LOCAL SEARCH AND OPTIMIZATION

Yeni Herdiyeni

Departemen Ilmu Komputer FMIPA IPB

Pelatnas 3 TOKI, 30 April 2015

Department of Computer Science
Bogor Agricultural University

Review Teknik Pencarian

1. *Blind Search* - Pencarian tanpa informasi (*uninformed search*): DFS, BFS, dll
2. *Heuristic Search* - Pencarian dengan informasi (*informed search*): Greedy Best First , A*
3. Pencarian dengan optimasi (*local search & optimization*) : Hill Climbing, Simulated Annealing, GA
4. Pencarian dengan informasi status lawan (*adversarial search*)
5. Pencarian dengan batasan kondisi (*constraint satisfaction problems*)

Review Teknik Pencarian

- *Uninformed* and *Informed Search*: mencari jalur (*path*) status solusi dari *initial state* sampai *goal state*.

  path: initial state…. State antara … goal state

- *Local search*: hanya membutuhkan state yang memenuhi kondisi final. Path-nya tidak penting

Solusi problem 8-queen = posisi 8 bidak dengan jumlah bidak tidak saling menyerang minimal

- Solusi adalah konfigurasi akhir 8 bidak
- Tidak perlu tahu urutan bidak yang diletakkan di papan

Konsep Dasar Local Search

- Lakukan algoritme *random search*

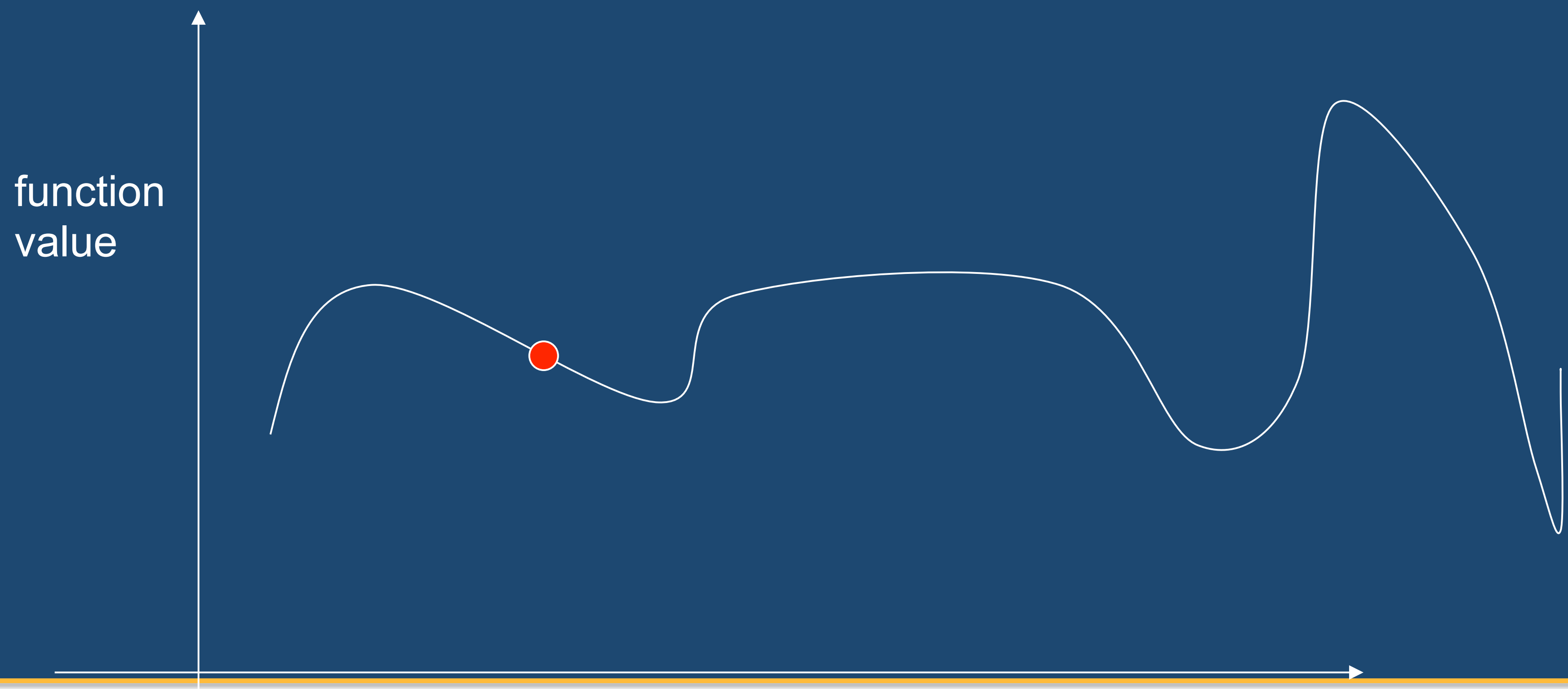
1. Pilih secara *random* suatu state (initial state) dan mulai mencari solusi dari state terdekat
2. Lakukan modifikasi terhadap *current state*.
3. Ulangi langkah 2 sampai goal ditemukan (atau waktu habis).

Algoritme Local Search & Optimization

- Hill Climbing Search
  - Pemilihan state berdasarkan nilai objectivenya
- Genetic Algorithm
  - Pemilihan state berdasarkan aturan seleksi alam yang diterapkan pada state collection (populasi)

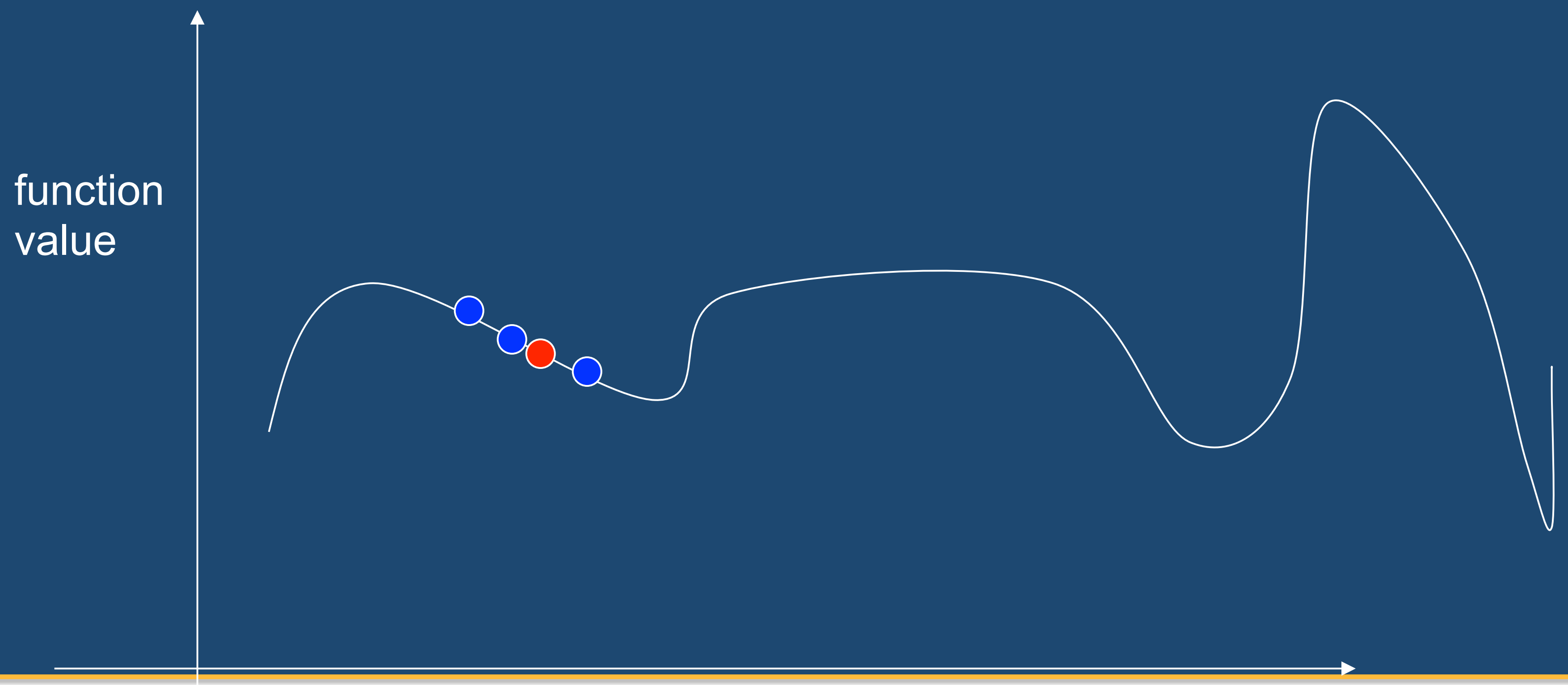
Simple Example

- Random Starting Point



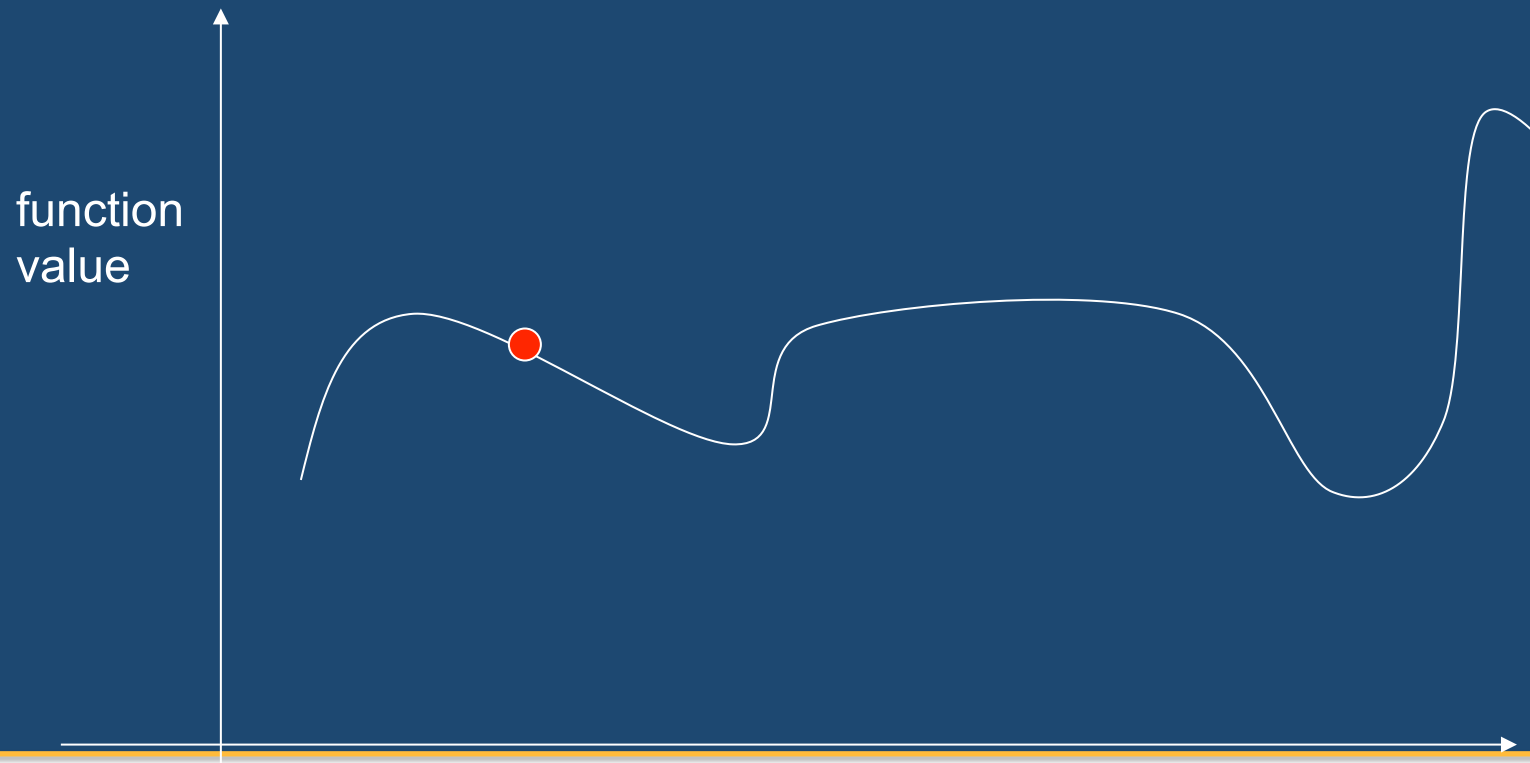
Simple Example

- Three random steps



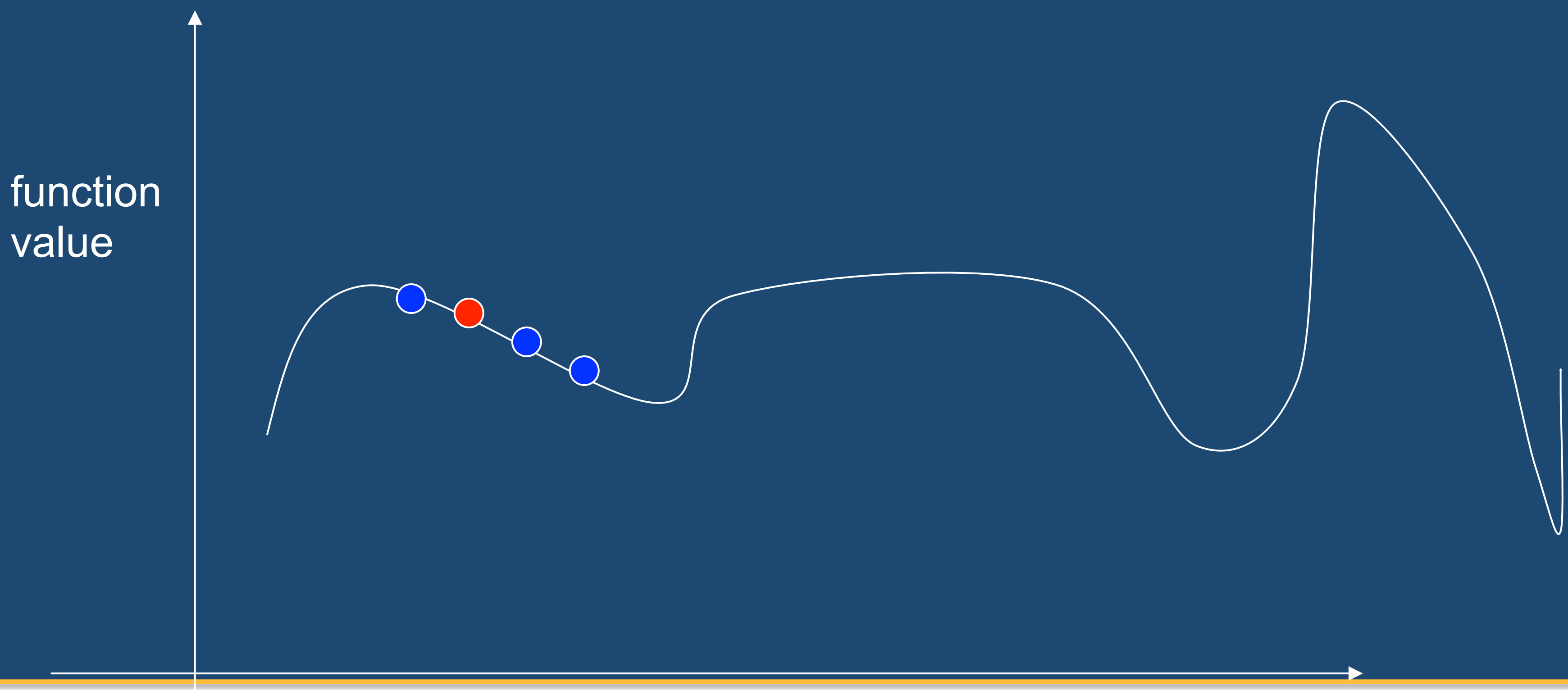
Simple Example

- Choose Best One for new position



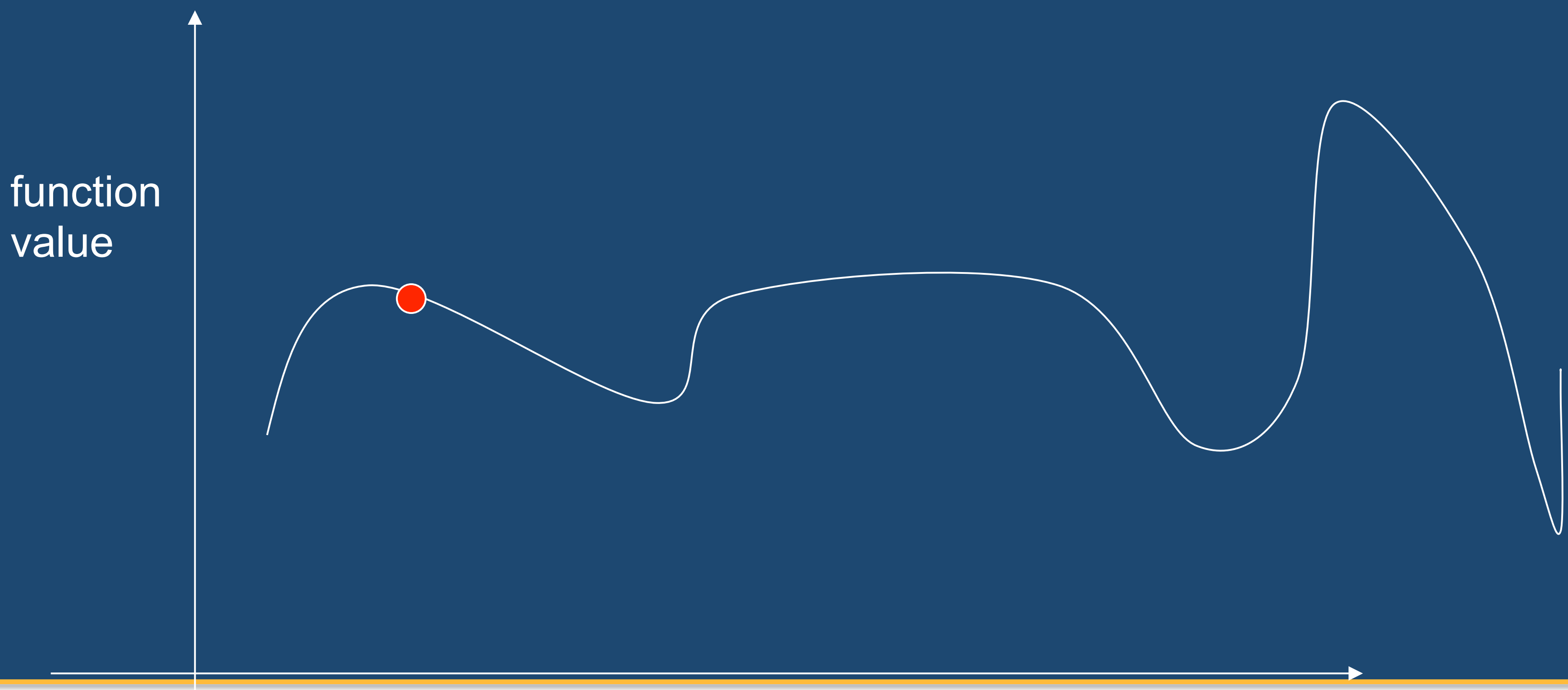
Simple Example

- Repeat



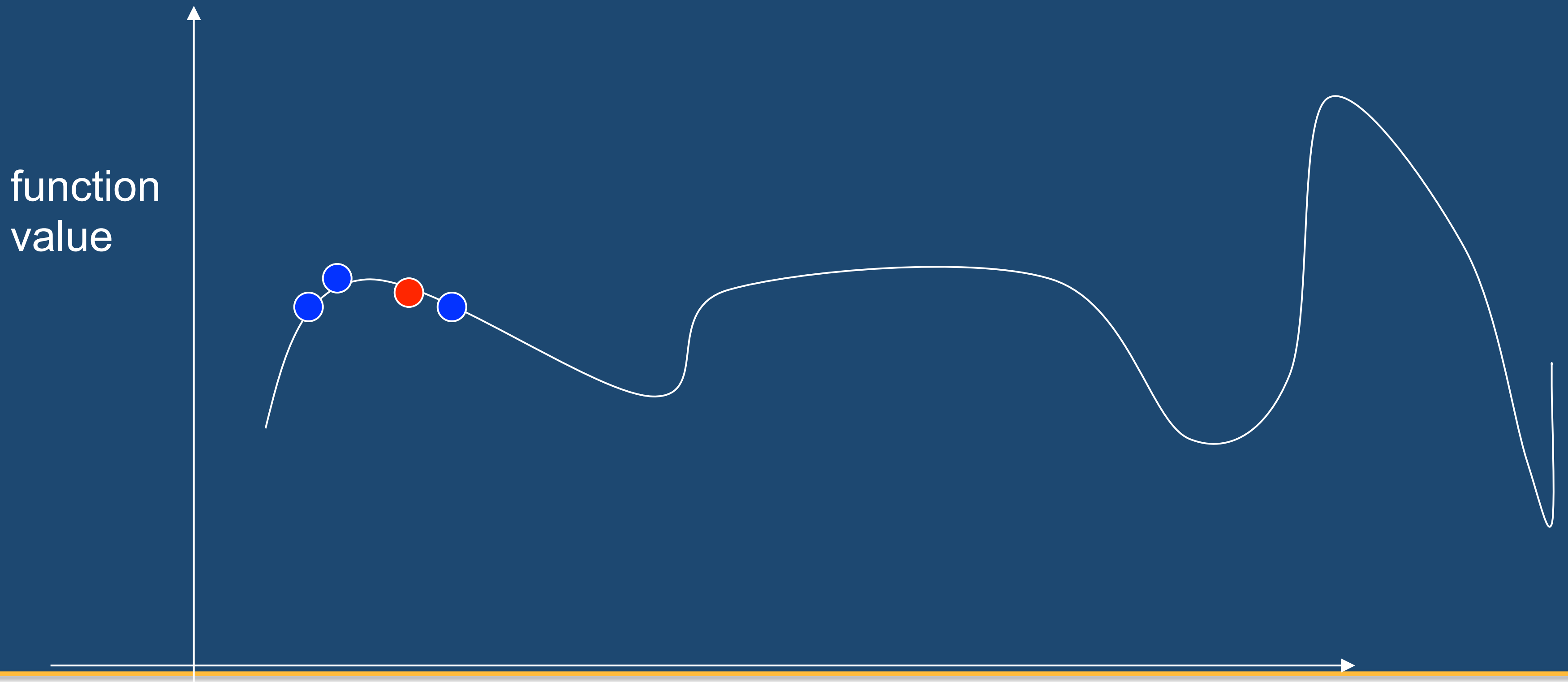
Simple Example

- Repeat



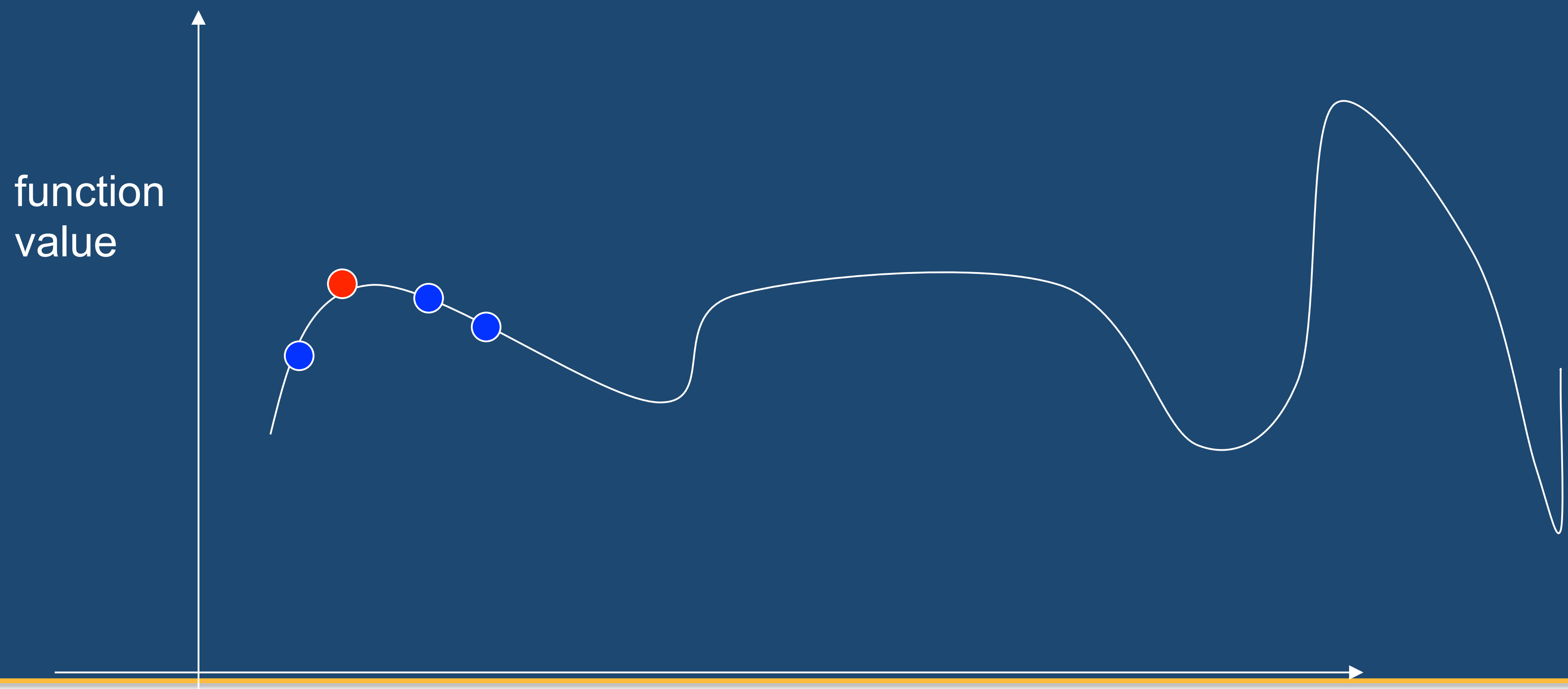
Simple Example

- Repeat



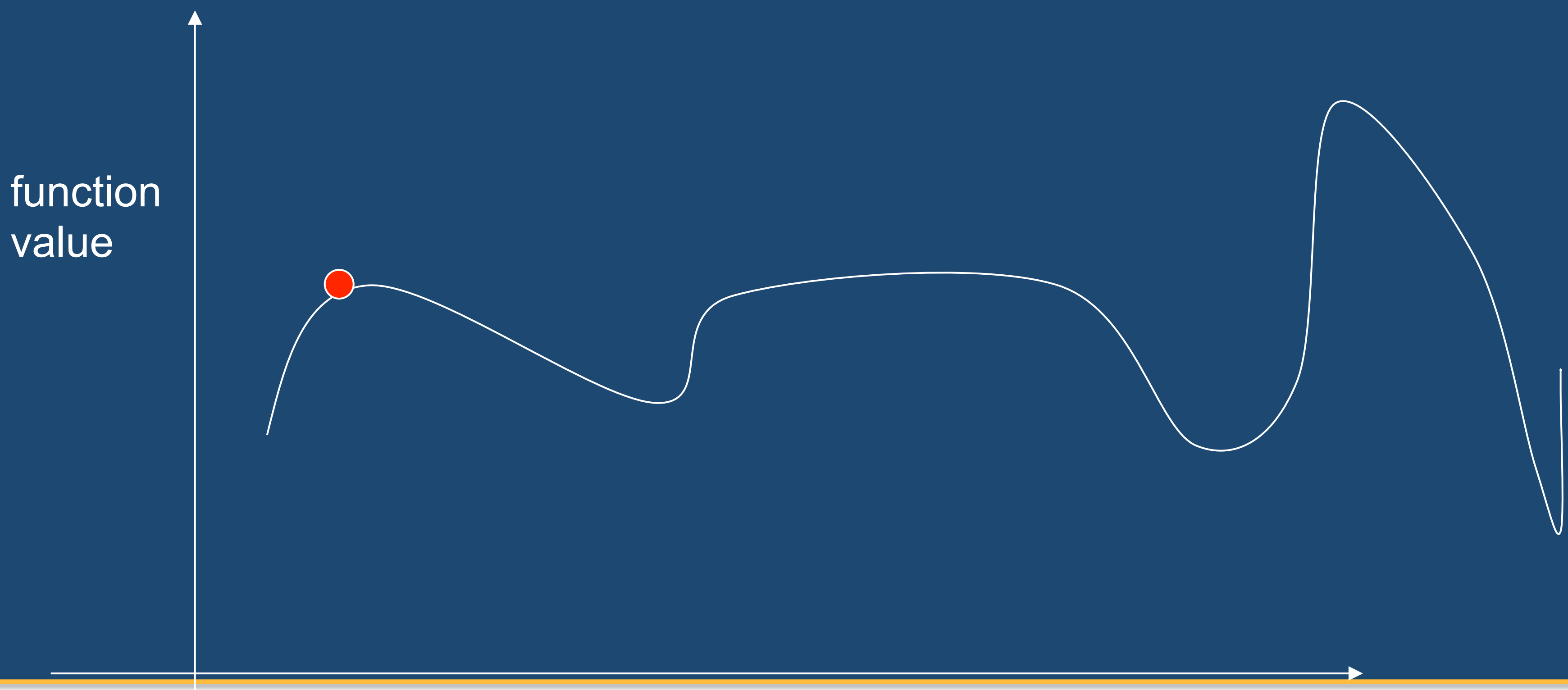
Simple Example

- Repeat



Simple Example

- No Improvement, so stop.



Algoritma 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
   end
```

- Nilai sebuah node → h(n) (*heuristic function*)
- Bayangkan seorang penderita amnesia mendaki gunung dengan kabut tebal…
  - State: posisi koordinat (X,Y)
  - H(n) : Ketinggian pendaki
- Disebut juga dengan Greedy Local Search

Persiapan untuk Hill-Climbing Search

- Menentukan initial state (secara acak)
- Fungsi Objektif untuk hitung nilai state

Contoh:

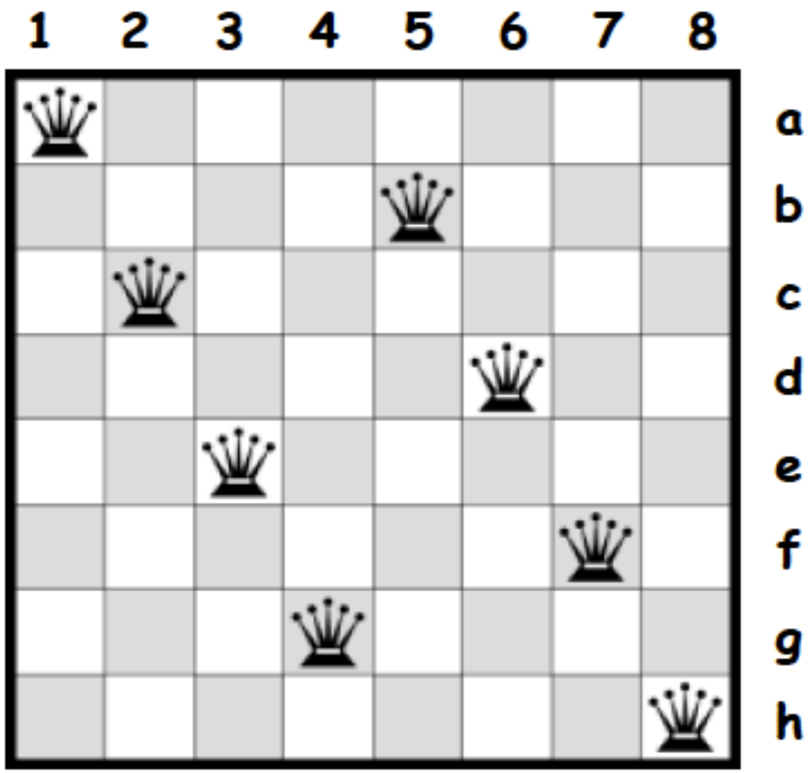
Solusi problem 8-queen = posisi 8 bidak dengan jumlah bidak tidak saling menyerang minimal

- Solusi adalah konfigurasi akhir 8 bidak
- Tidak perlu tahu urutan bidak yang diletakkan di papan

Problem: 8-queens

- Pilih *initial state*
  - Posisi bidak random dari 1-8; posisi teratas
- Fungsi Objektif
  - *Heuristic cost function h* = jumlah pasangan bidak ratu yang dapat saling menyerang

Contoh *Heuristic Cost Function h* untuk Problem 8-queens



h = 1 pasangan bidak ratu yang dapat saling menyerang

Yaitu 1a-8h

Contoh 8-queens dengan Hill-Climbing

Secara random, state current = 1e 2f 3g 4d 5e 6f 7g 8f

Nilai h(current)=17 pasangan saling menyerang

1e-2f; 1e-3g; 1e-5e; 2f-3g; 2f-4d; 2f-6f; 2f-8f;
3g-5e; 3g-7g; 4d-5e; 4d-6f; 4d-7g; 5e-6f; 5e-7g;
6f-7g; 6f-8f; 7g-8f;

	1	2	3	4	5	6	7	8
a	18	12	14	13	13	12	14	14
b	14	16	13	15	12	14	12	16
c	14	12	18	13	15	12	14	14
d	15	14	14	♛	13	16	13	16
e	♛	14	17	15	♛	14	16	16
f	17	♛	16	18	15	♛	15	♛
g	18	14	♛	15	15	14	♛	16
h	14	14	13	17	12	14	12	18

Hitung semua nilai h jika posisi satu bidak catur dirubah

Misal 1: posisi 1e dirubah ke 1d maka state 1d 2f 3g 4d 5e 6f 7g 8f; h=15

Misal 2: posisi 2f dirubah ke 2e maka state 1e 2e 3g 4d 5e 6f 7g 8f; h=14

Nilai h terkecil adalah h=12, jadi Kondisi state dapat diubah ke:

1e 2a 3g 4d 5e 6f 7g 8f … … … …

1e 2f 3g 4d 5e 6f 7h 8f (8 pilihan)

Aturan Hill-Climbing, pilihan state selanjutnya nilai h <= 12

Misal state current = 1e 2a 3g 4d 5e 6f 7g 8f

Contoh 8-queens dengan Hill-Climbing

Secara random, state current = 1e 2f 3g 4d 5e 6f 7g 8f

Nilai h(current)=17 pasangan saling menyerang

1e-2f; 1e-3g; 1e-5e; 2f-3g; 2f-4d; 2f-6f; 2f-8f;
3g-5e; 3g-7g; 4d-5e; 4d-6f; 4d-7g; 5e-6f; 5e-7g;
6f-7g; 6f-8f; 7g-8f;

1	2	3	4	5	6	7	8	
18	12	14	13	13	12	14	14	a
14	16	13	15	12	14	12	16	b
14	12	18	13	15	12	14	14	c
15	14	14	♛	13	16	13	16	d
♛	14	17	15	♛	14	16	16	e
17	♛	16	18	15	♛	15	♛	f
18	14	♛	15	15	14	♛	16	g
14	14	13	17	12	14	12	18	h

Hitung semua nilai h jika posisi satu bidak catur dirubah

Misal 1: posisi 1e dirubah ke 1d maka state 1d 2f 3g 4d 5e 6f 7g 8f; h=15

Misal 2: posisi 2f dirubah ke 2e maka state 1e 2e 3g 4d 5e 6f 7g 8f; h=14

Nilai h terkecil adalah h=12, jadi Kondisi state dapat diubah ke:

1e 2a 3g 4d 5e 6f 7g 8f … … … …

1e 2f 3g 4d 5e 6f 7h 8f (8 pilihan)

Aturan Hill-Climbing, pilihan state selanjutnya nilai h <= 12

Misal state current = 1e 2a 3g 4d 5e 6f 7g 8f

Contoh Hill Climbing untuk Cari Jalur

	M	C	W	E	S
M	0	.9	.6	.8	.7
C		0	1.3	1.5	1.3
W			0	.2	.3
E				0	.2
S					0

- Cari jalur dengan jarak terpendek!
- Nilai h = total jarak

Contoh Hill-Climbing untuk Cari Jalur

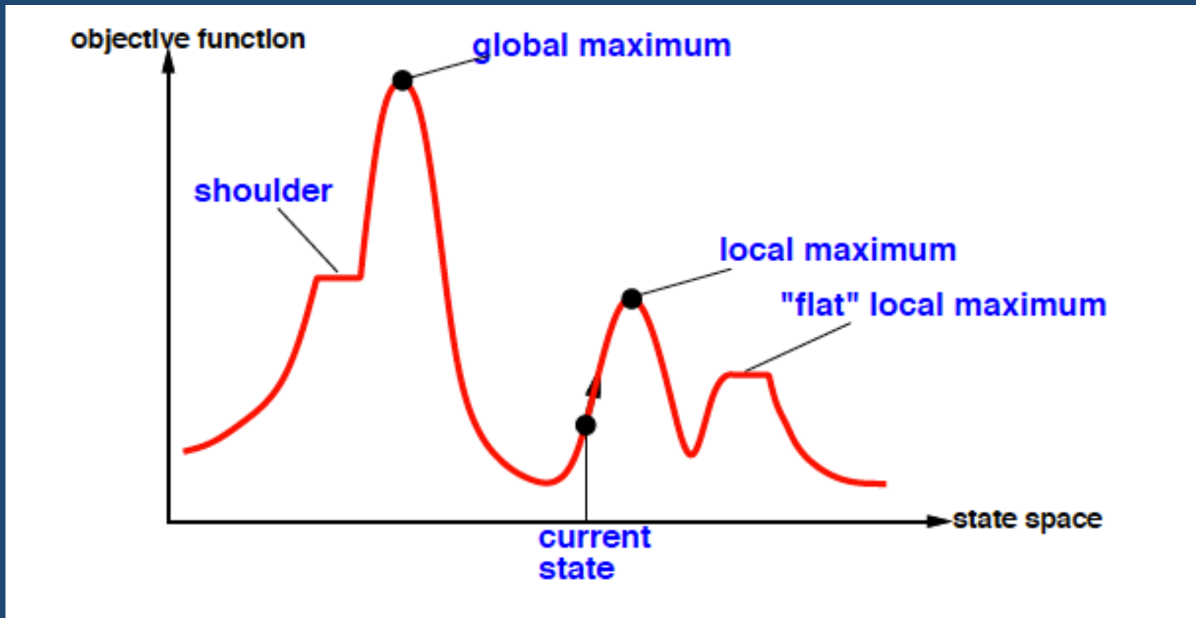
- Initial state = (M -> E -> C -> S -> W) secara random
- Nilai h = .8 + 1.5 + 1.3 + .3 = 3.9
- Contoh proses swap: (A->B->C)
  - Tukar A-B didapat (B->A->C)
  - Tukar A-C didapat (C->B->A)
  - Tukar B-C didapat (A->C->B)
- Neighbor terbentuk dengan merubah (*swap*) 2 lokasi

(E -> M -> C -> S -> W) = 3.3	(C -> E -> M -> S -> W) = 3.3
(S -> E -> C -> M -> W) = 3.2	(W -> E -> C -> S -> M) = 3.7
(M -> C -> E -> S -> W) = 2.9	(M -> S -> C -> E -> W) = 3.7
(M -> W -> C -> S -> E) = 3.4	(M -> E -> S -> C -> W) = 3.6
(M -> E -> W -> S -> C) = 2.6	(M -> E -> C -> W -> S) = 3.9

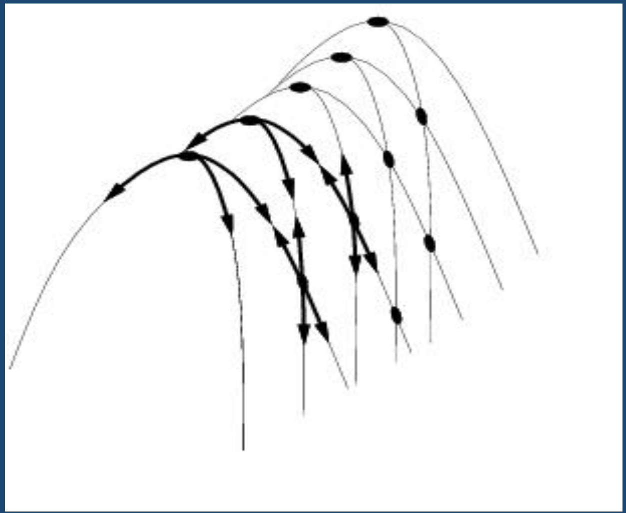
- Hill-Climbing akan memilih h=2.6
  - untuk current state (M -> E -> W -> S -> C)
- Proses swap dilakukan 1x lagi, dan solusi optimal didapat
  - (S->E->W->M->C) dengan h=1.9

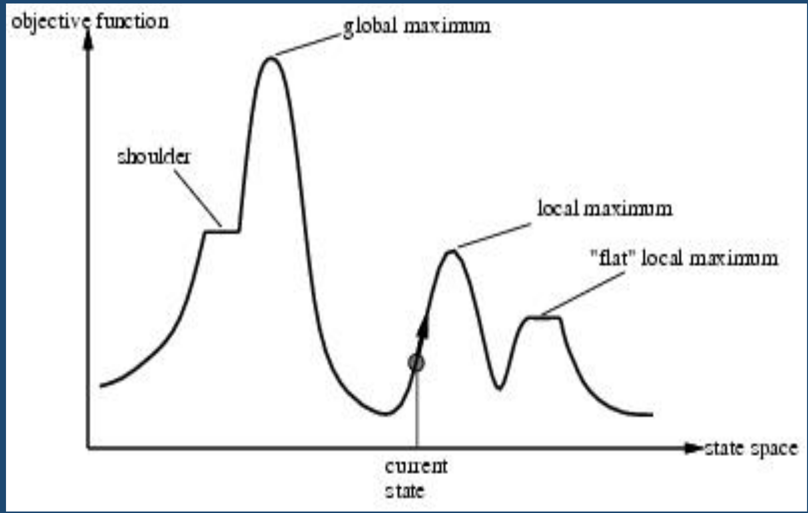
Masalah Hill Climbing

- Tergantung pada pilihan initial state, hill climbing bisa terperangkap dalam local maximum



Masalah Hill Climbing





- Local Maximum : Tidak tetangga yang lebih baik, tetapi bukan solusi optimal (ridge = a sequence of local maxima)
- Plateau: Semua tetangga sama baiknya.

Solusi Masalah Hill Climbing

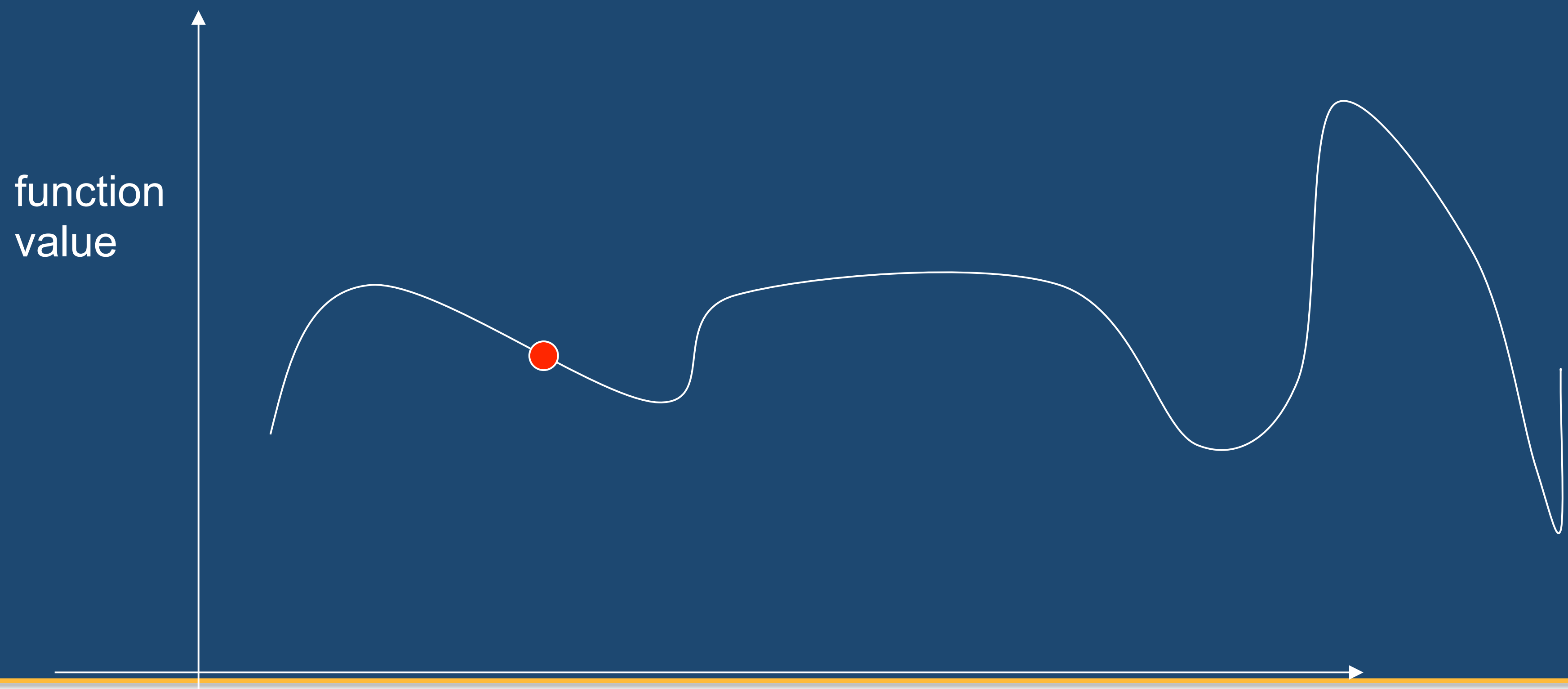
- Gradient Descent/Ascent (Random-restart Hill Climbing)
- Simulated Annealing
- Stochastic Beam Search → GA

Gradient Descent (or Ascent)

- Simple modification to Hill Climbing
  - Generallly assumes a continuous state space
- Idea is to take more intelligent steps
- Look at local gradient: the direction of largest change
- Take step in that direction
  - Step size should be proportional to gradient
- Tends to yield much faster convergence to maximum

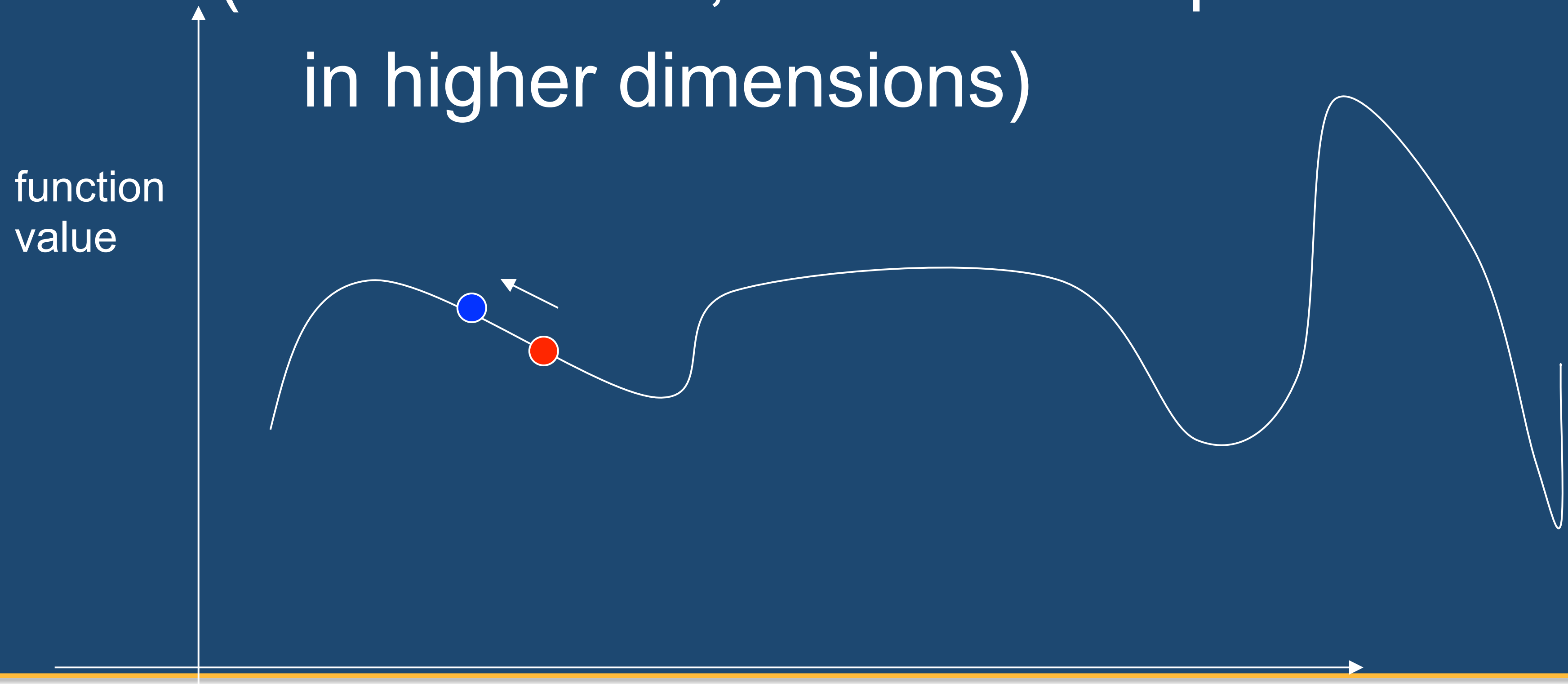
Gradient Ascent

- Random Starting Point



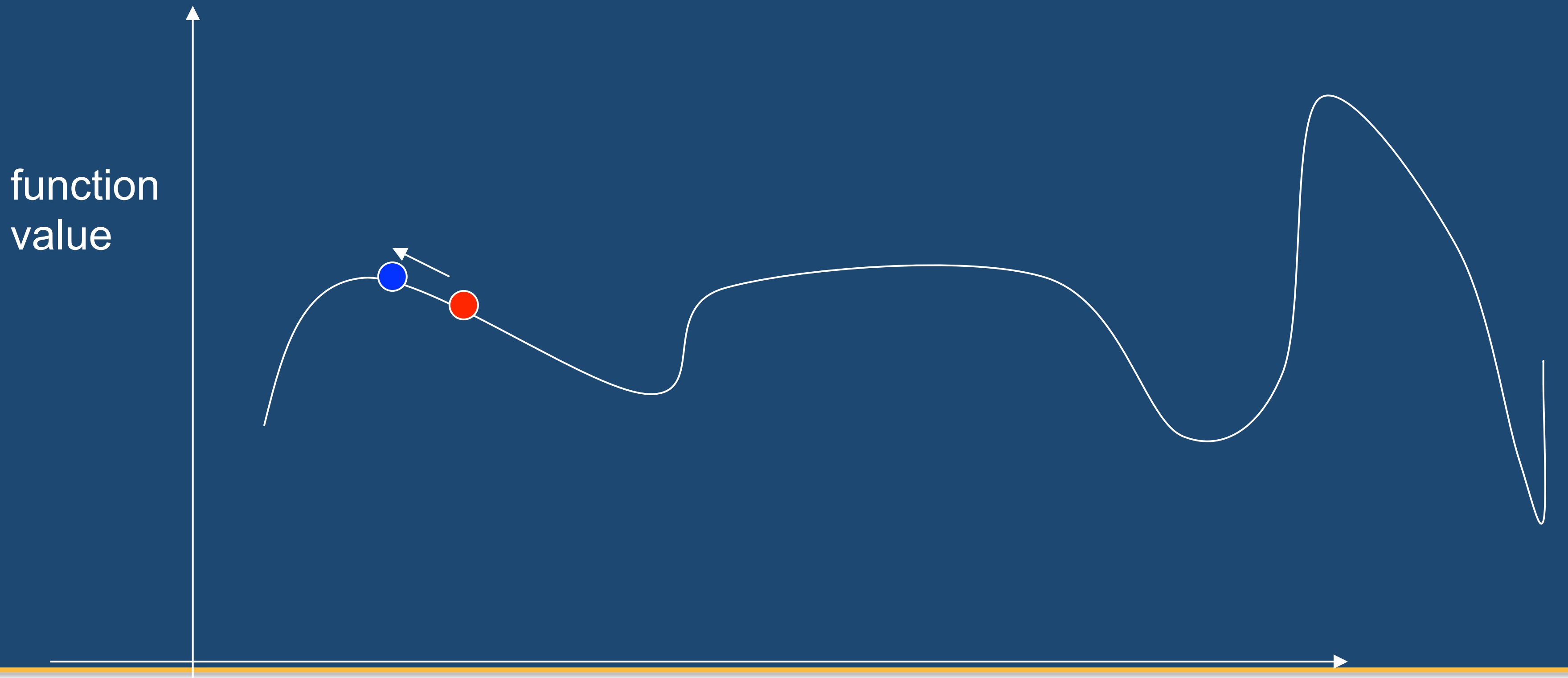
Gradient Ascent

- Take step in direction of largest increase
  (obvious in 1D, must be computed
  in higher dimensions)



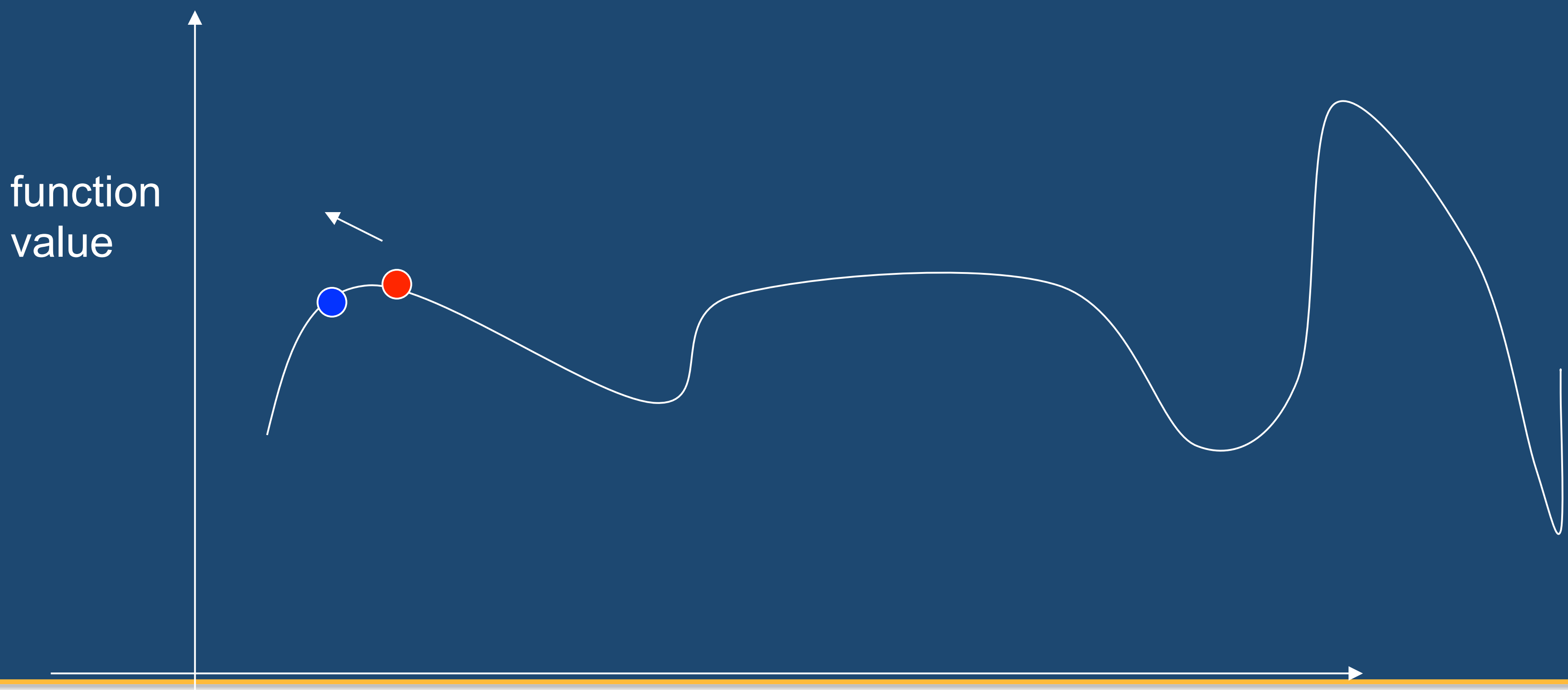
Gradient Ascent

- Repeat



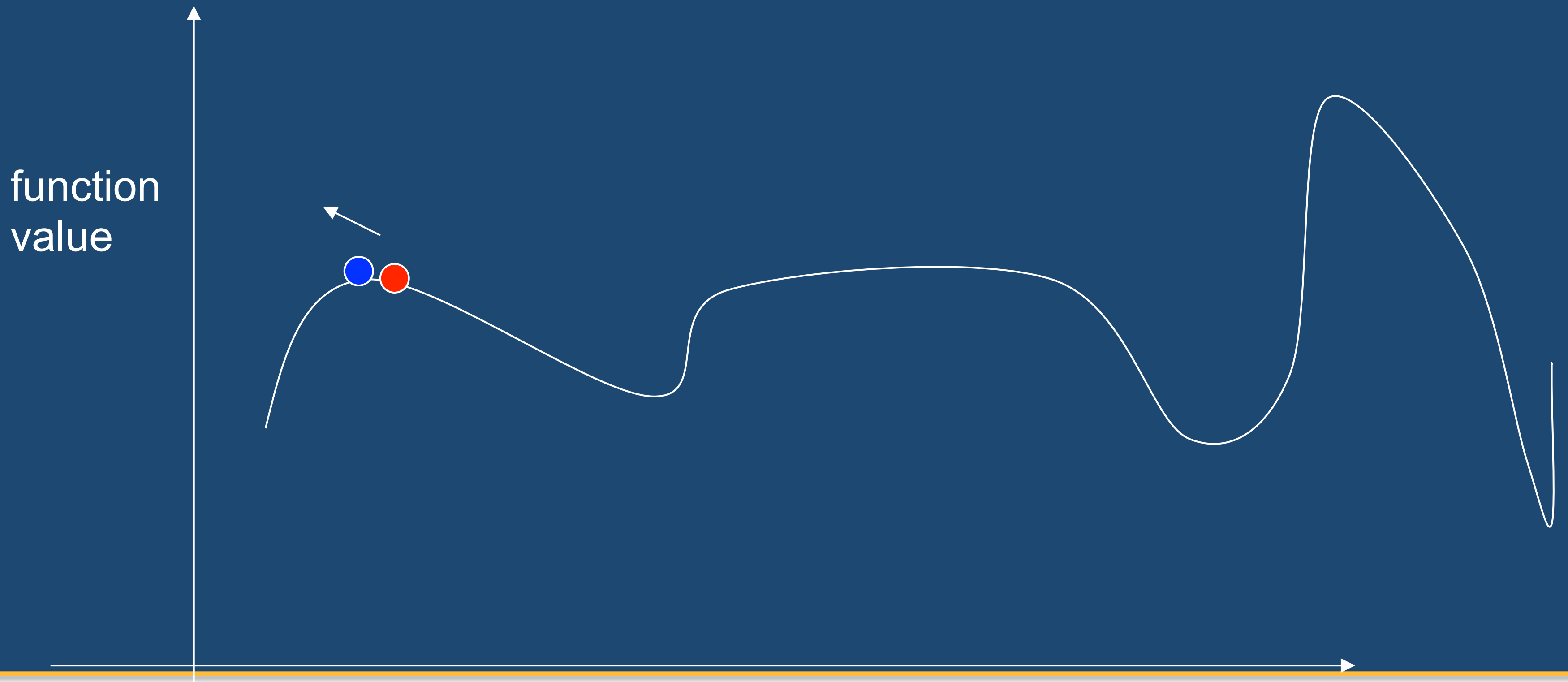
Gradient Ascent

- Next step is actually lower, so stop



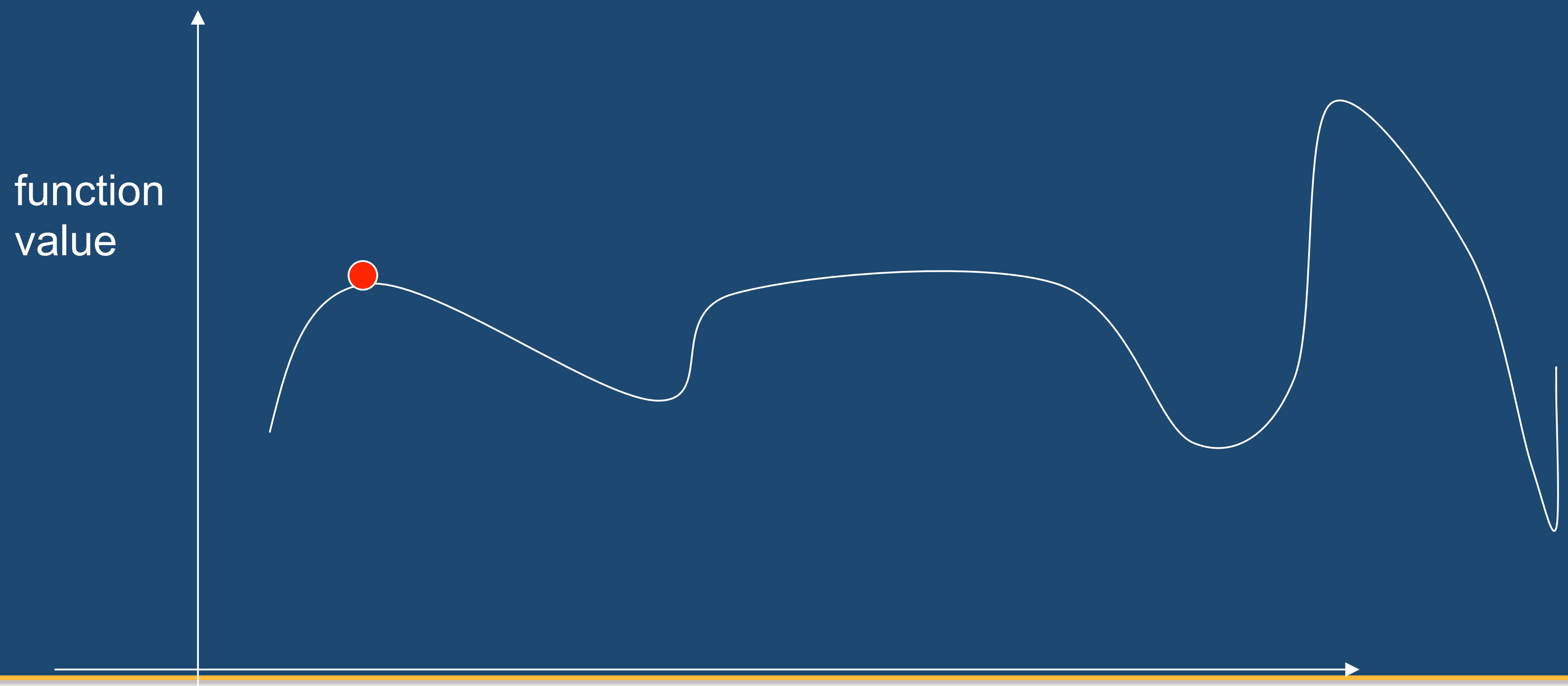
Gradient Ascent

- Could reduce step size to "hone in"



Gradient Ascent

- Converge to (local) maximum



Simulated Annealing

- salah satu algoritma untuk untuk optimisasi yang berbasiskan probabilitas dan mekanika statistik
- algoritma ini dapat digunakan untuk mencari pendekatan terhadap solusi optimum global dari suatu permasalahan

Simulated Annealing

- Annealing adalah satu teknik yang dikenal dalam bidang metalurgi, digunakan dalam mempelajari proses pembentukan kristal dalam suatu materi.
- Agar dapat terbentuk susunan kristal yang sempurna, diperlukan pemanasan sampai suatu tingkat tertentu, kemudian dilanjutkan dengan pendinginan yang perlahan-lahan dan terkendali dari materi tersebut.
- Pemanasan materi di awal proses annealing, memberikan kesempatan pada atom-atom dalam materi itu untuk bergerak secara bebas
- Proses pendinginan yang perlahan-lahan memungkinkan atom-atom yang tadinya bergerak bebas itu, pada akhirnya menemukan tempat yang optimum, di mana energi internal yang dibutuhkan atom itu untuk mempertahankan posisinya adalah minimum.

Simulated Annealing

- High temperature→ High Disorder → High Energy
- SA differs from hill climbing in that a move is selected at random and then decides whether to accept it
- In SA better moves are always accepted. Worse moves are not

Simulated Annealing

- The probability of accepting a worse state is a function of both the temperature of the system and the change in the cost function
- As the temperature decreases, the probability of accepting worse moves decreases
- If T=0, no worse moves are accepted (i.e. hill climbing)

Simulated Annealing

- Heuristic/goal/fitness function E (energy)
- Generate a move (randomly) and compute
  $\Delta E = E_{new} - E_{old}$
- If $\Delta E <= 0$, then accept the move
- If $\Delta E > 0$, accept the move with probability:
  Set

$$P(\Delta E) = e^{-\frac{\Delta E}{kT}}$$

- T is "Temperature"

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, the temperature

$current \leftarrow$ MAKENODE(RANDOMSTATE[*problem*])

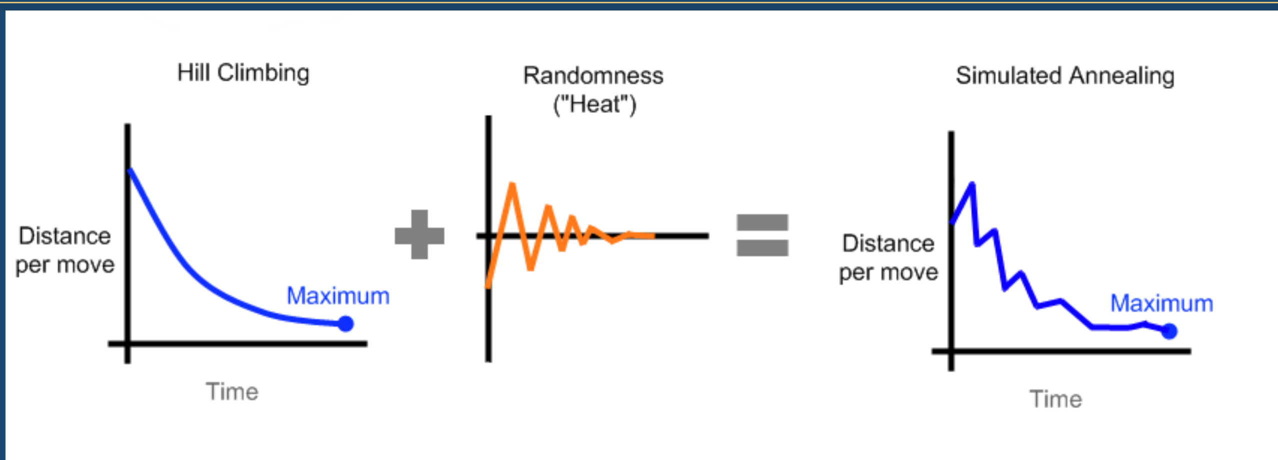
for $t \leftarrow 1$ **to** ∞ **do**
$T \leftarrow schedule[t]$
if $T = 0$ **then return** *current*
next $\leftarrow$ a randomly selected successor of *current*
$\Delta E \leftarrow$ VALUE[*next*] $-$ VALUE[*current*]
if $\Delta E > 0$ **then** $current \leftarrow next$
else $current \leftarrow next$ only with probability $e^{\Delta E / T}$

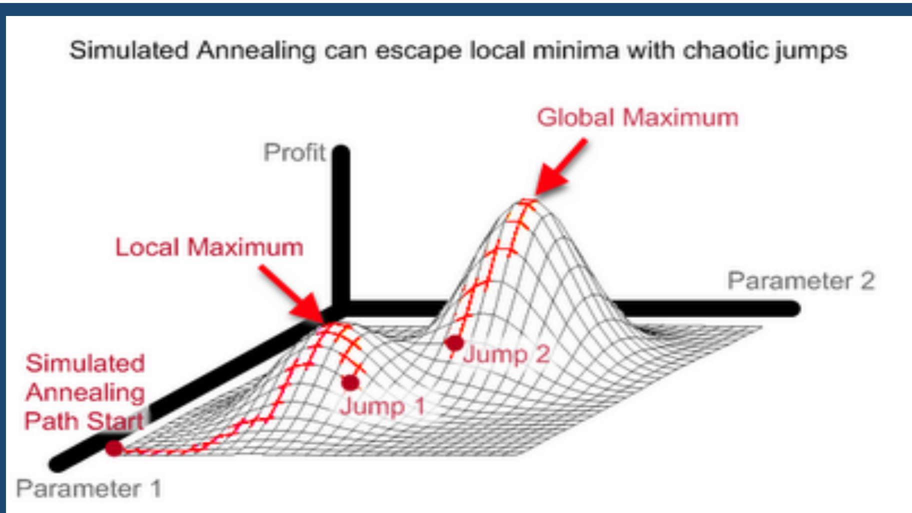
Let us assume that the current and next point in a search space differ by 13 (i.e., $\Delta E = -13$). Then:

$\mathbf{T}$	$\mathbf{e^{\Delta E/T}}$
1	0.000002
5	0.0743
10	0.2725
20	0.52
50	0.77
10^{10}	0.9999...

Thus, at high values of T, simulated annealing behaves like a random walk; at low values of T, it behaves like hill-climbing.

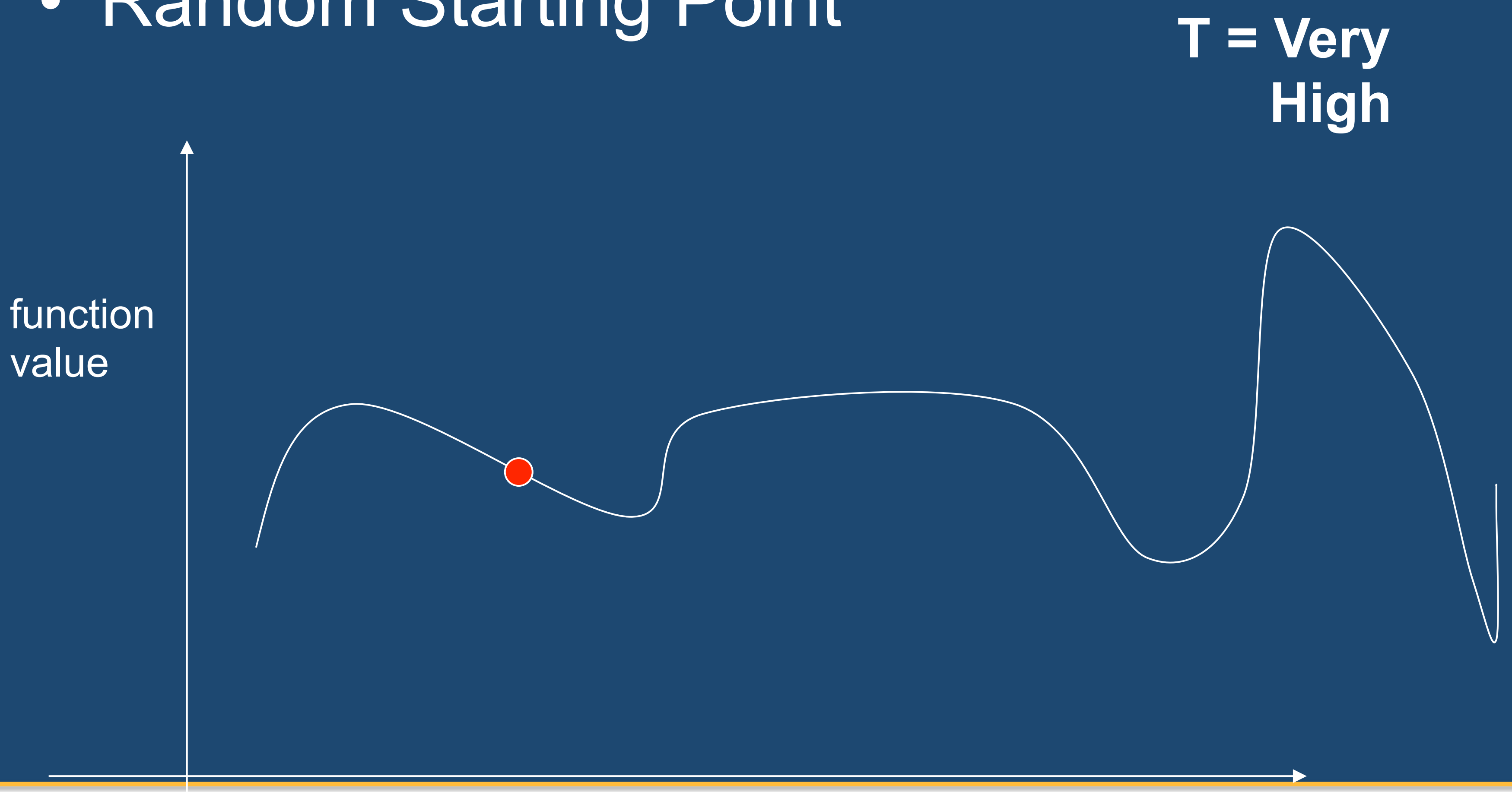
Simulated Annealing





Simulated Annealing

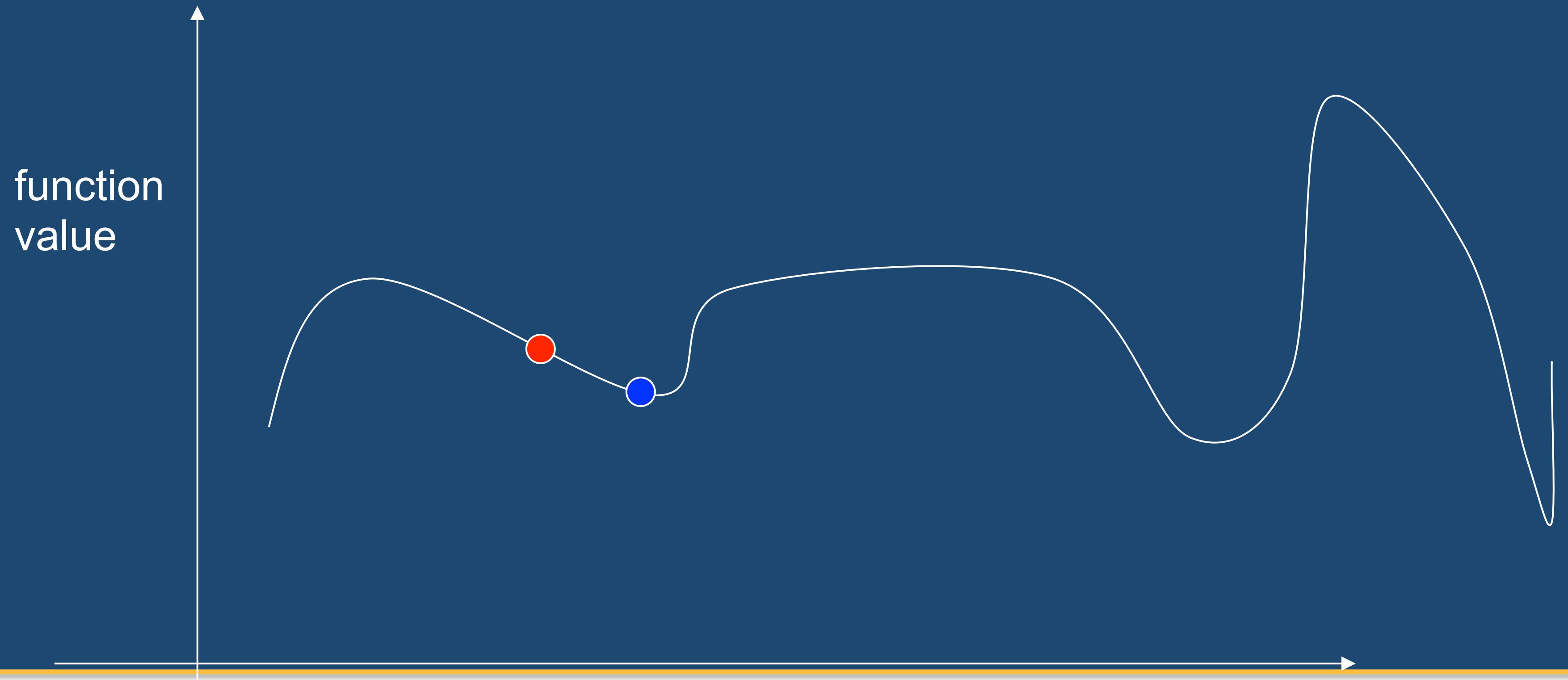
- Random Starting Point



Simulated Annealing

- Random Step

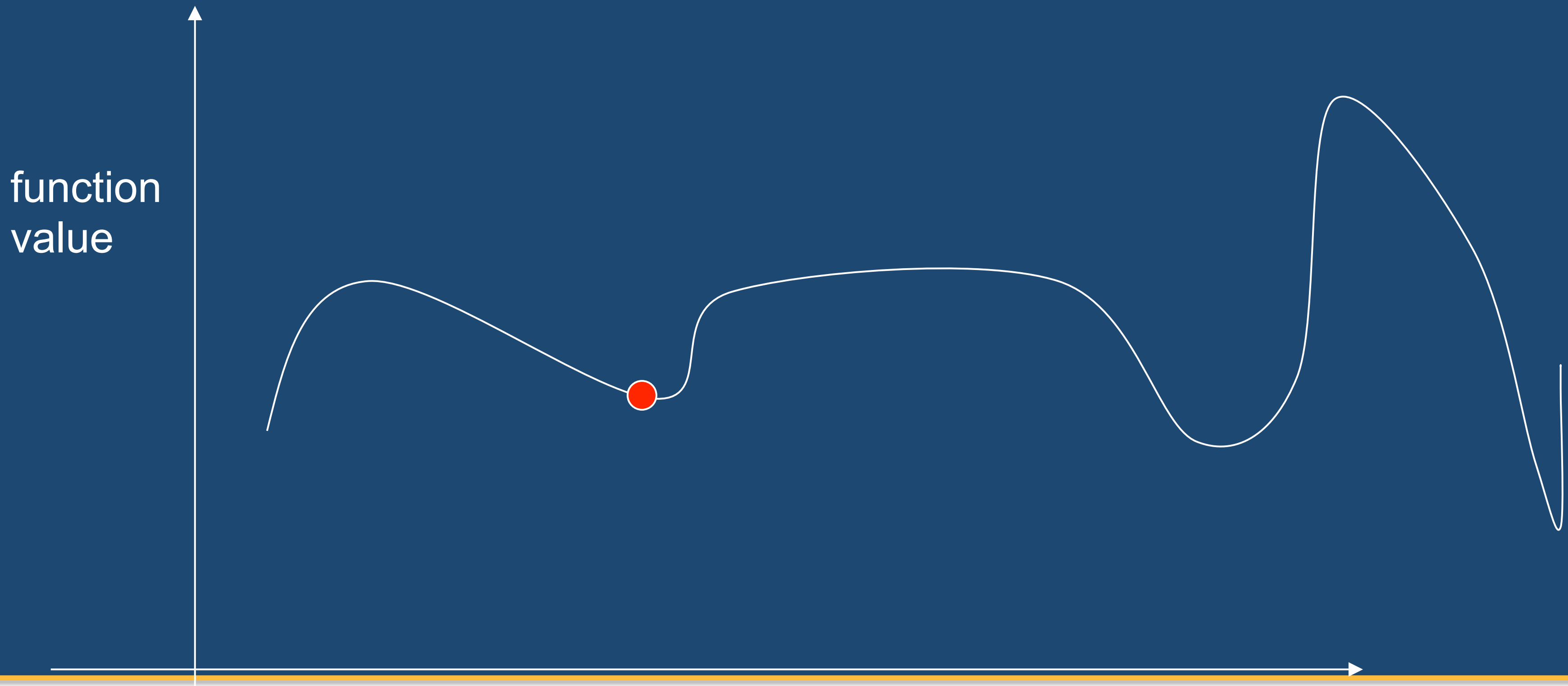
T = Very High



Simulated Annealing

- Even though E is lower, accept

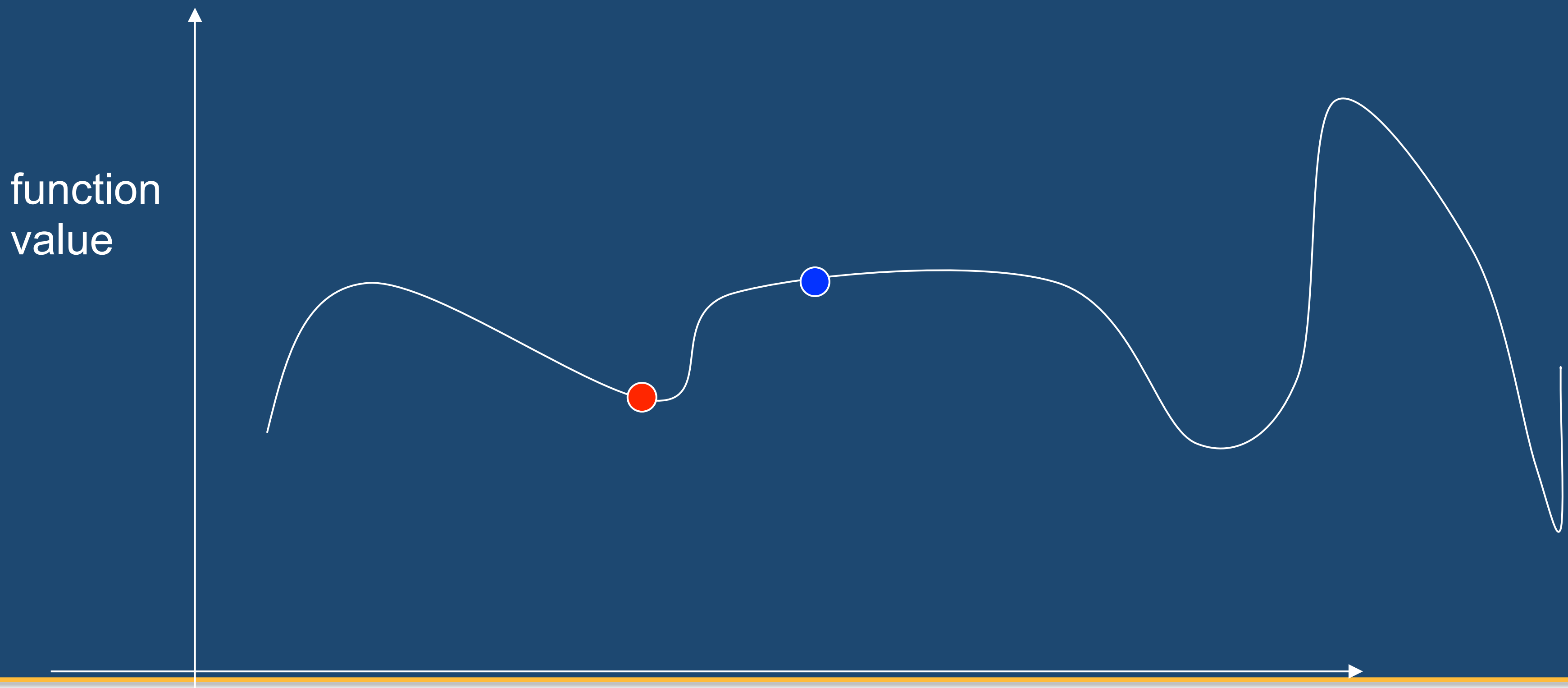
T = Very High



Simulated Annealing

- Next Step; accept since higher E

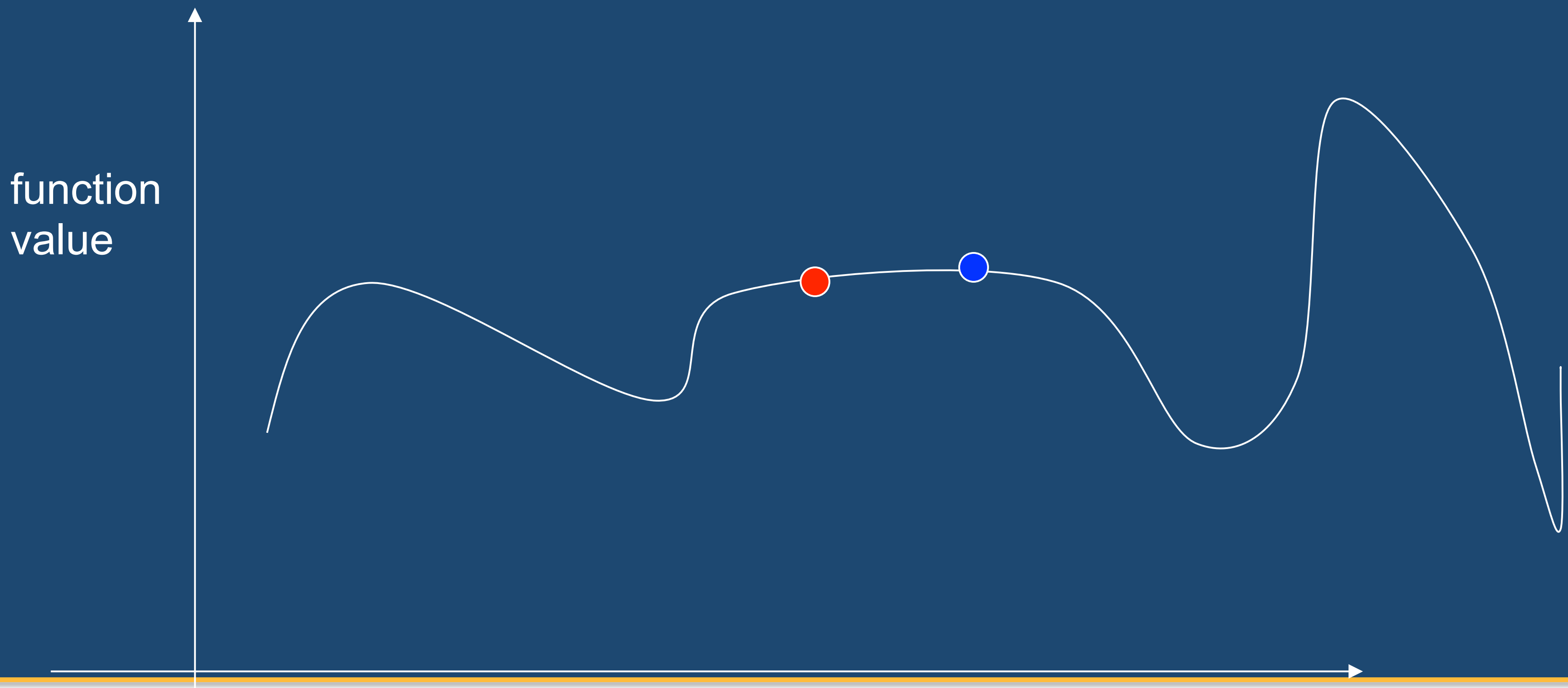
T = Very High



Simulated Annealing

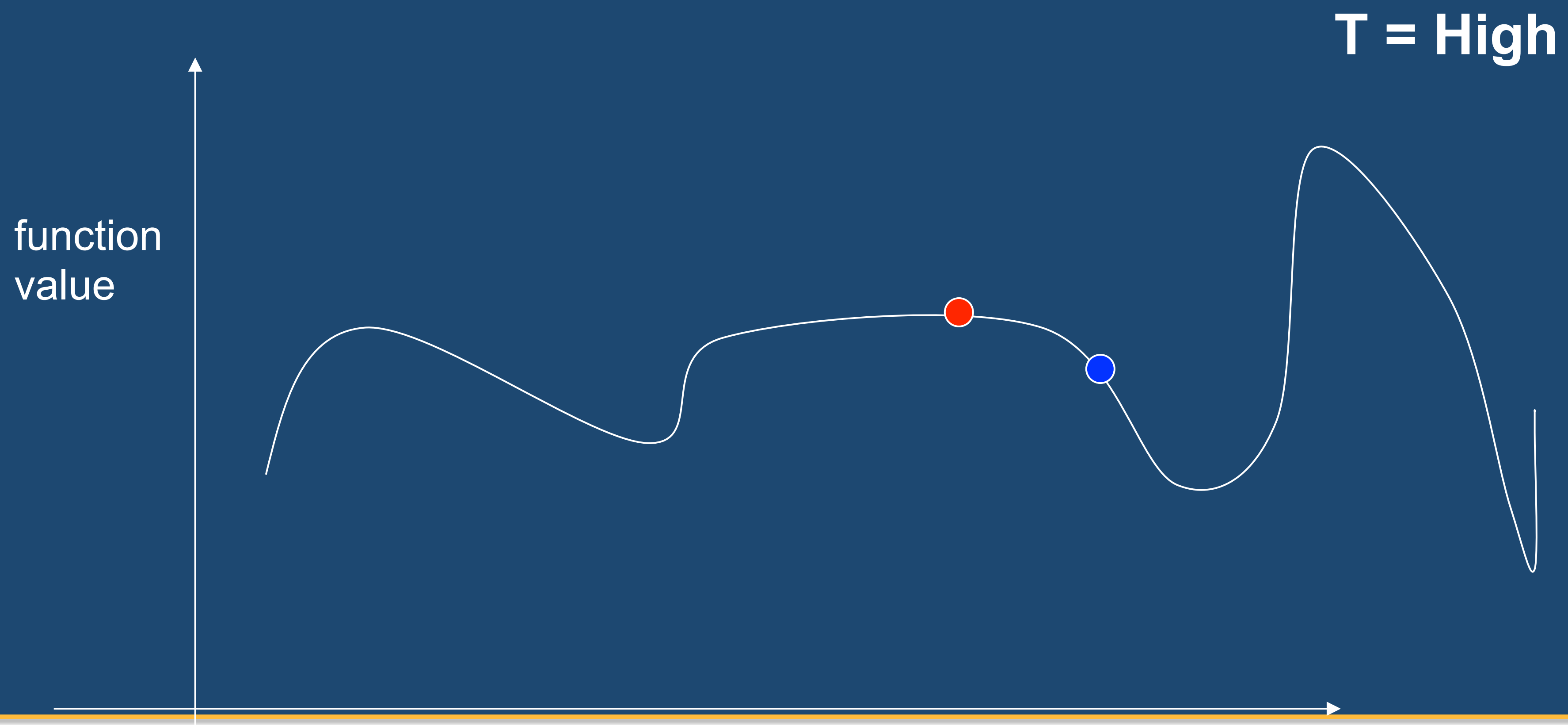
- Next Step; accept since higher E

T = Very High



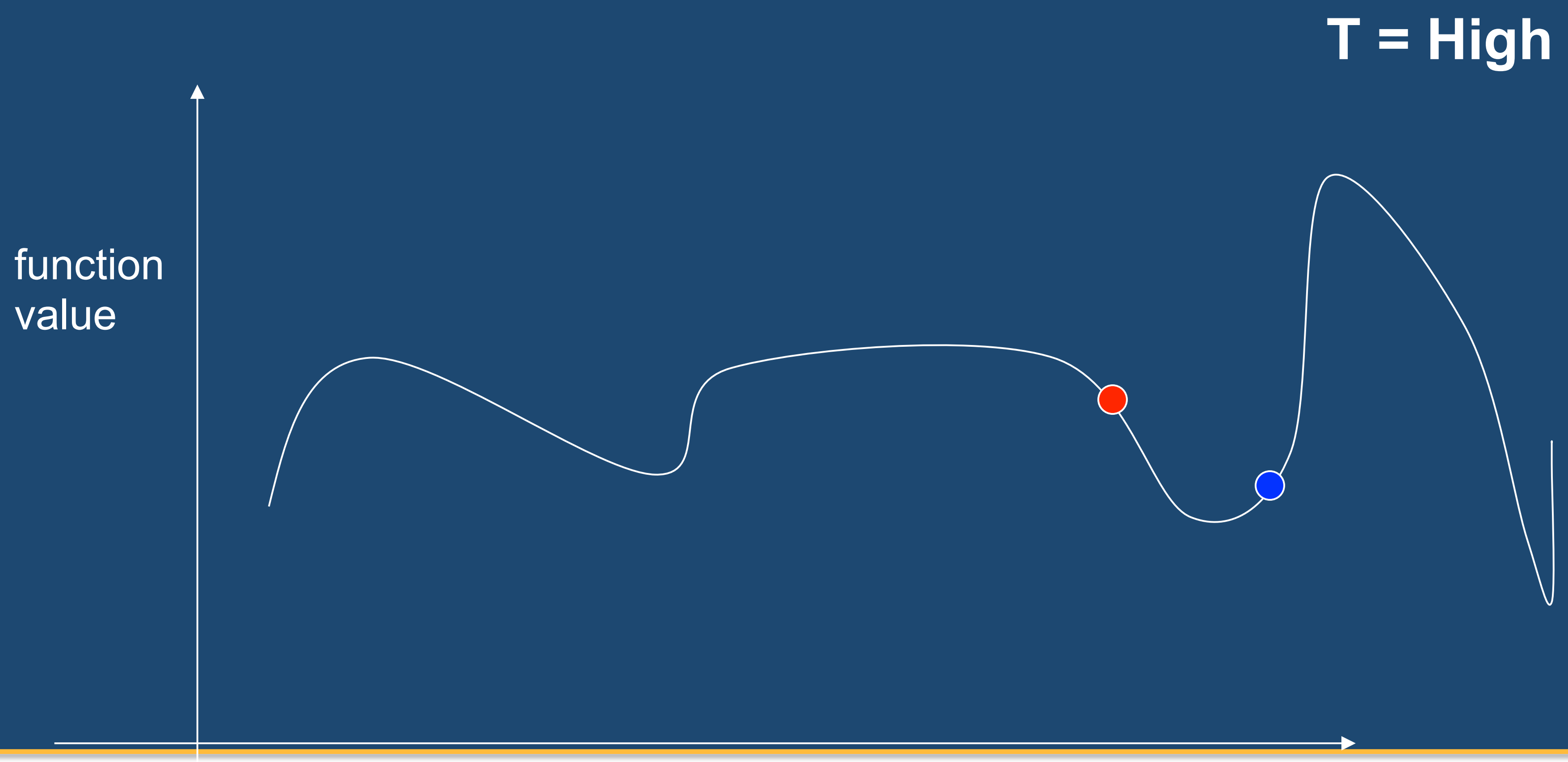
Simulated Annealing

- Next Step; accept even though lower



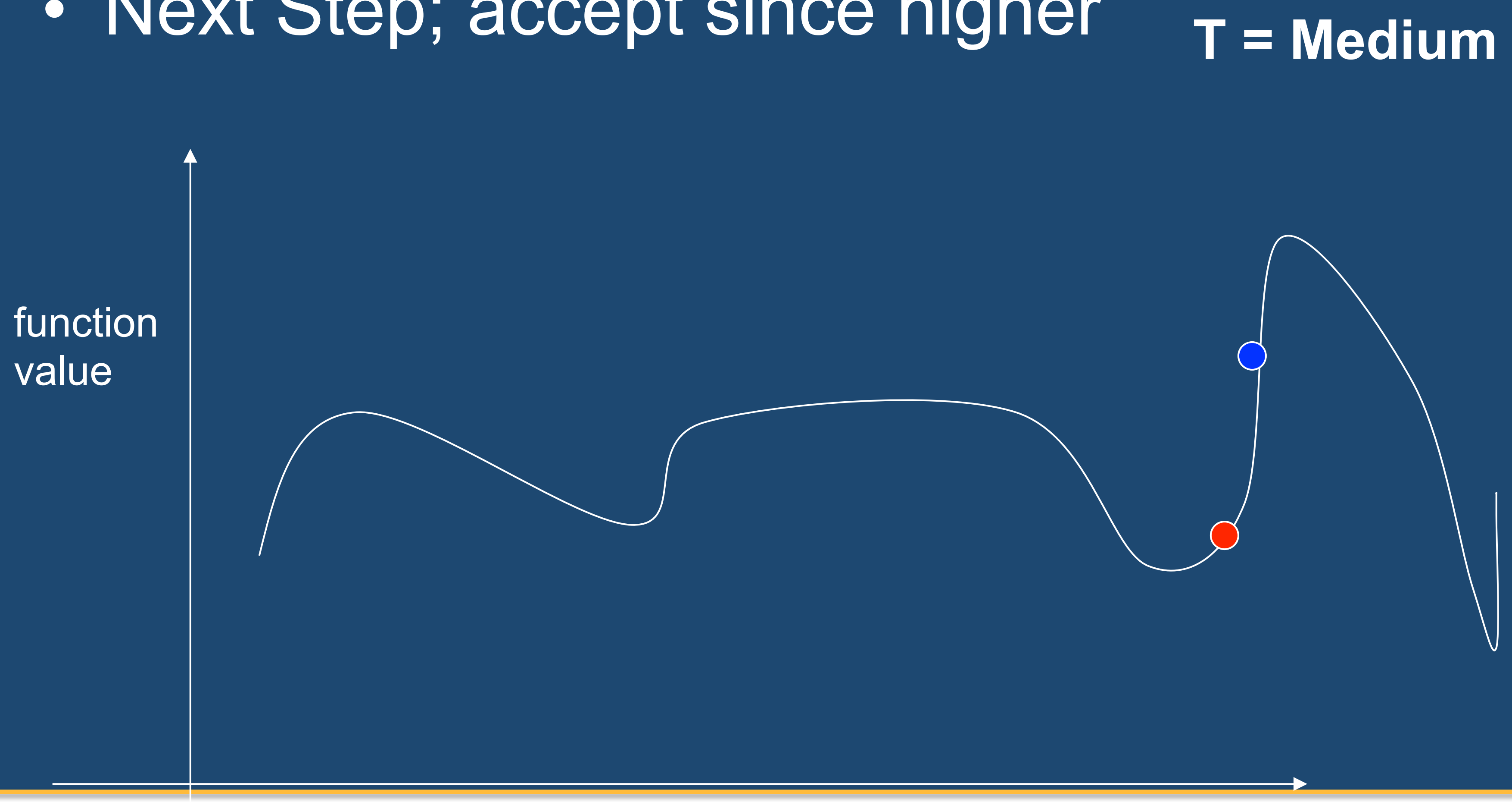
Simulated Annealing

- Next Step; accept even though lower



Simulated Annealing

- Next Step; accept since higher



Simulated Annealing

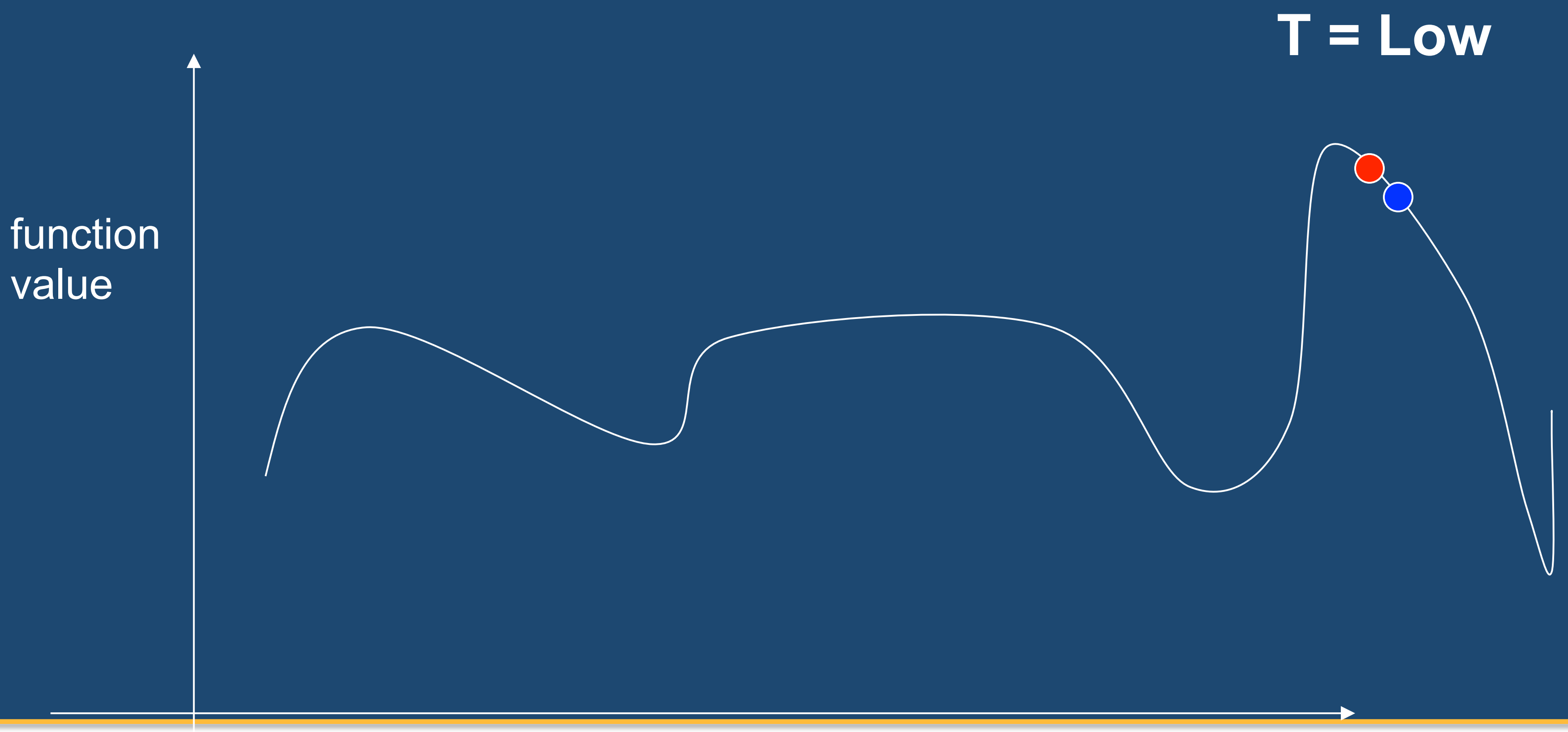
- Next Step; lower, but reject (T is falling)

Simulated Annealing

- Next Step; Accept since E is higher

Simulated Annealing

- Next Step; Accept since E change small

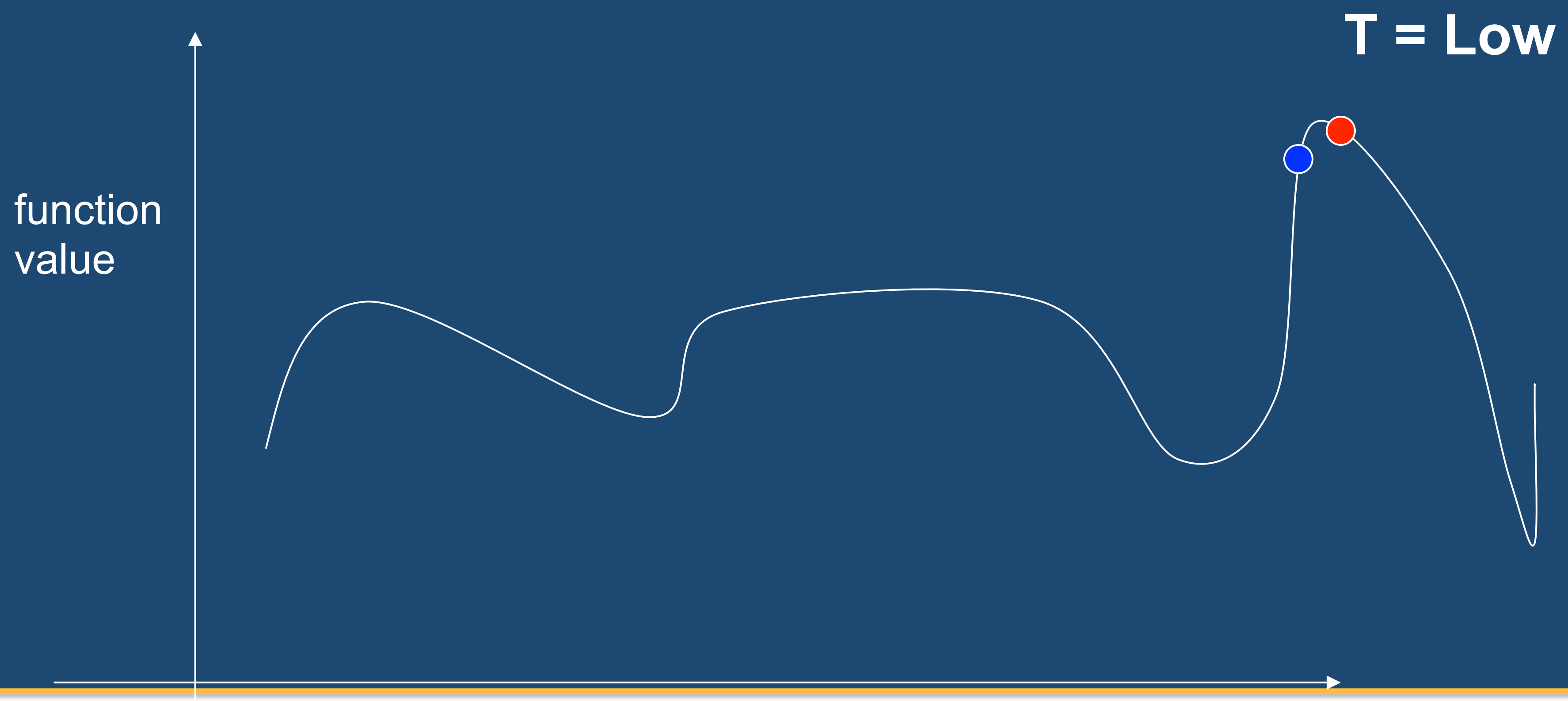


Simulated Annealing

- Next Step; Accept since E larget

Simulated Annealing

- Next Step; Reject since E lower and T low



Simulated Annealing

- Eventually converge to Maximum