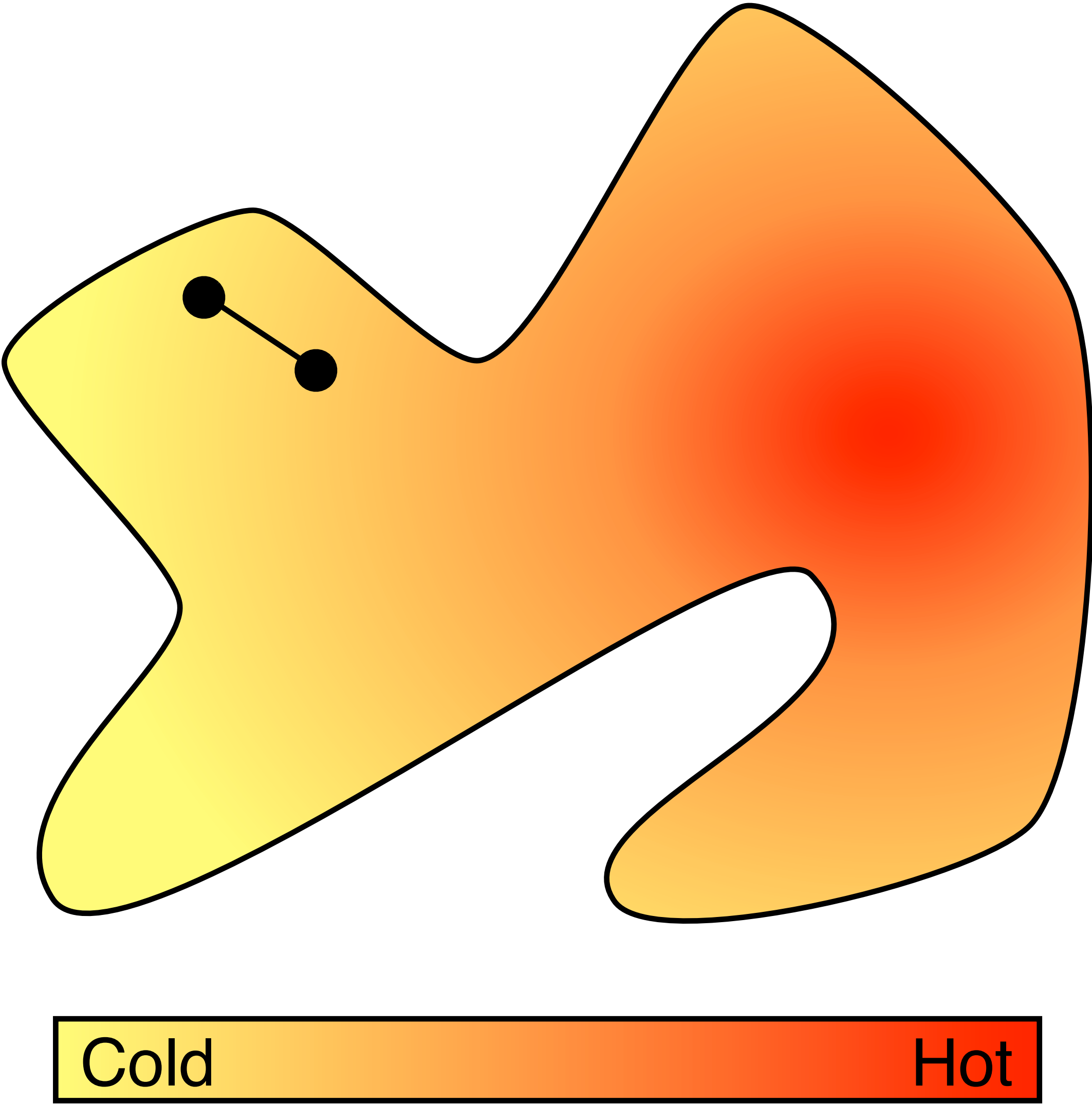
Local Search



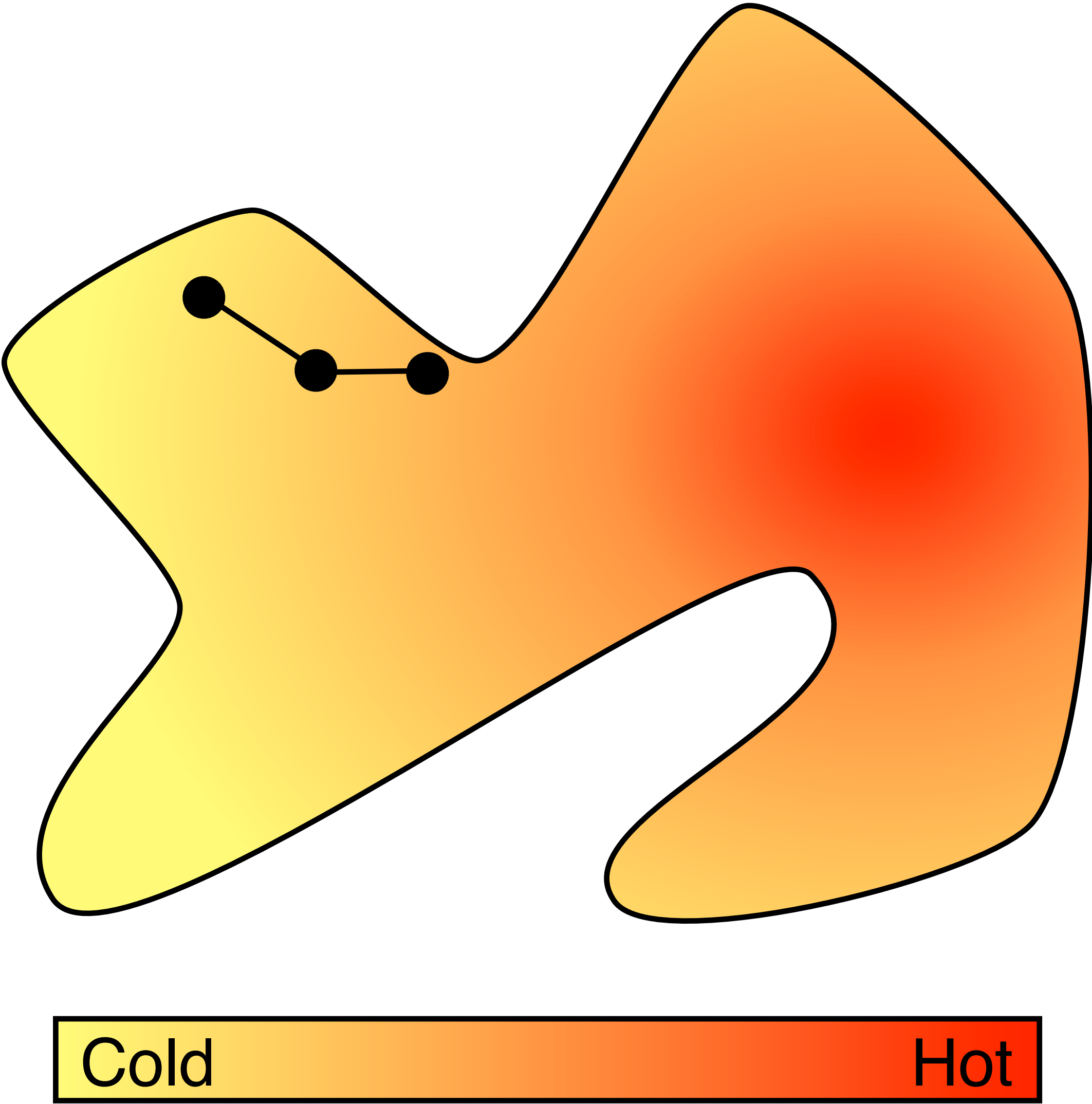
Local Search



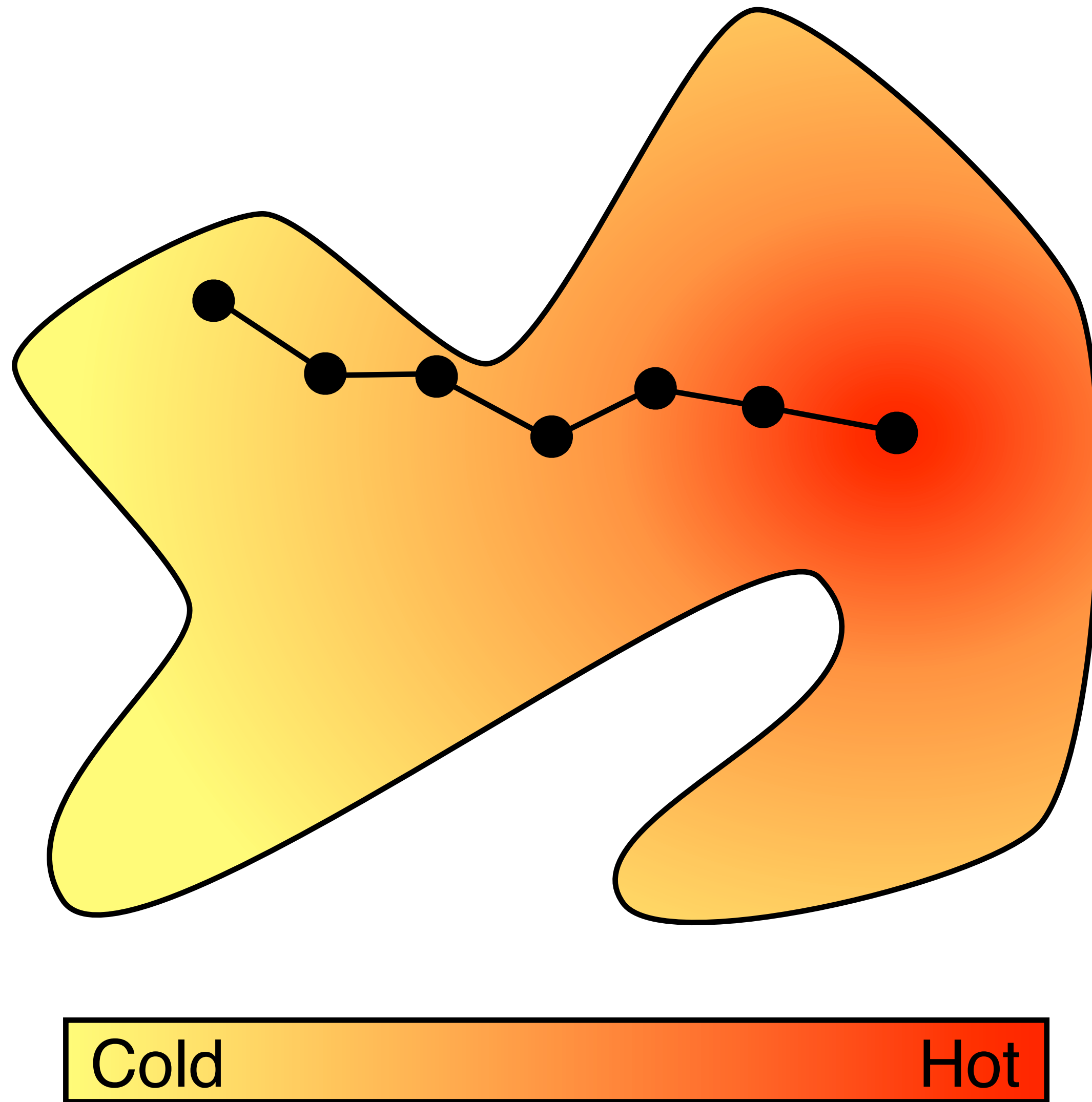
Local Search



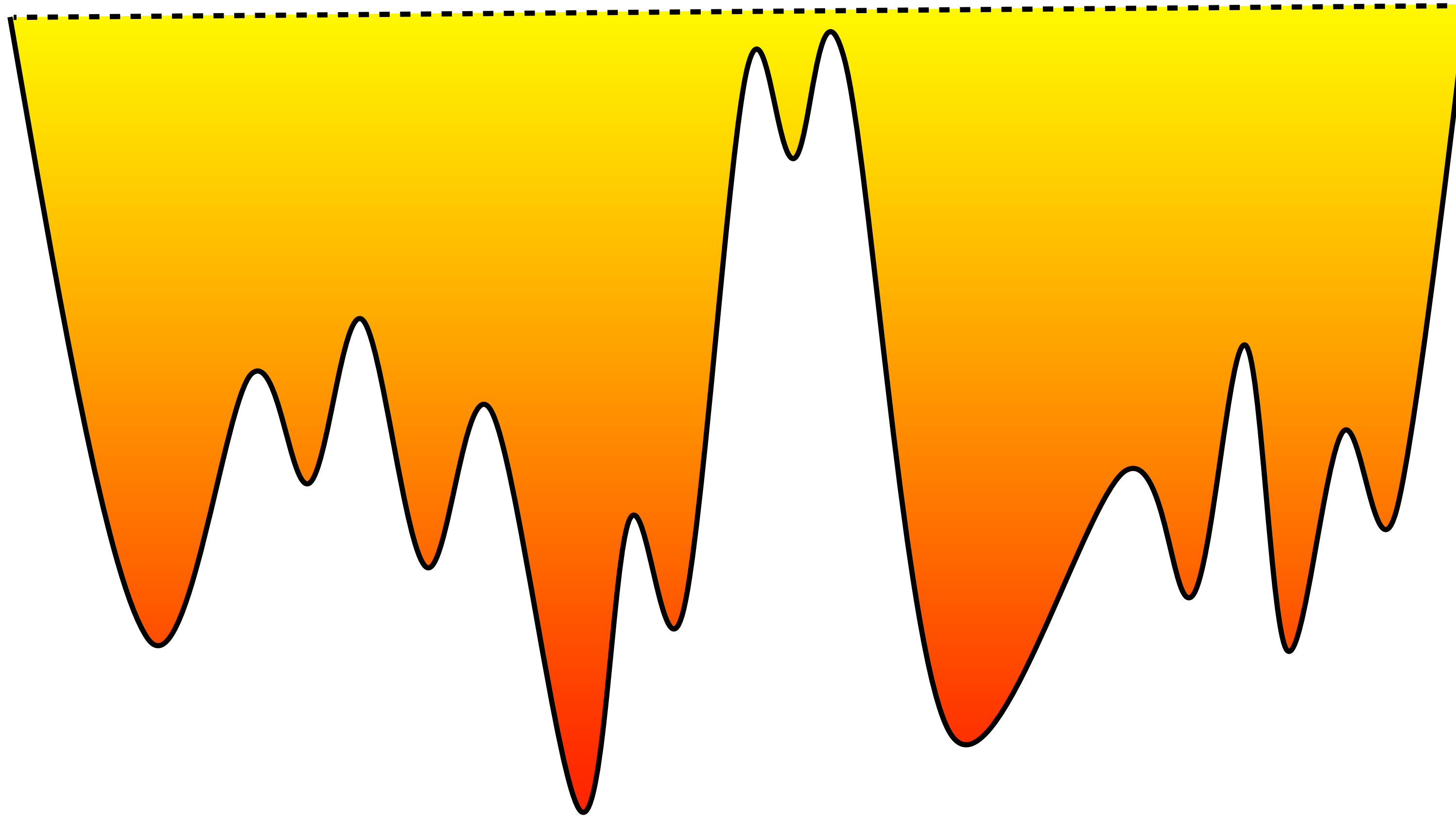
Local Search



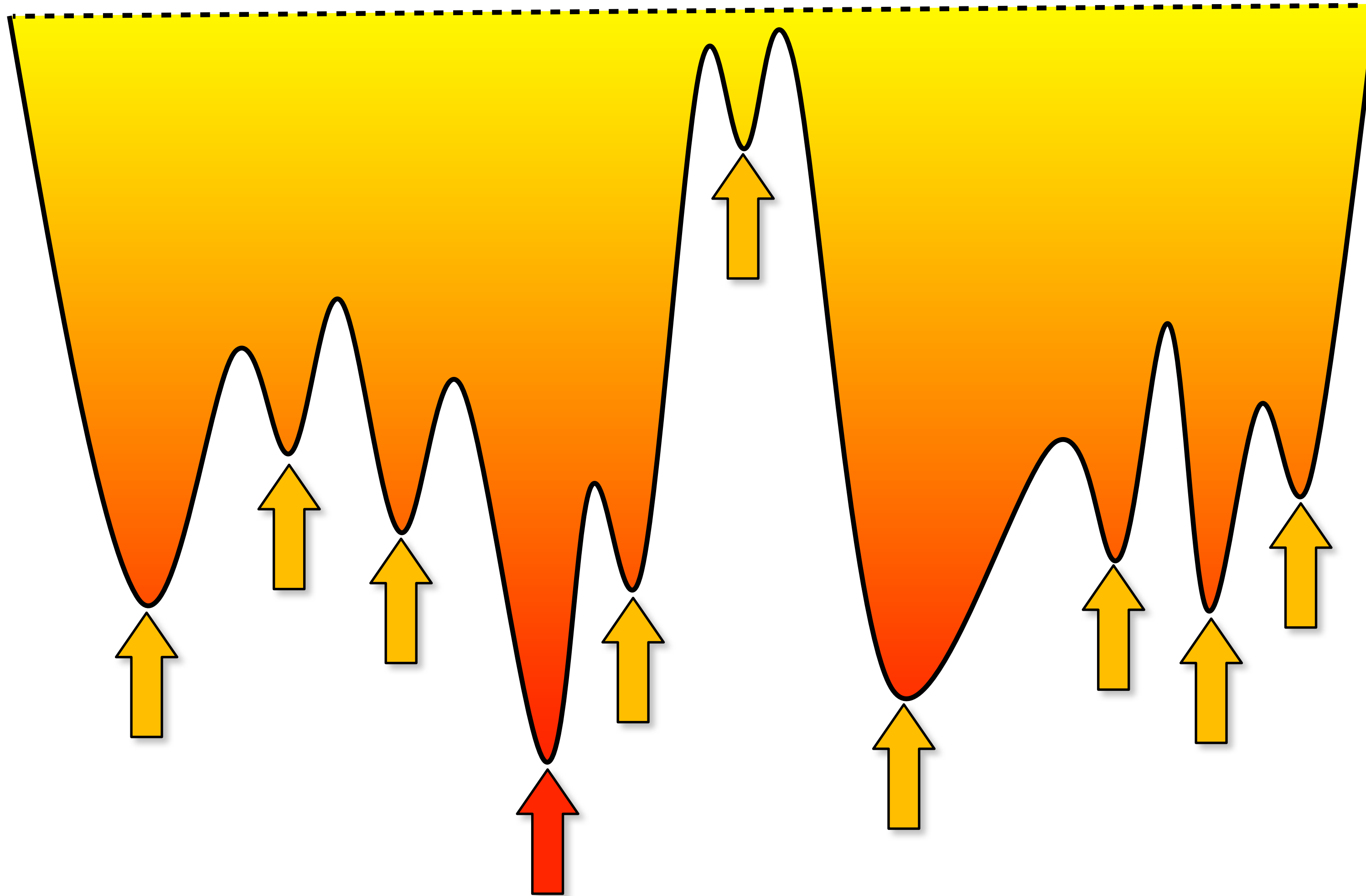
Local Search



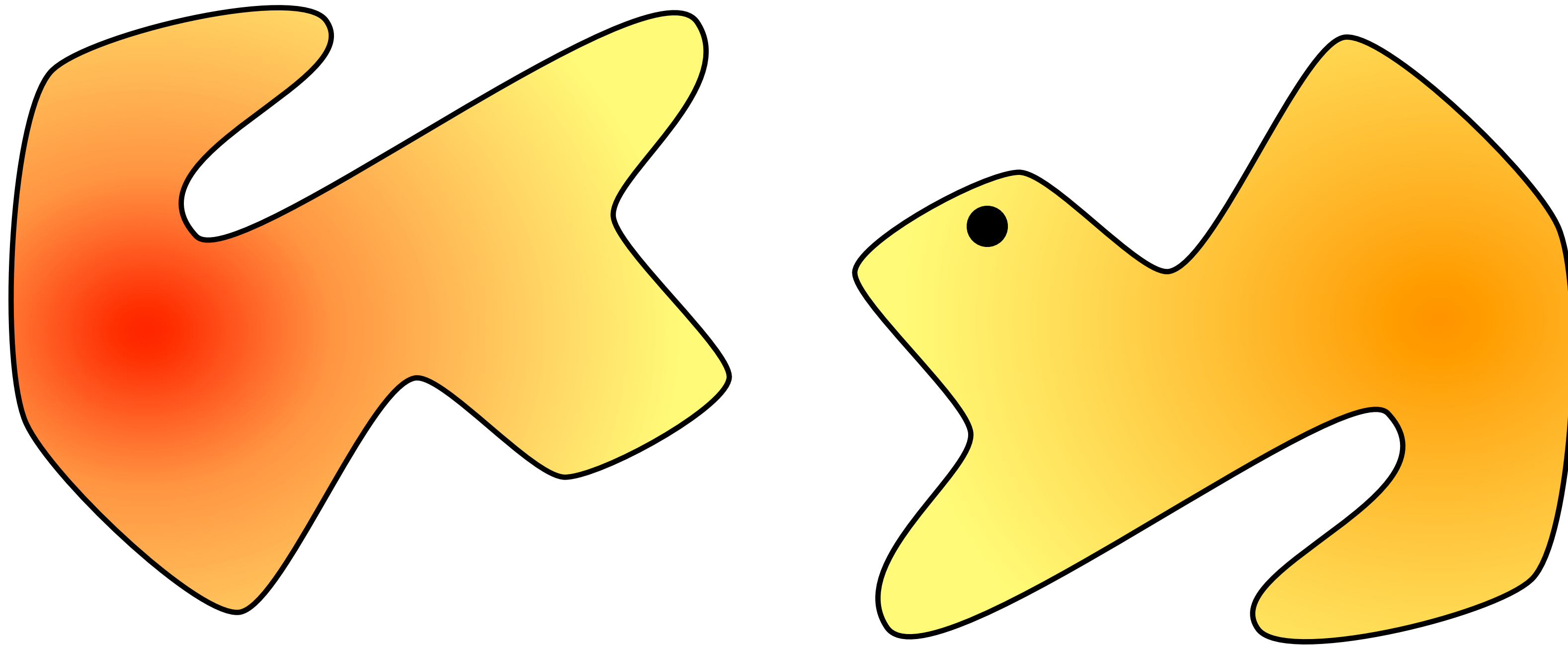
How to Find High-Quality Minima?



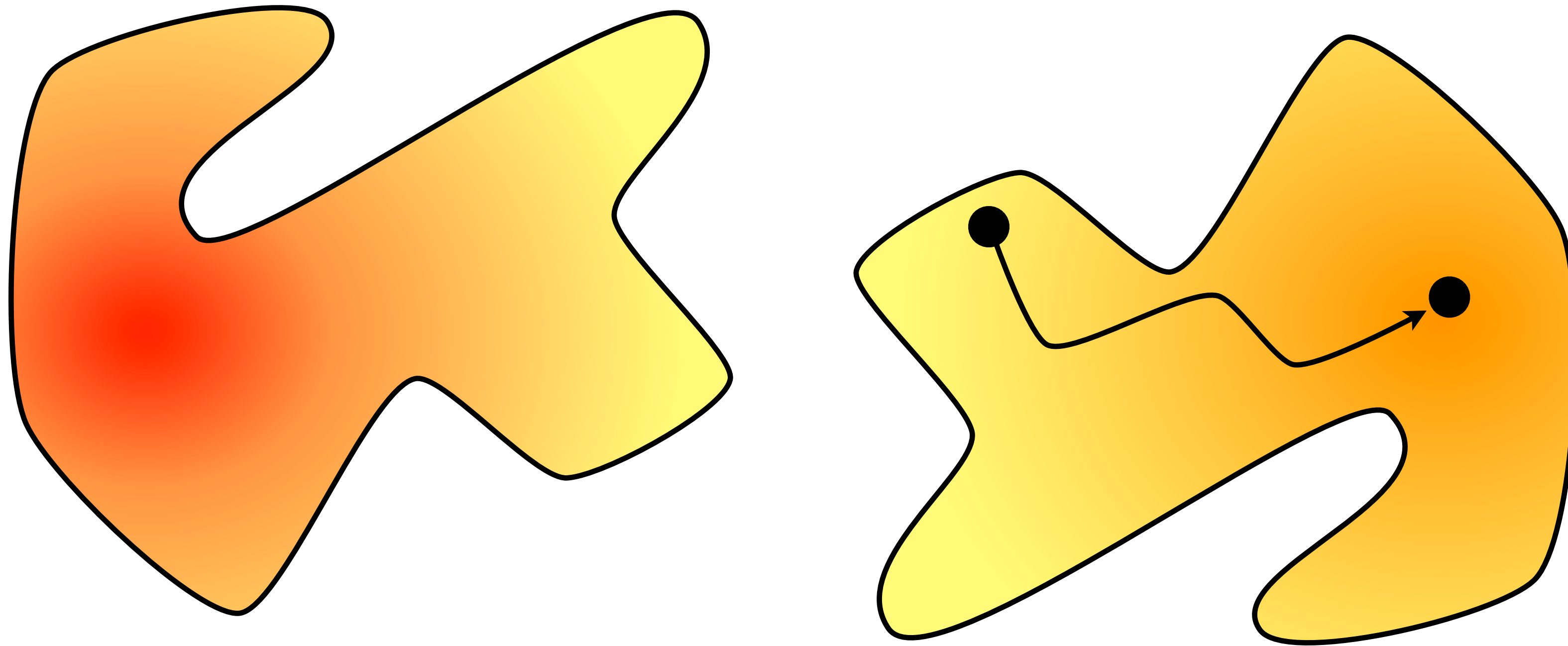
How to Find High-Quality Minima?



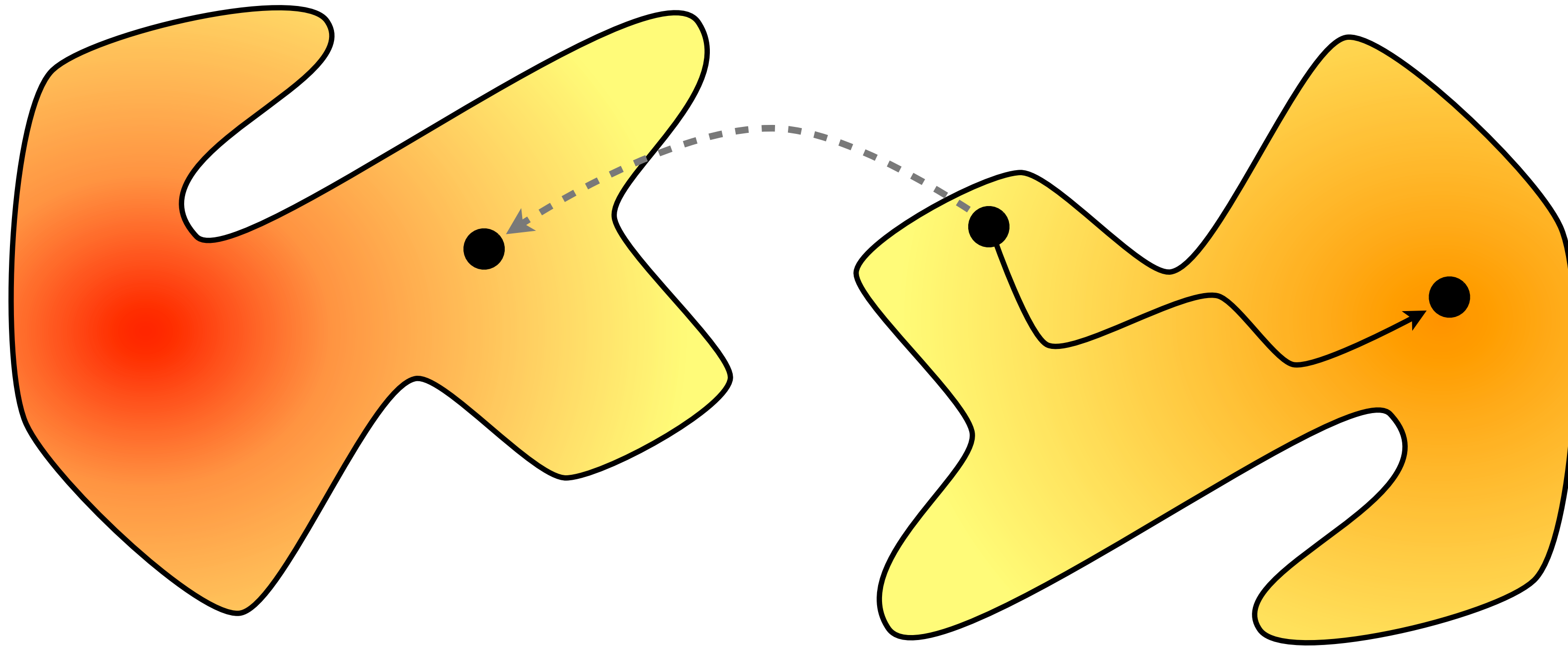
How to Find High-Quality Minima?



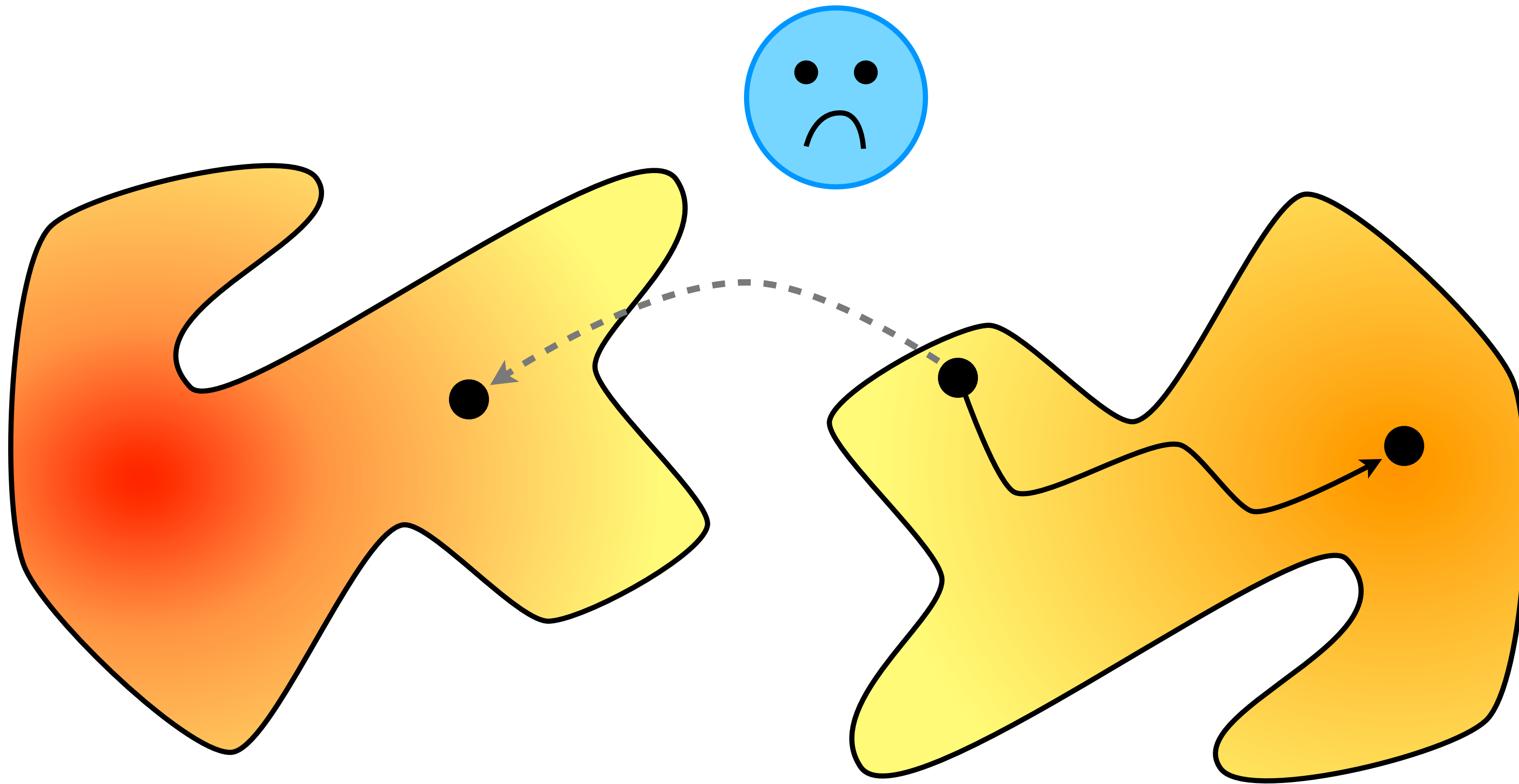
How to Find High-Quality Minima?



How to Find High-Quality Minima?



How to Find High-Quality Minima?



Connectivity

- ▶ A neighborhood N is connected if, from every configuration S , some optimal solution O can be reached by a sequence of moves, i.e.,

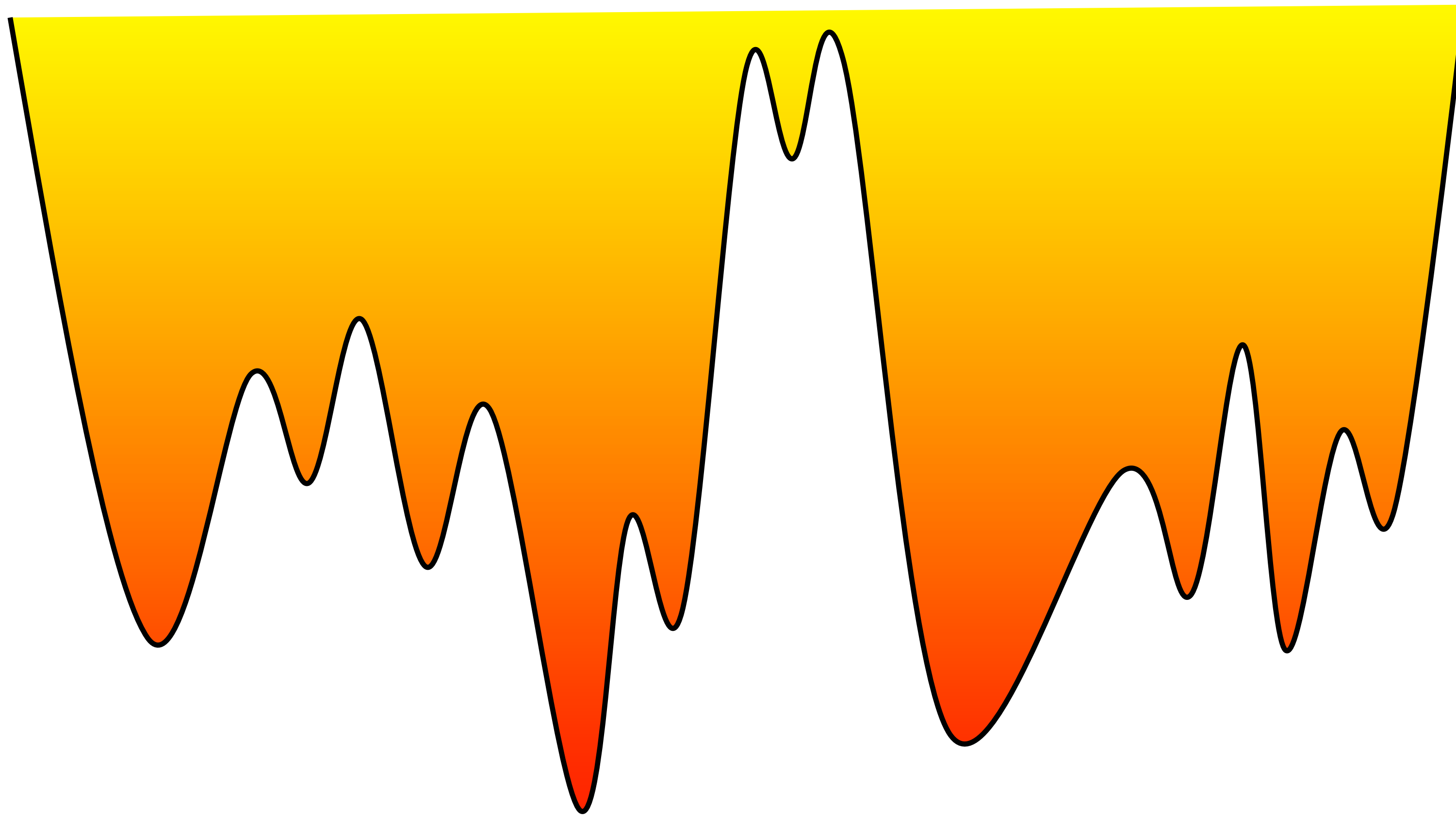
$$S = S_0 \rightarrow S_1 \rightarrow S_2 \rightarrow \dots \rightarrow S_n = O$$

where

$$S_i \in N(S_{i-1}).$$

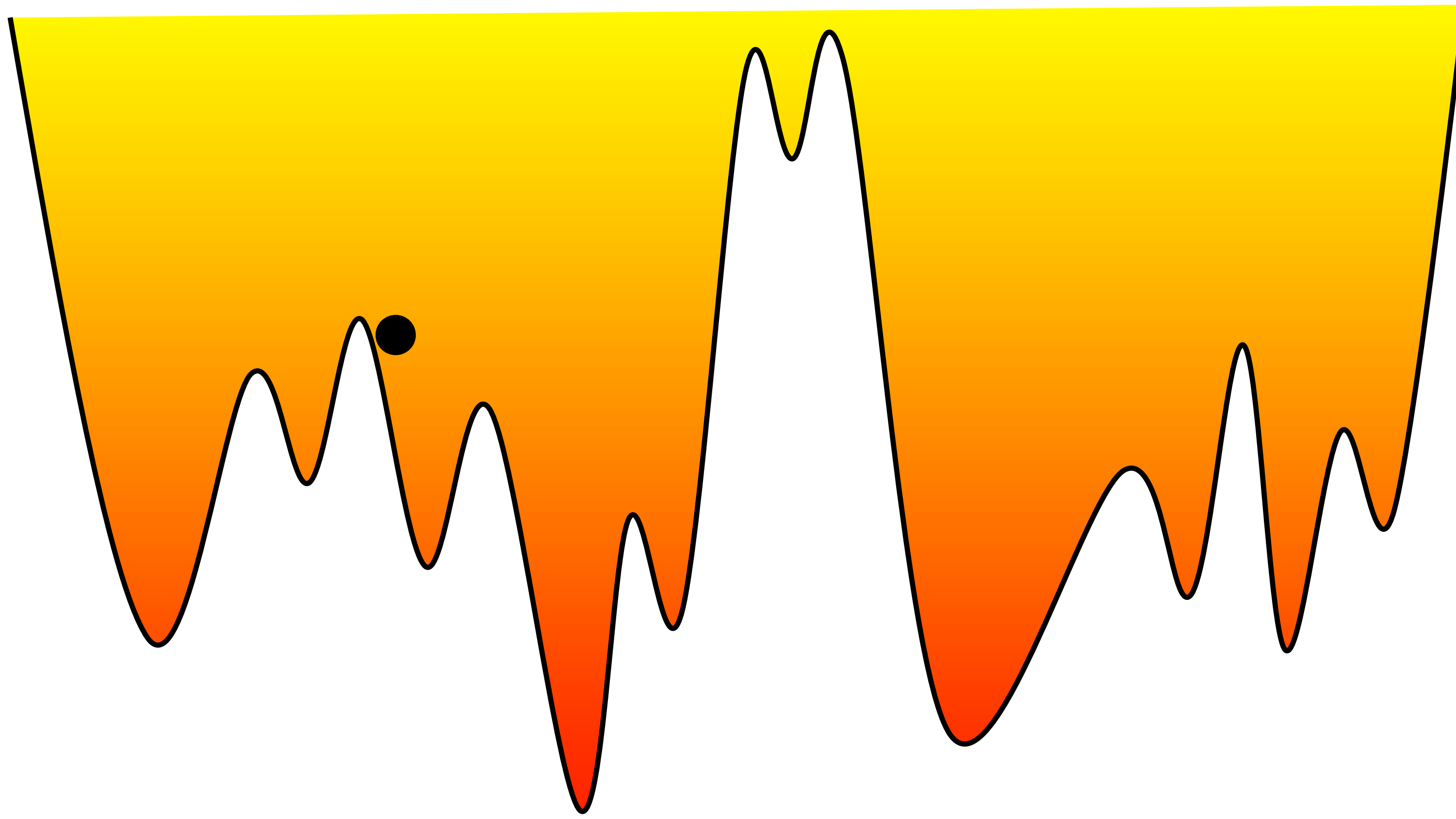
Connectivity

- ▶ Connectivity does not guarantee optimality
 - our local searches have been greedy



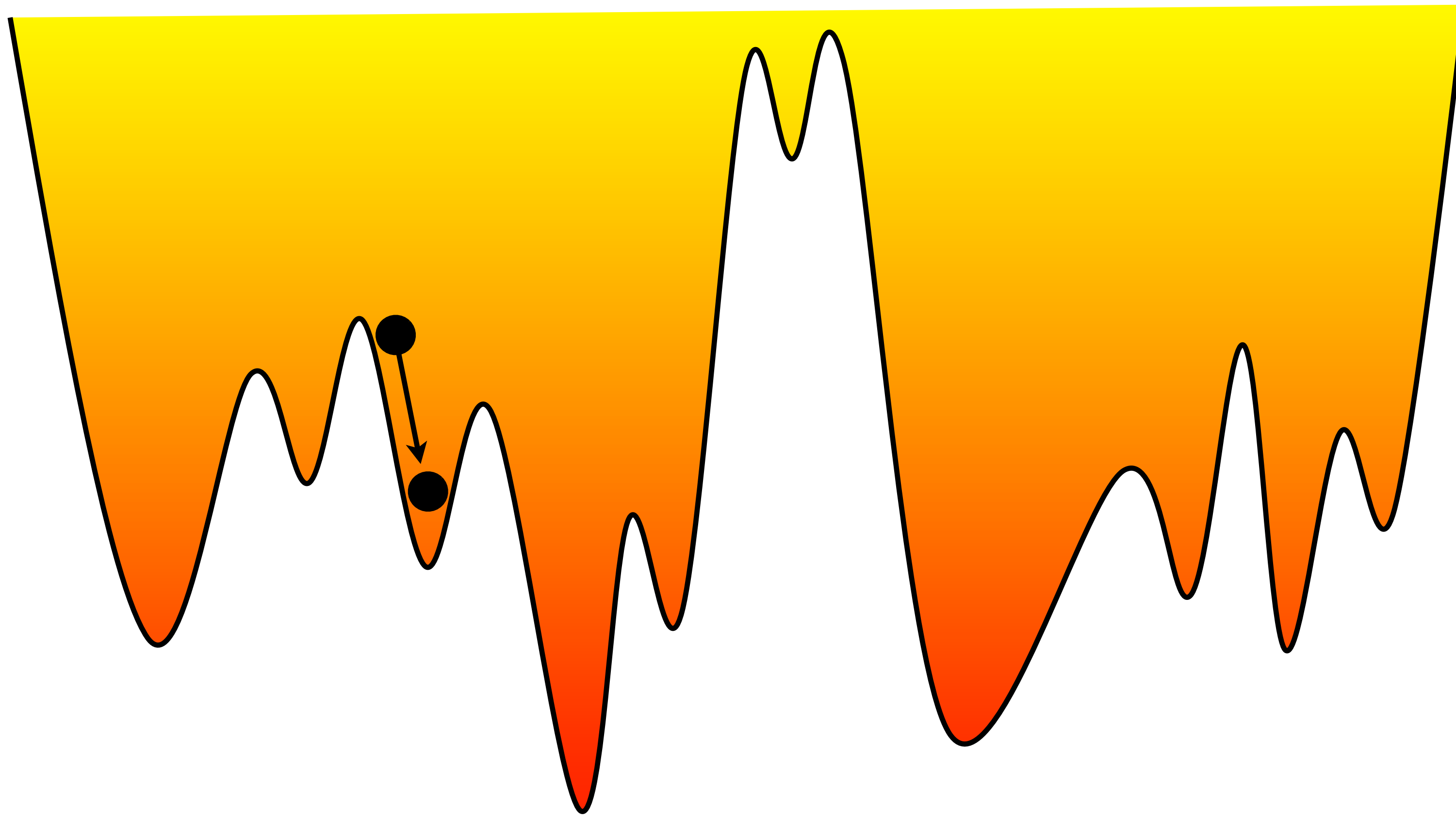
Connectivity

- ▶ Connectivity does not guarantee optimality
 - our local searches have been greedy



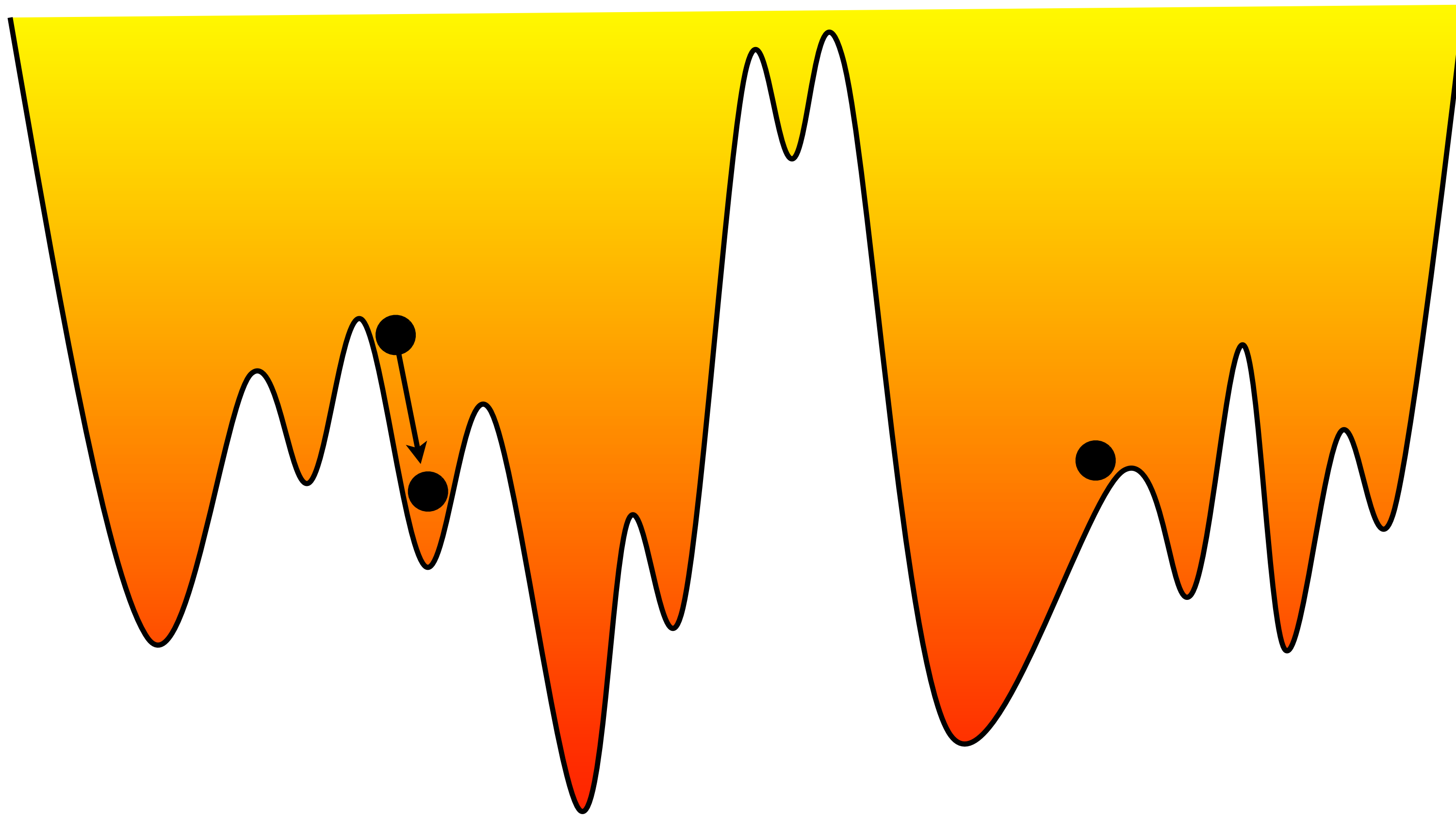
Connectivity

- ▶ Connectivity does not guarantee optimality
 - our local searches have been greedy



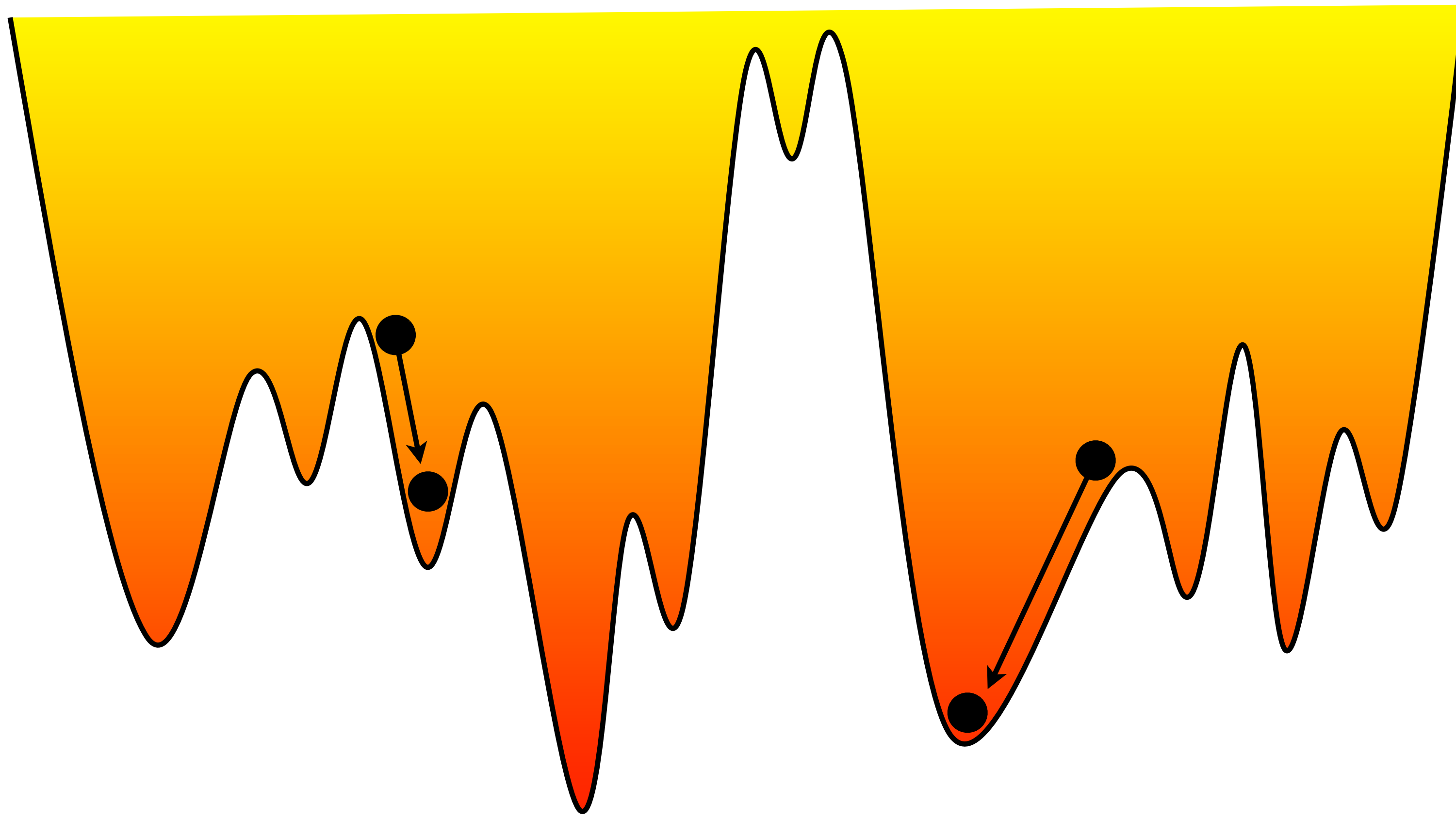
Connectivity

- ▶ Connectivity does not guarantee optimality
 - our local searches have been greedy



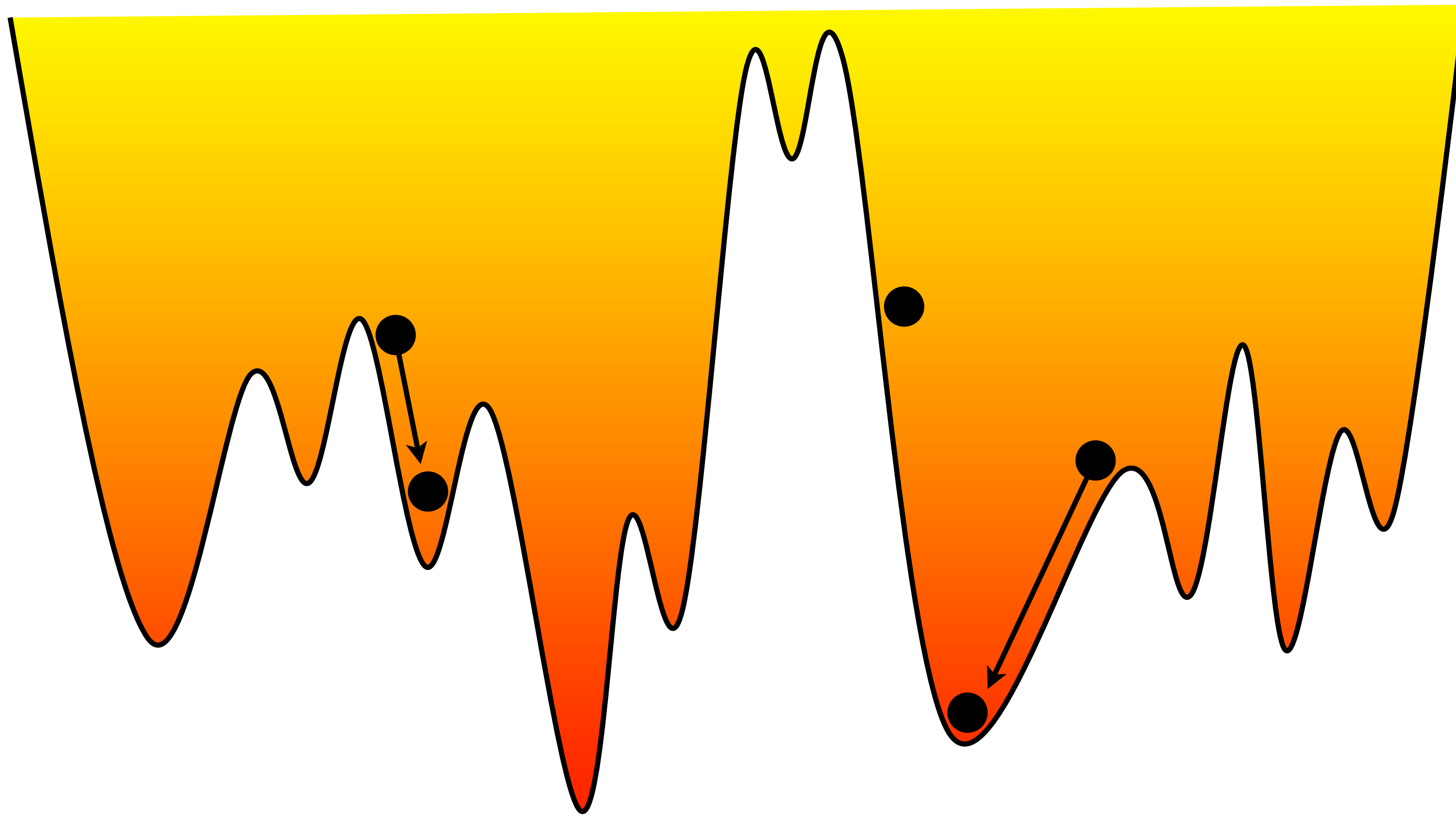
Connectivity

- ▶ Connectivity does not guarantee optimality
 - our local searches have been greedy



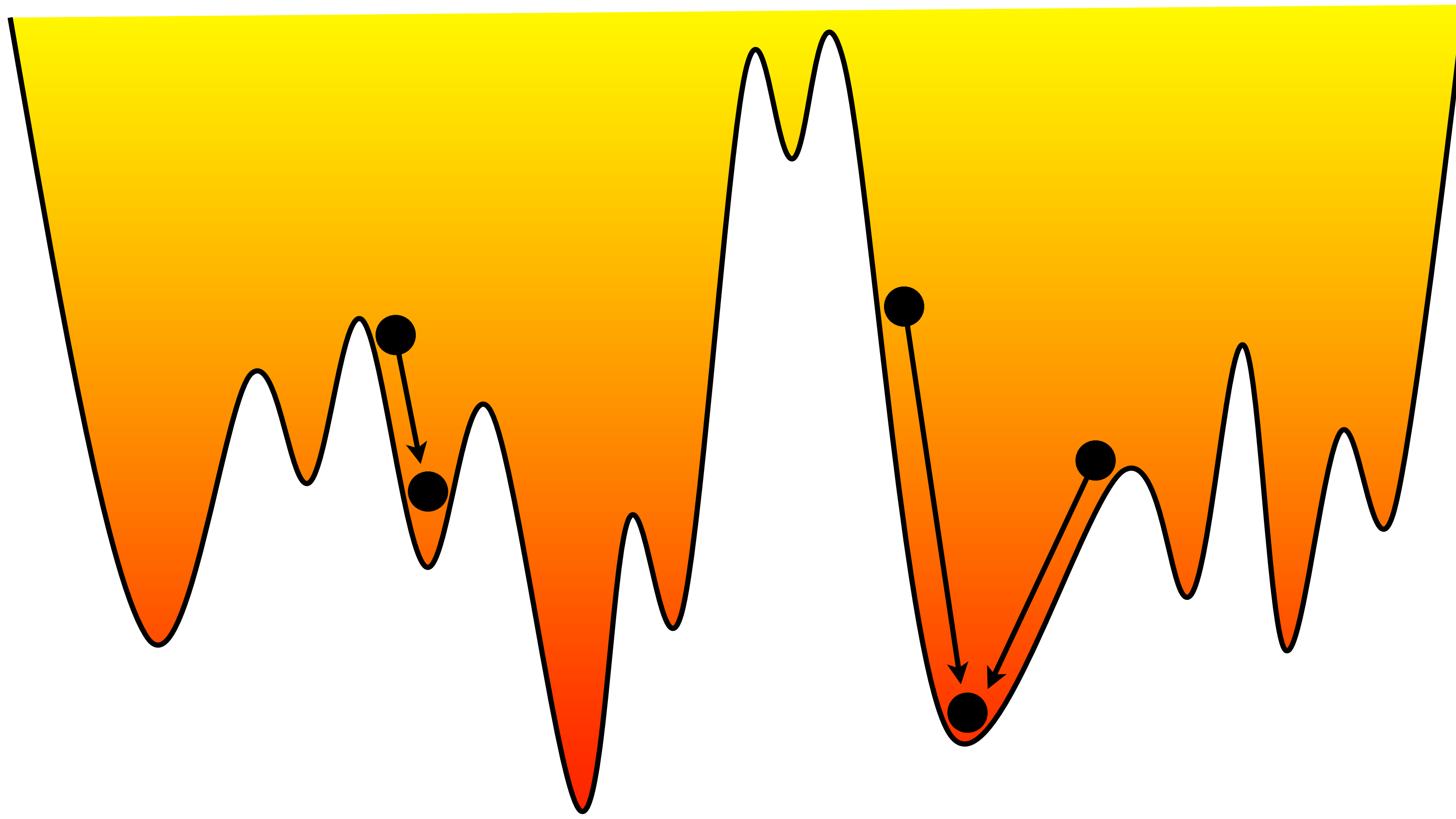
Connectivity

- ▶ Connectivity does not guarantee optimality
 - our local searches have been greedy



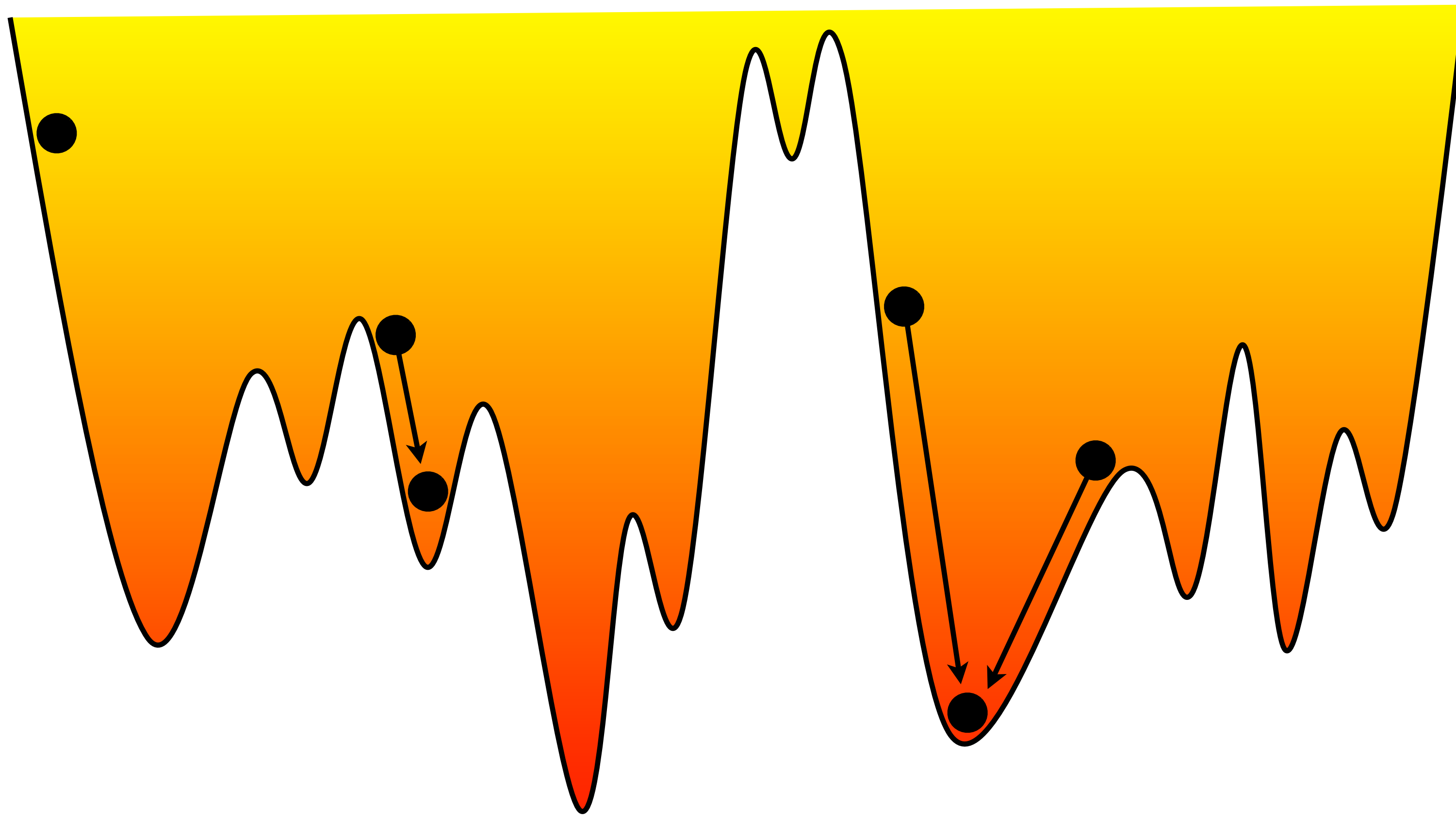
Connectivity

- ▶ Connectivity does not guarantee optimality
 - our local searches have been greedy



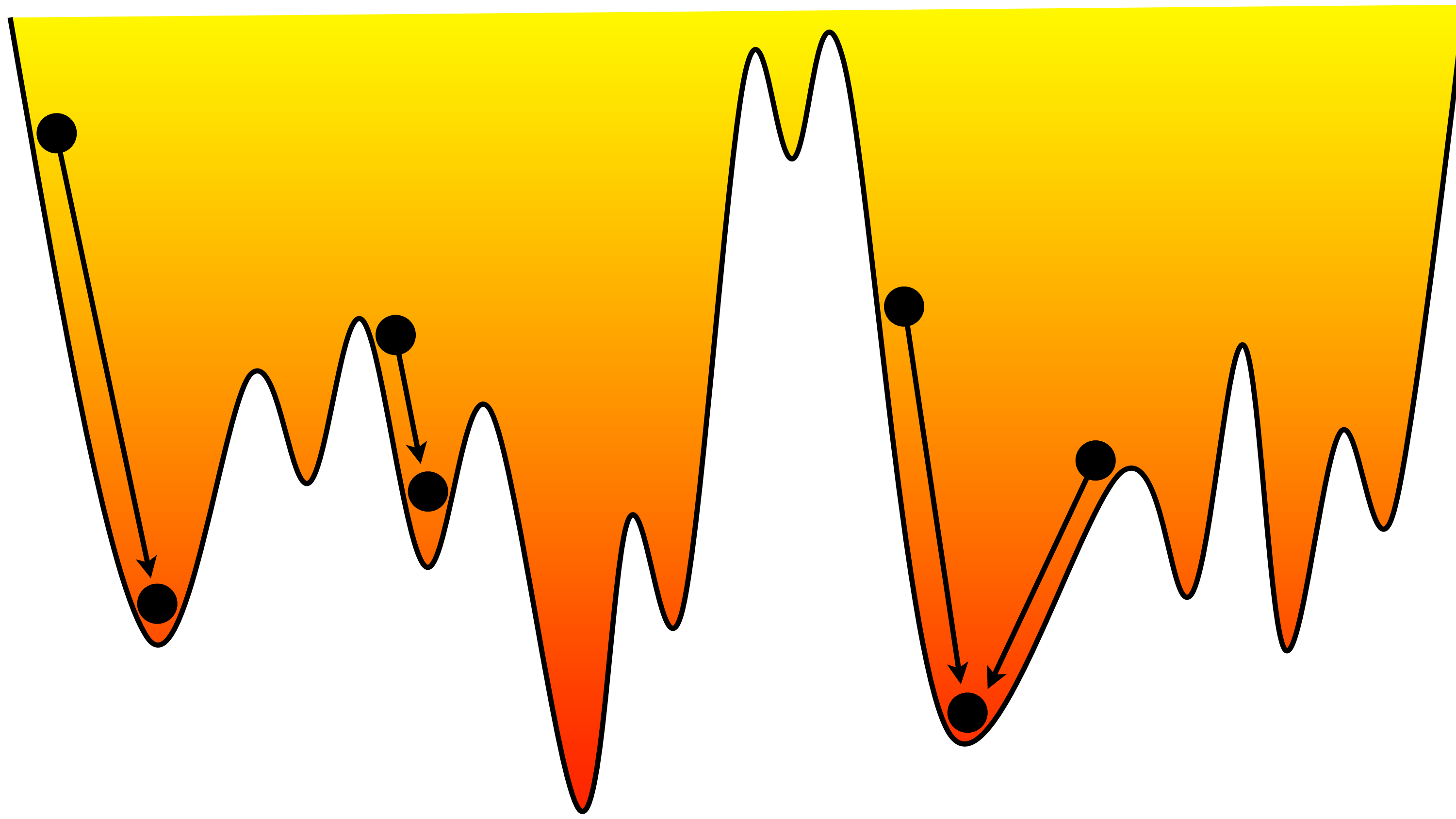
Connectivity

- ▶ Connectivity does not guarantee optimality
 - our local searches have been greedy



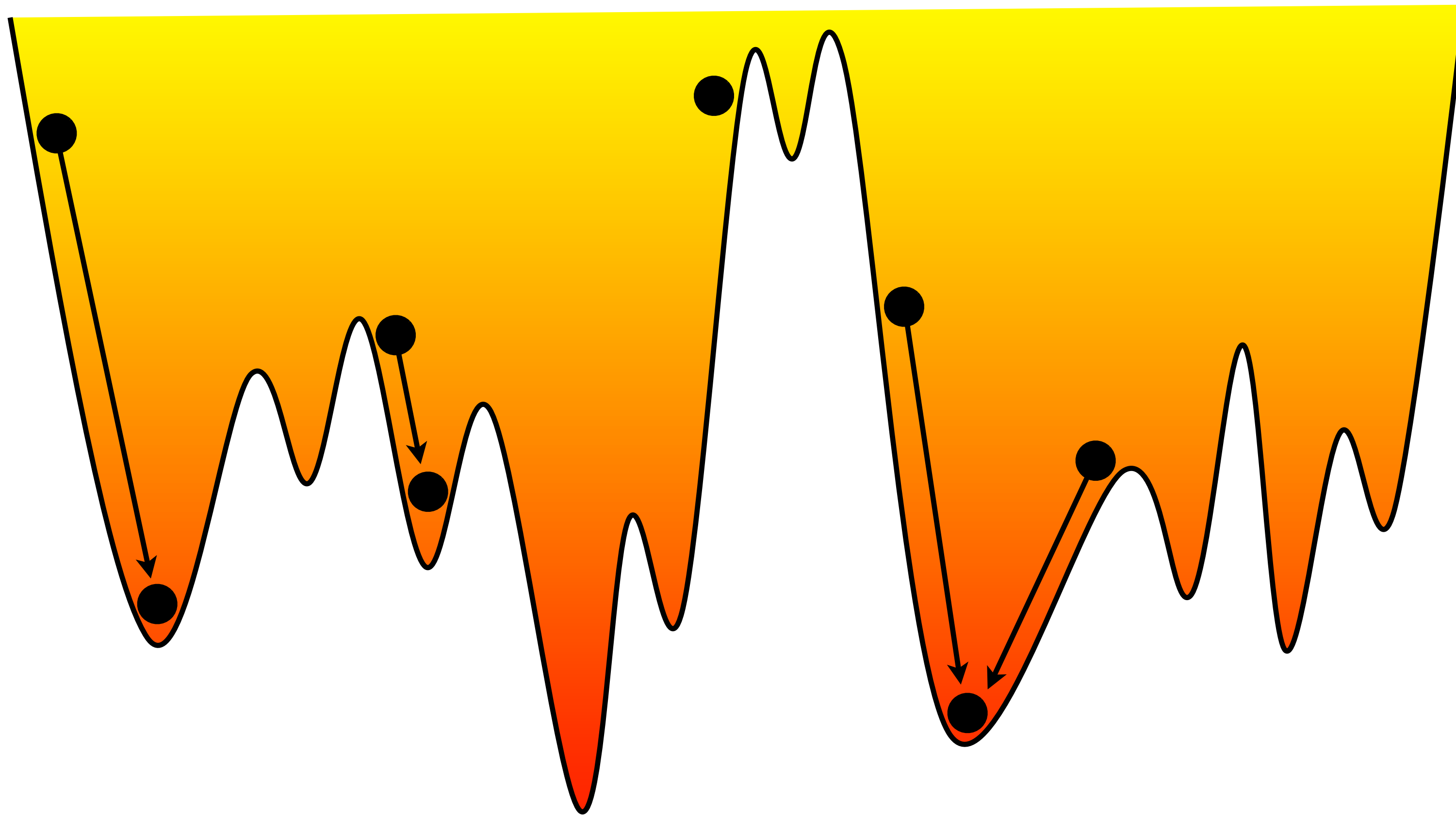
Connectivity

- Connectivity does not guarantee optimality
 - our local searches have been greedy



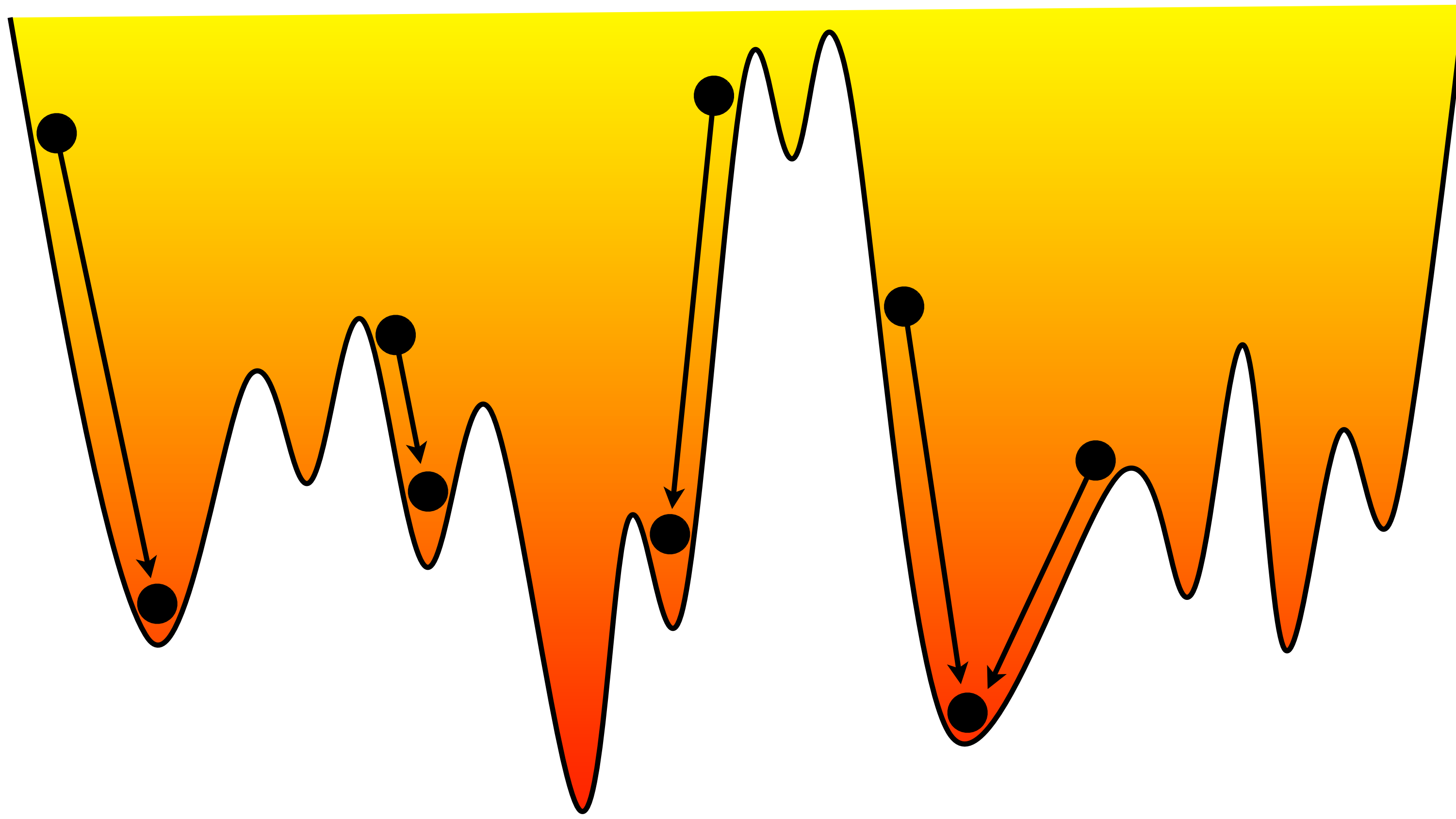
Connectivity

- Connectivity does not guarantee optimality
 - our local searches have been greedy



Connectivity

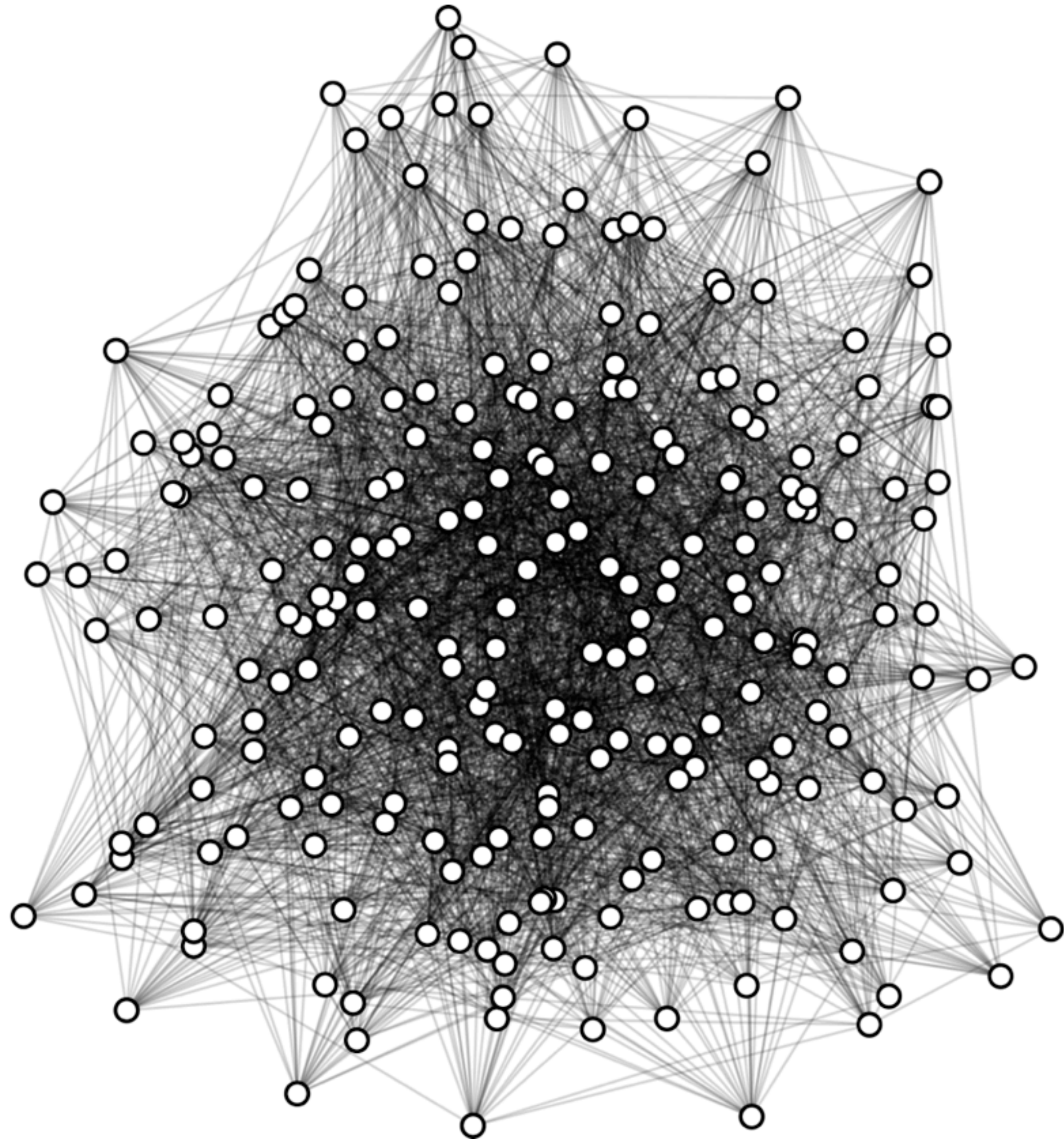
- Connectivity does not guarantee optimality
 - our local searches have been greedy



Connectivity

- ▶ Connectivity does not guarantee optimality
 - our local searches have been greedy

Graph Coloring









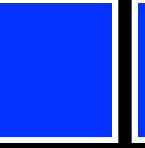



Graph Coloring


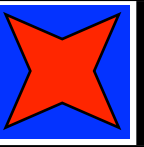
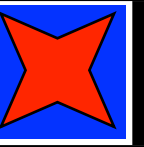
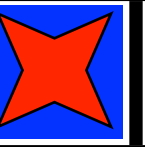
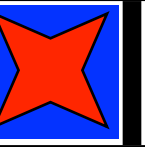
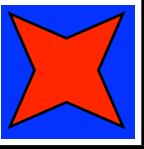
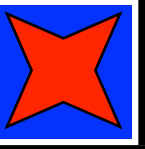
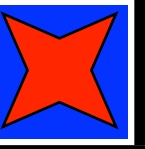
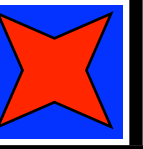



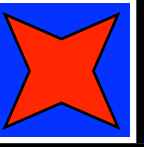



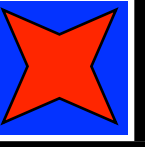
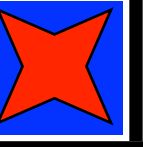
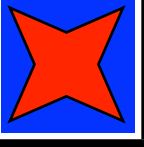
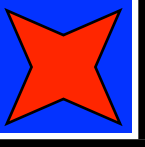
- ▶ Neighborhood
 - Change the color of a node
- ▶ The neighborhood is connected
 - simple algorithm
 - S_n is the color of node n in configuration S
 - O is the optimal configuration

```
S := some configuration
for each node n
  if  $S_n \neq O_n$ 
     $S_n := O_n$ 
```


Car Sequencing

Slots	1	2	3	4	5	6	7	8	9	10	Demand
Class 1											1
Class 2											1
Class 3											2
Class 4											2
Class 5											2
Class 6											2

Options	1	2	3	4	5	Demand
Class 1	yes		yes	yes		1
Class 2				yes		1
Class 3		yes			yes	2
Class 4		yes		yes		2
Class 5	yes		yes			2
Class 6	yes	yes				2
Capacity	1/2	2/3	1/3	2/5	1/5	

Setup	1	2	3	4	5	6	7	8	9	10	Capacity
Option 1											1/2
Option 2											2/3
Option 3											1/3
Option 4											2/5
Option 5											1/5

3
2
2
2
3

Car Sequencing

Slots	1	2	3	4	5	6	7	8	9	10	Demand
Class 1											1
Class 2											1
Class 3											2
Class 4											2
Class 5											2
Class 6											2

Options	1	2	3	4	5	Demand
Class 1	yes		yes	yes		1
Class 2				yes		1
Class 3		yes			yes	2
Class 4		yes		yes		2
Class 5	yes		yes			2
Class 6	yes	yes				2
Capacity	1/2	2/3	1/3	2/5	1/5	

Setup	1	2	3	4	5	6	7	8	9	10	Capacity
Option 1											1/2
Option 2											2/3
Option 3											1/3
Option 4											2/5
Option 5											1/5

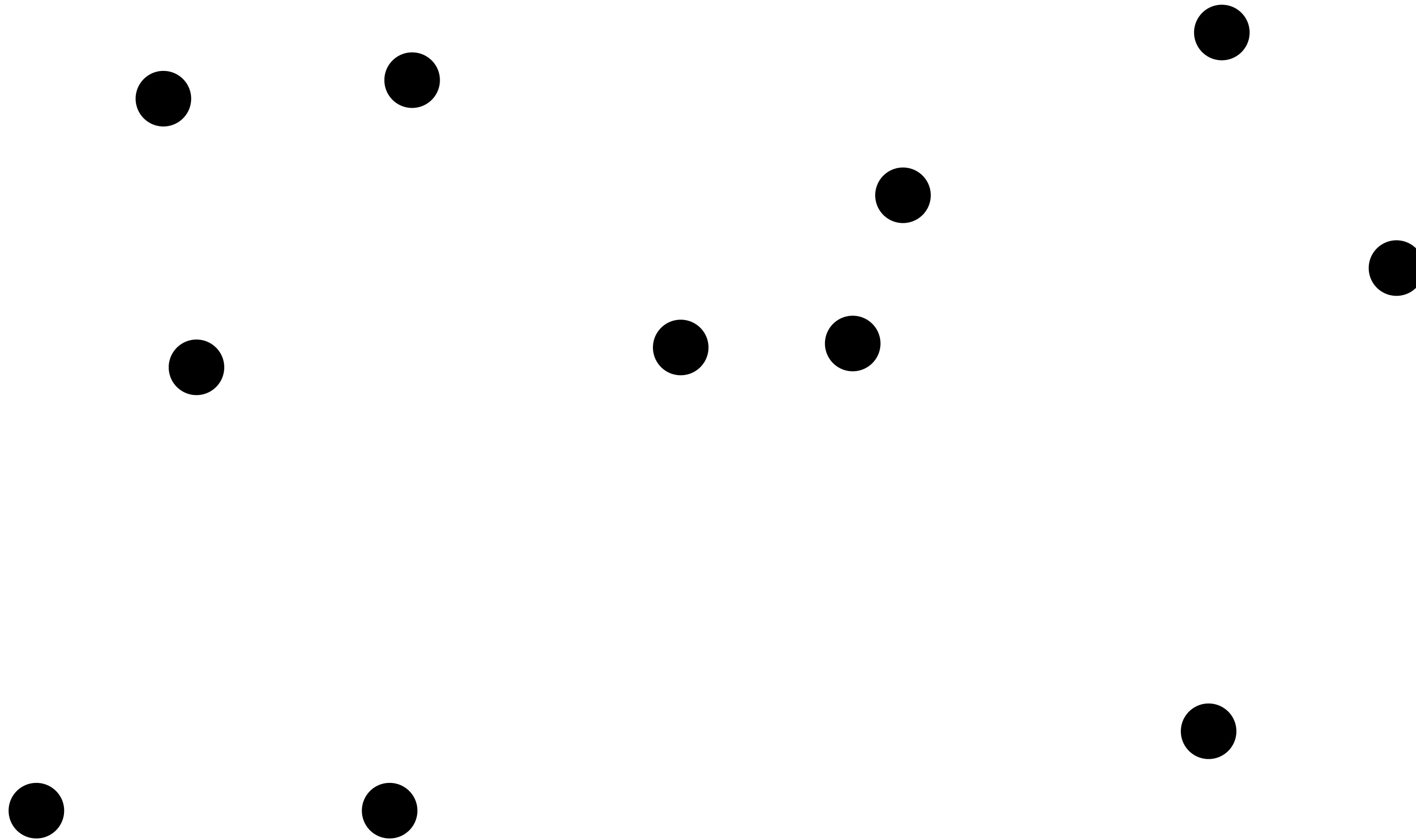
3
2
2
2
3

The Swap Neighborhood is Connected

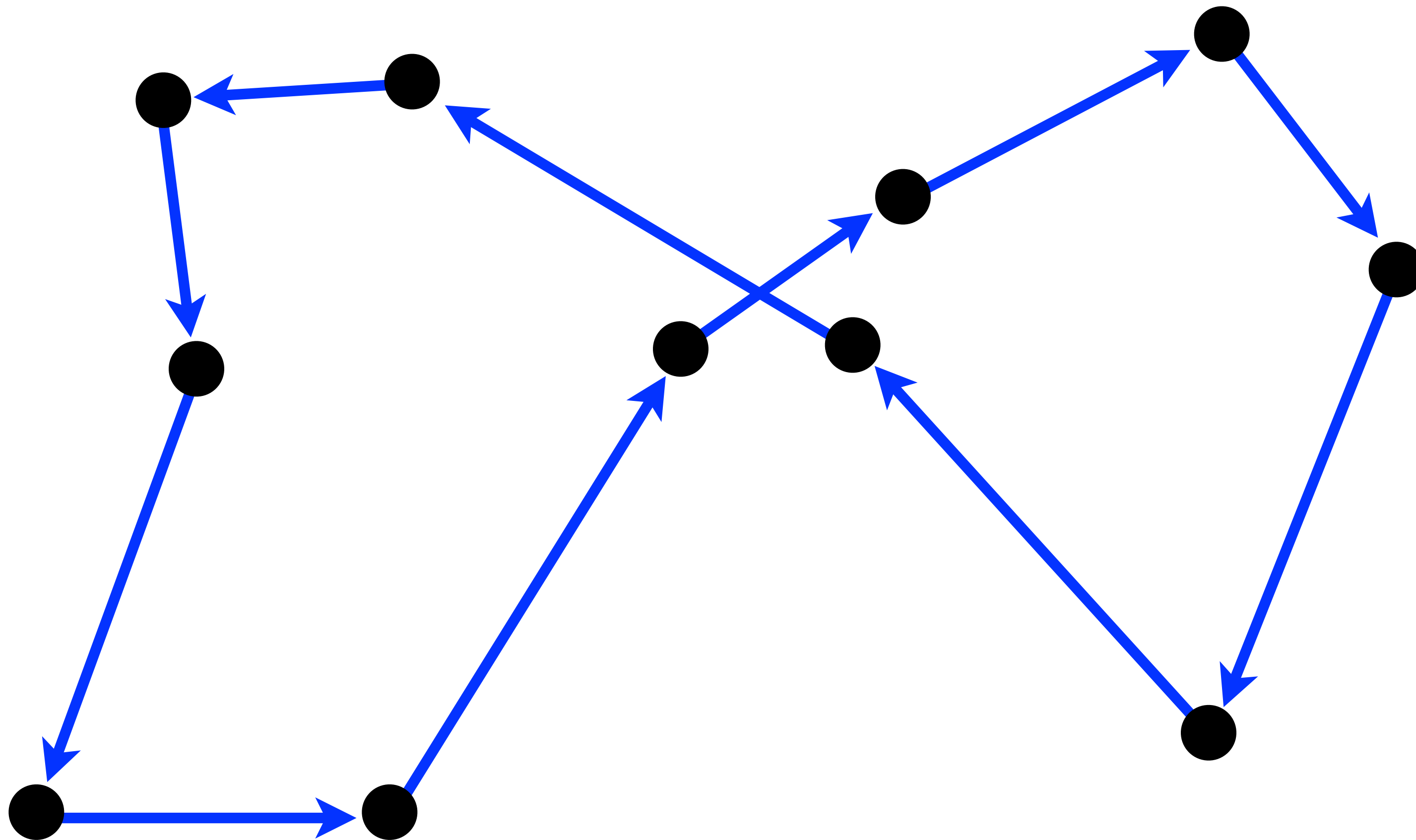
- ▶ Neighborhood
 - Swap two slots in the assembly
- ▶ The neighborhood is connected
 - simple algorithm

```
S := some configuration
for(int i = 1; i <= n; i++)
    if (Si != Oi)
        let Sj = Oi (j > i)
        Si <-> Sj
```


Traveling Salesman Problem

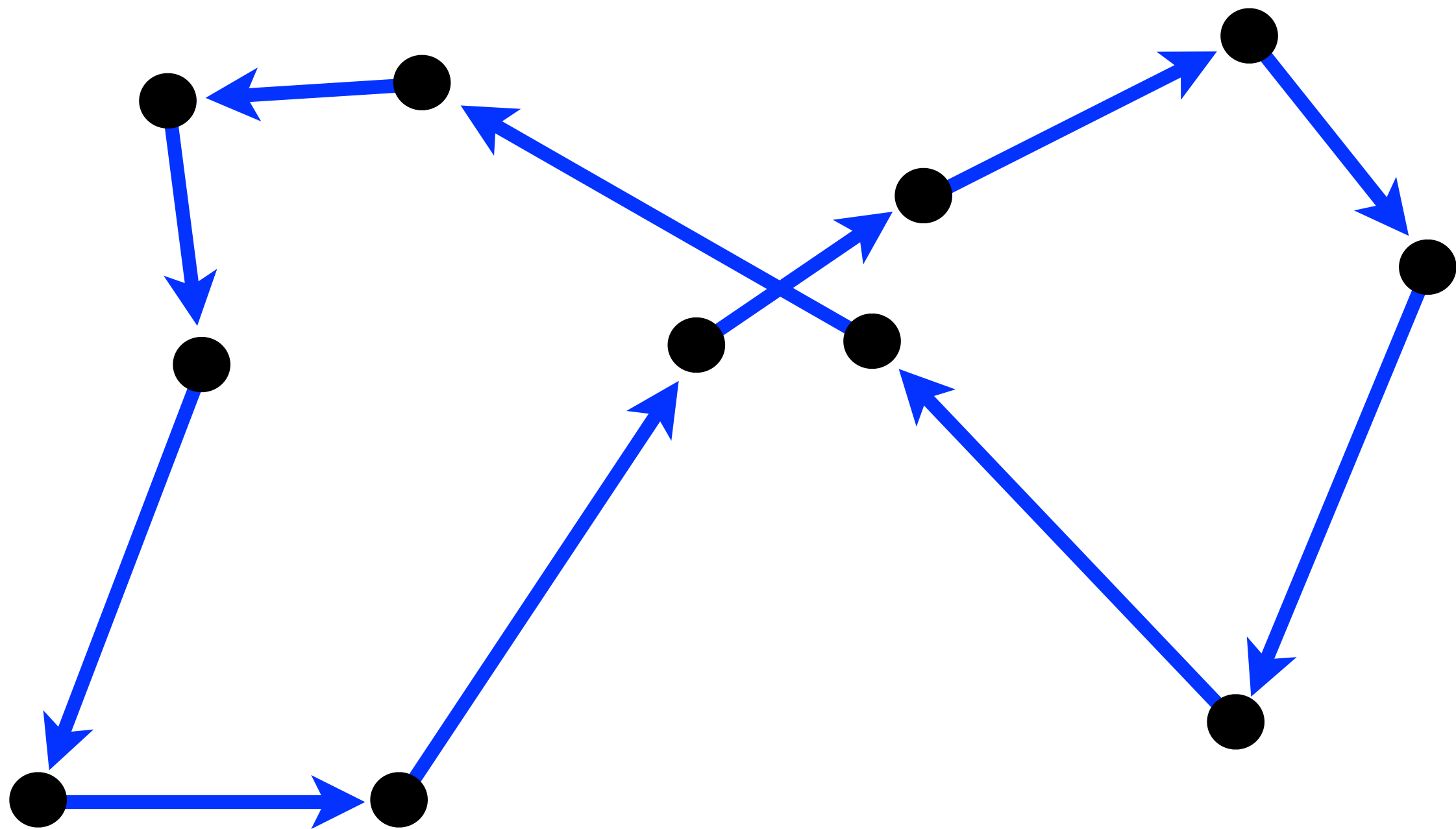


2-OPT

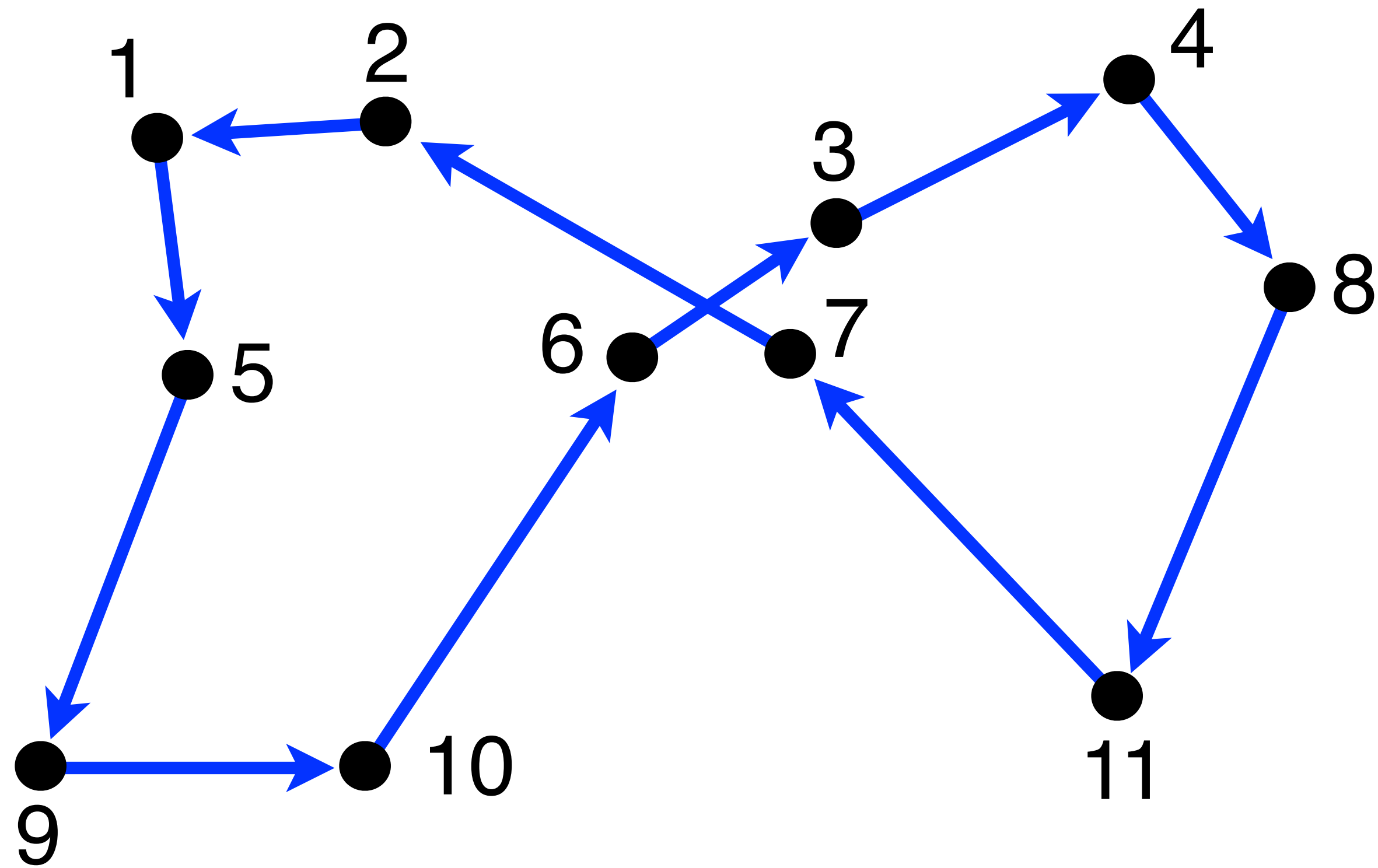


Is 2-OPT Connected?

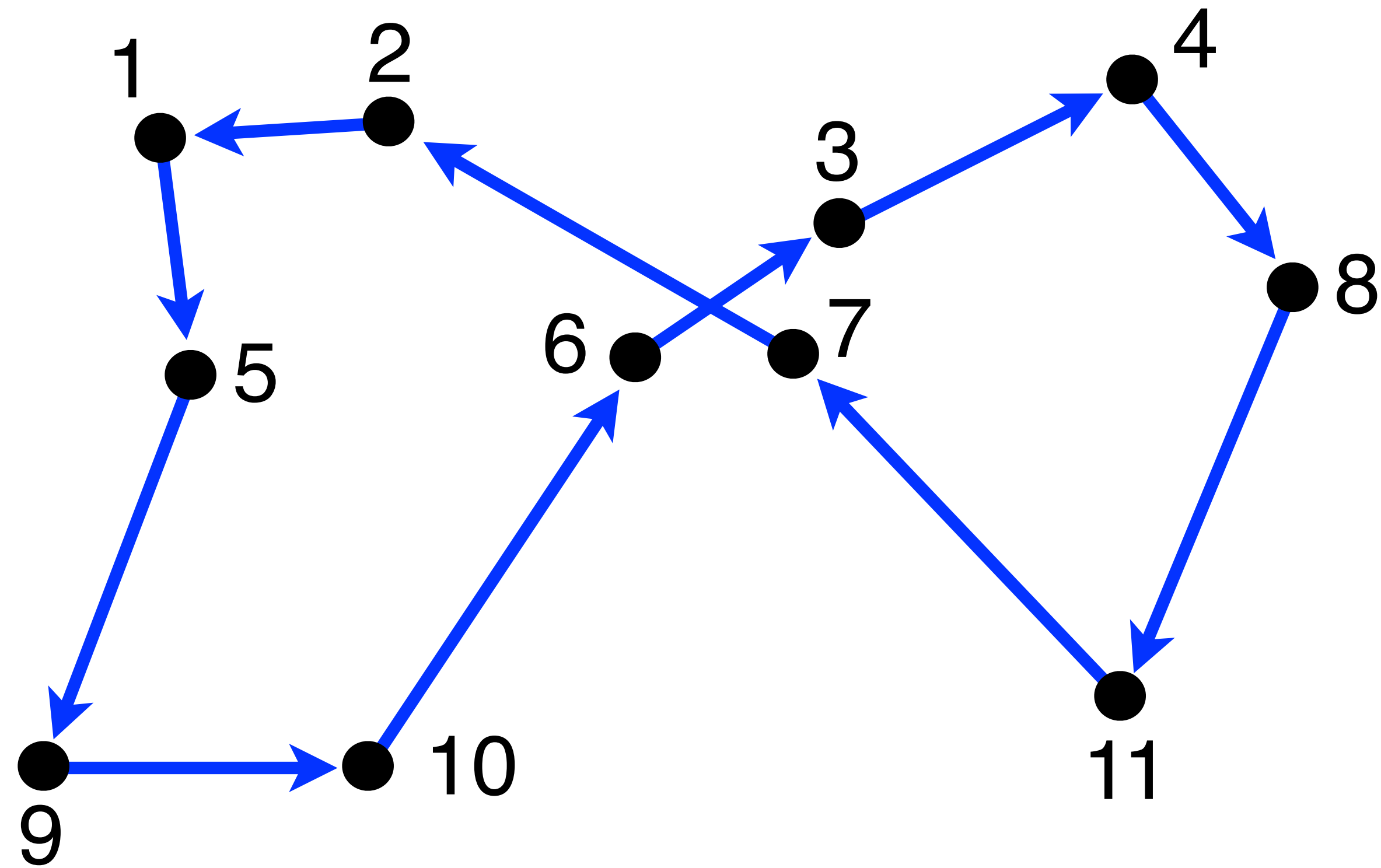
2-OPT



2-OPT

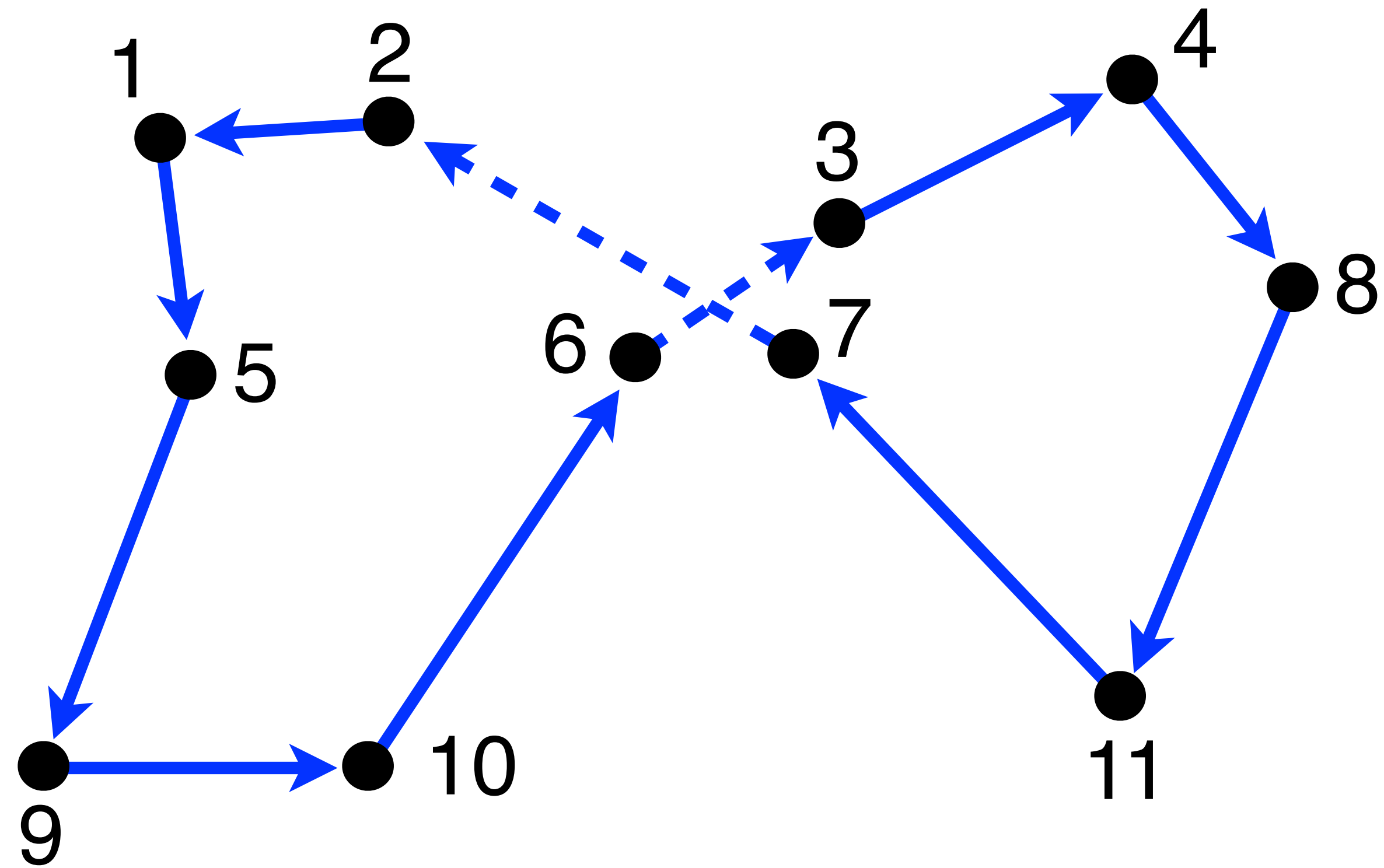


2-OPT



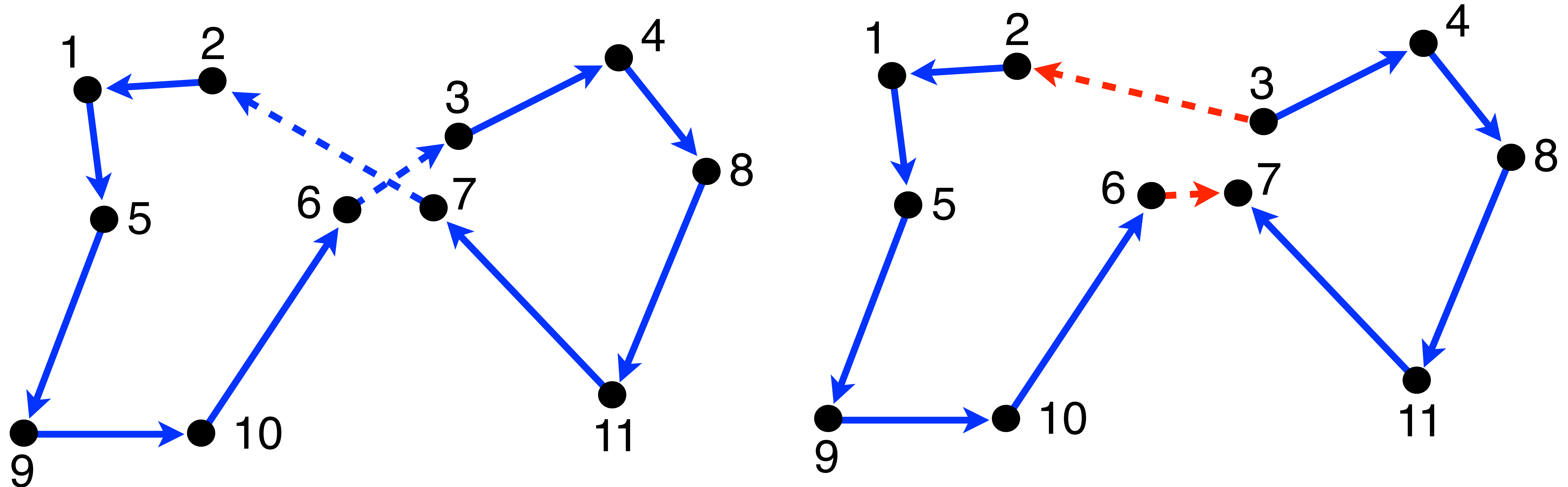
1 - 5 - 9 - 10 - 6 - 3 - 4 - 8 - 11 - 7 - 2

2-OPT



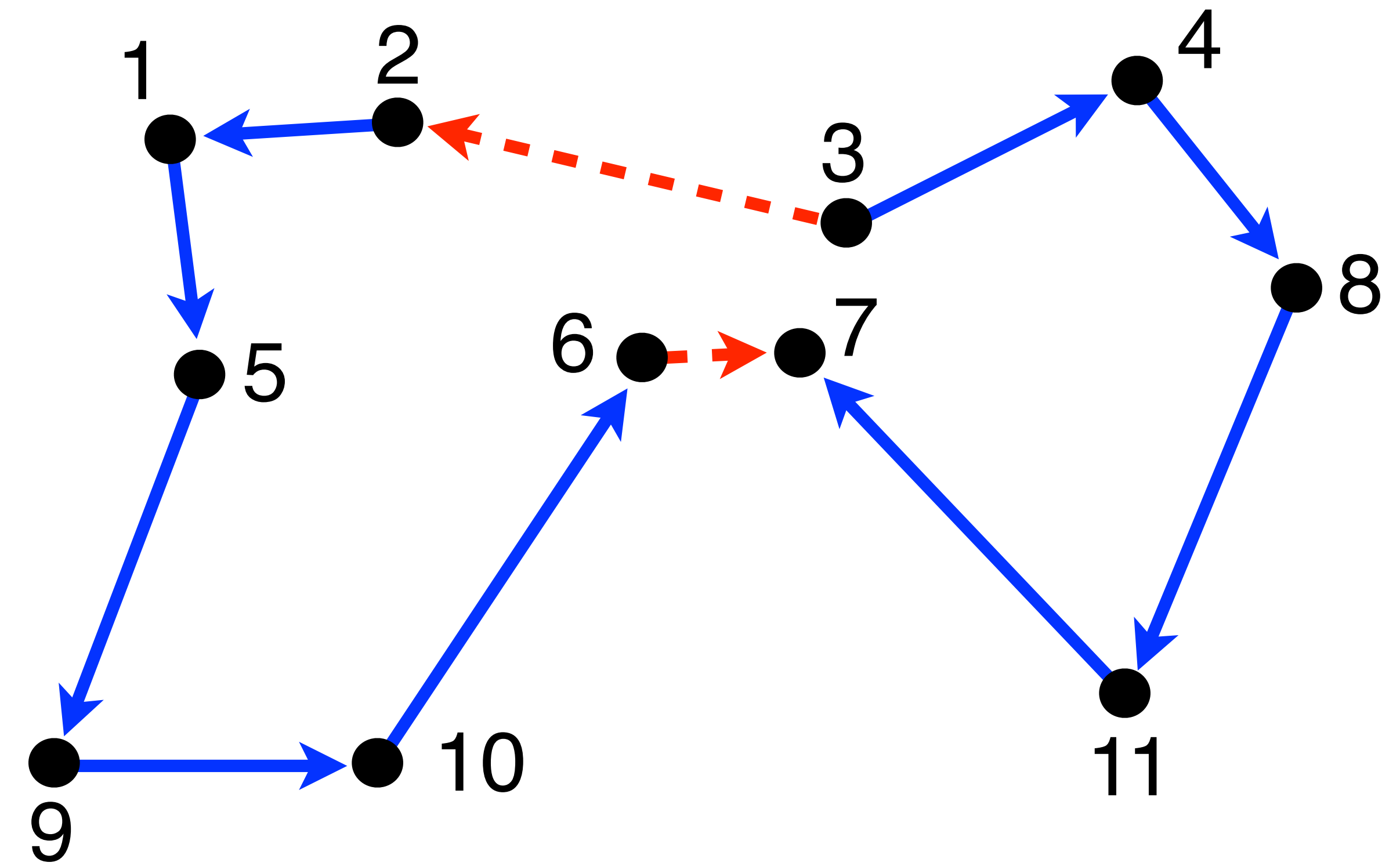
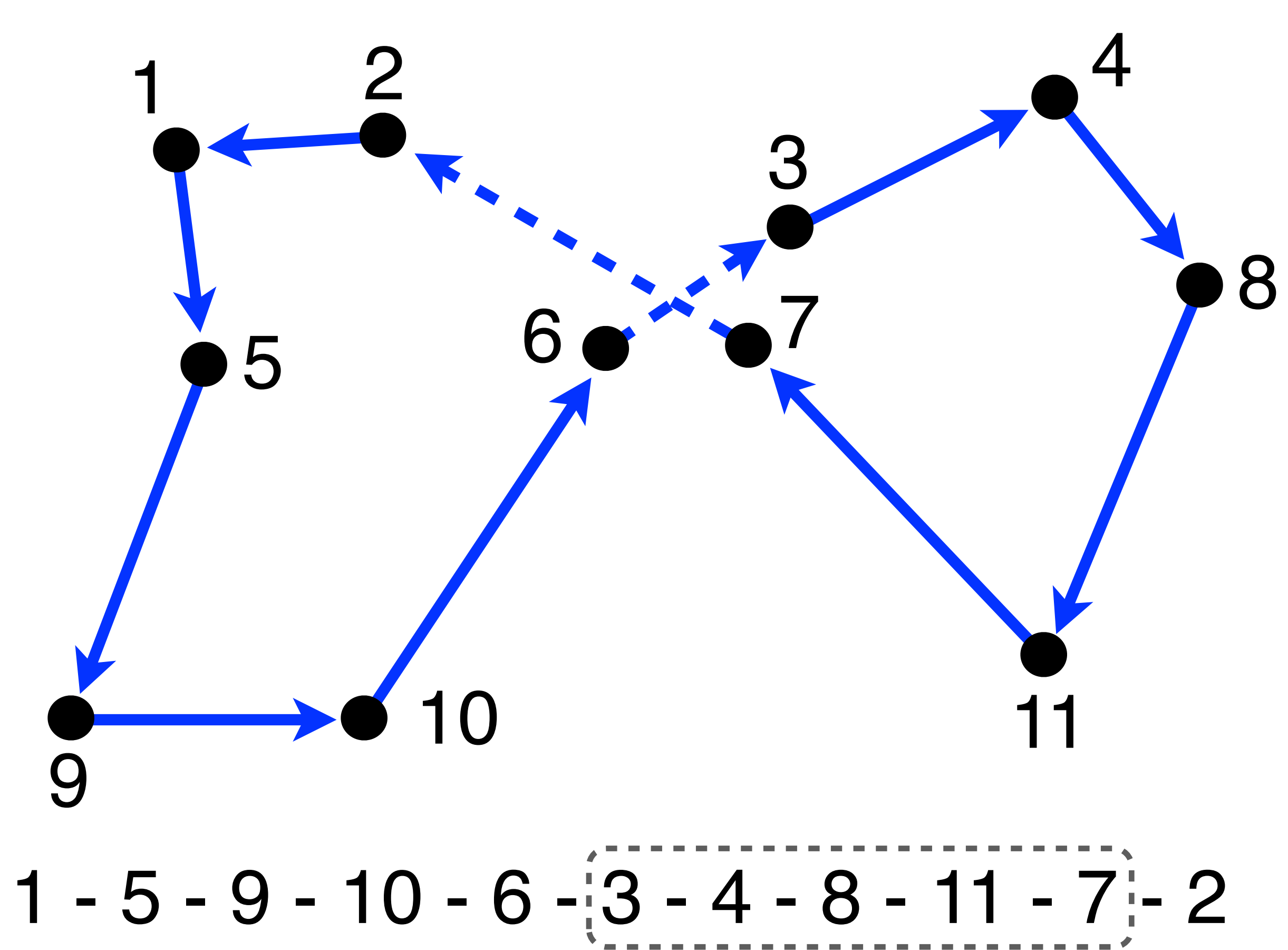
1 - 5 - 9 - 10 - 6 - 3 - 4 - 8 - 11 - 7 - 2

2-OPT

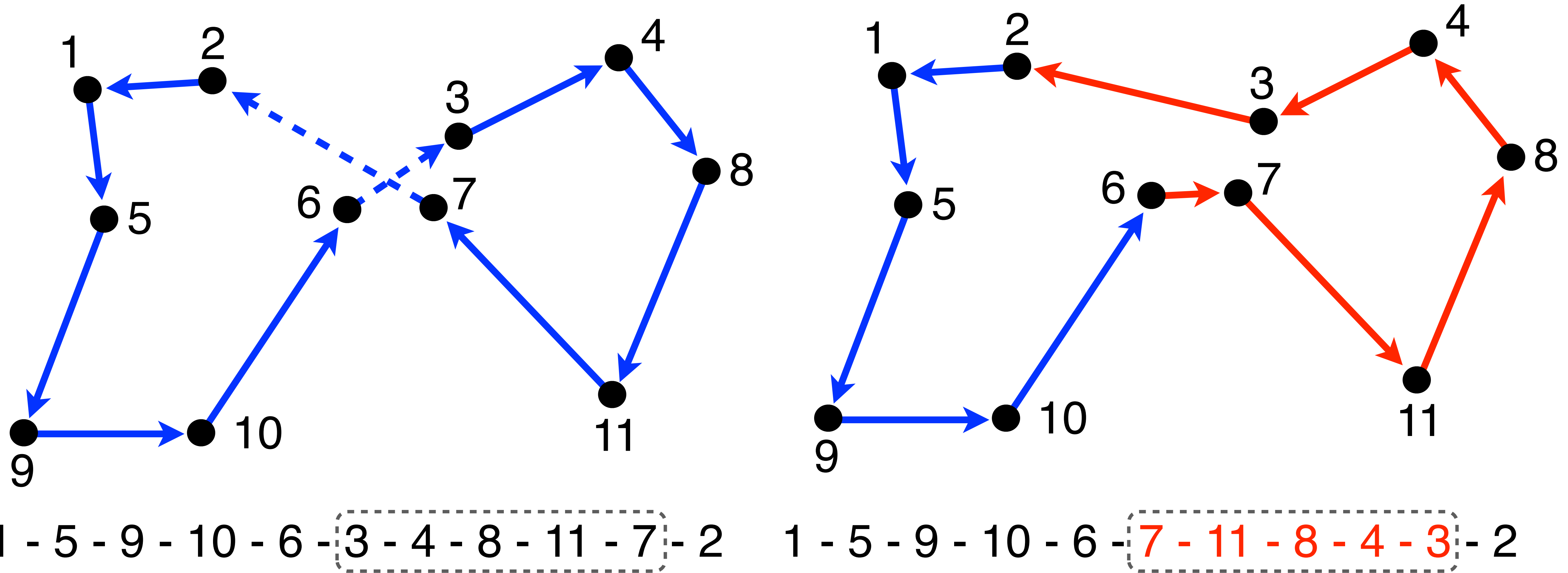


1 - 5 - 9 - 10 - 6 - 3 - 4 - 8 - 11 - 7 - 2

2-OPT



2-OPT



Is 2-OPT Connected?

Is 2-OPT Connected?

- The neighborhood is connected
 - simple algorithm

```
T := some tour
for(int i = 1; i <= n; i++)
    if (Ti != Oi)
        find Si, ..., Si+k such that Si+k = Oi
        S := (S1, ..., Si-1, Si+k, Si+k-1, ..., Si, Si+k+1, ..., Sn)
```

Is the TPP Neighborhood Connected?

Is the TPP Neighborhood Connected?

- ▶ Call me if you have the proof

Until Next Time