

Local Search Pseudocode

Hill Climbing

```
1  current := initial state
2  value := evaluate(current)
3  while True:
4      candidate, candidate_value := find_best_child(current)
5      if candidate_value <= value
6          return current
7      current := candidate; value := candidate_value
```

Simulated Annealing

```
1  current := initial state
2  value := evaluate(current)
3  t = 0
4  while True:
5      t = t + 1
6      temp = annealing_schedule(t)
7      if temp = 0 then return current
8      candidate = random_successor(current)
9      diff = evaluate(candidate) - evaluate(current)
10     if diff > 0 then current := candidate
11     else if rand() < e(diff/temp) then current := candidate
```

Evolutionary Computation

```
1  population := generate_random_population(N)
2  while generations < limit
3      evaluate(population)
4      next_population := list
5      for n = 0 to N/2
6          parent1, parent2 := pick_parents(population)
7          child1, child2 := reproduce(parent1, parent2)
8          add child1, child2 to next_population.
9      population := next_population; generations += 1
```