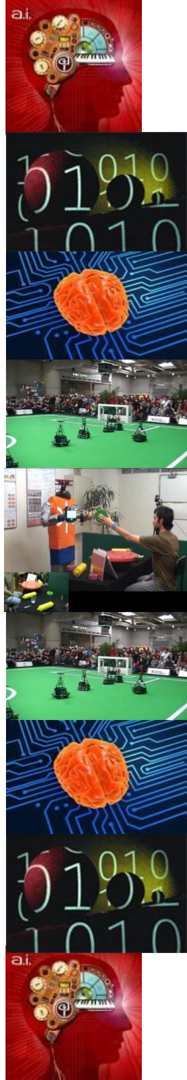
Local Optima & Tabu Search

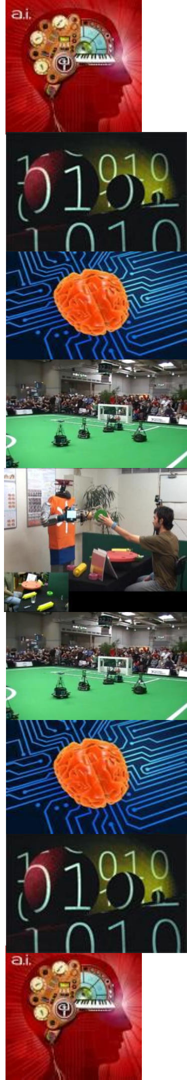
Local Optima

- Local searches have a tendency to get stuck in a local maxima or minima.
- This is a peak (or valley) in the landscape that is better than its neighbourhood.
- But not better in the search space.

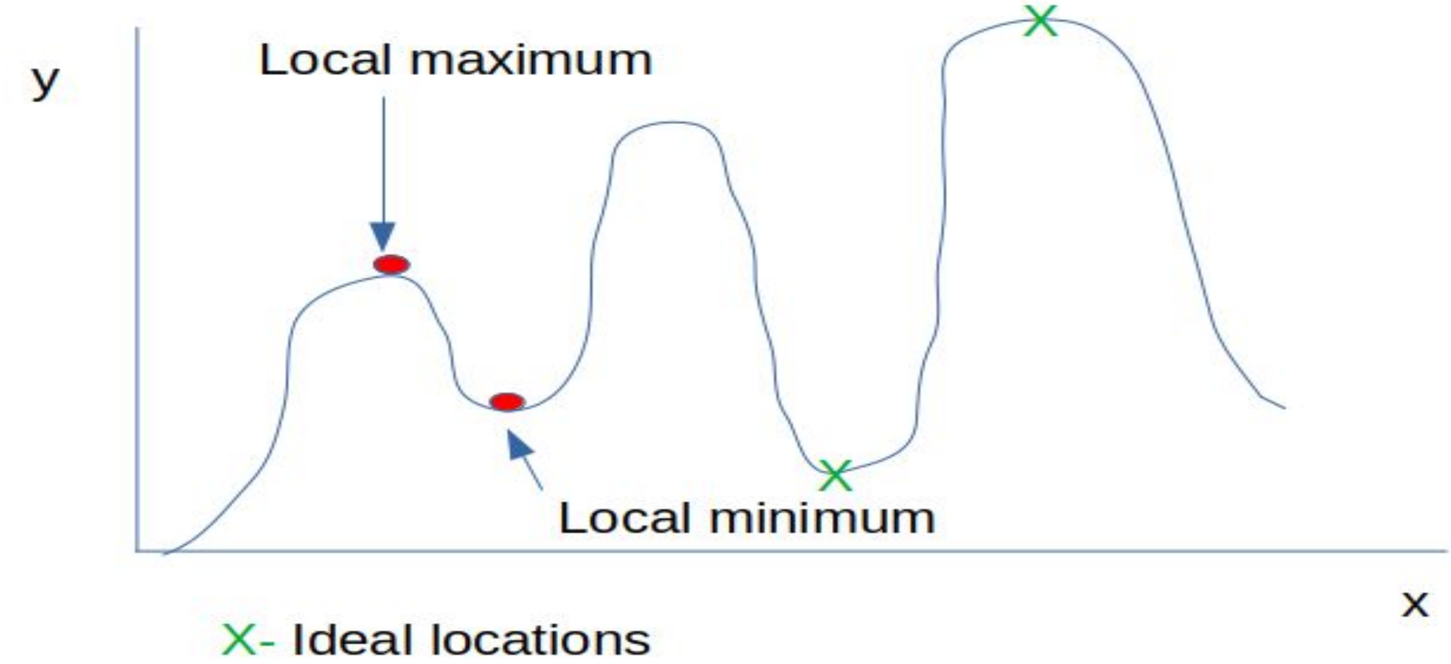


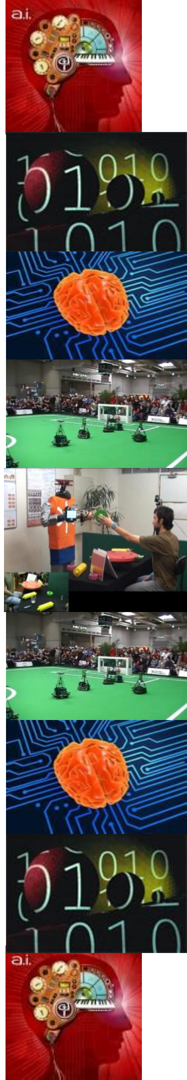
Neighbourhood

- Local neighbours of the current solution.
  - First improvement.
  - Best improvement.
  - Random selection.



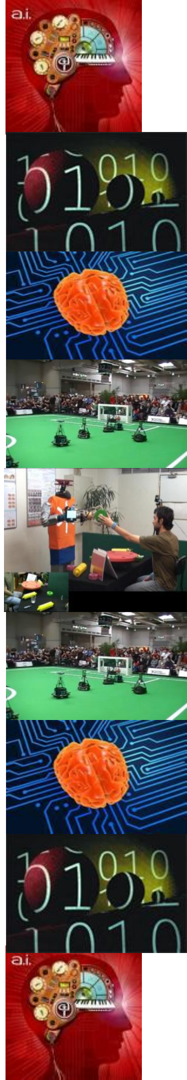
Local Optima





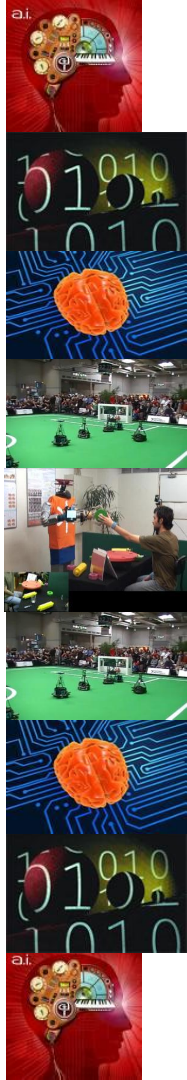
Local Optima

- Greedy - hill climbing can be used to find of an optima.
- However, due to its greedy-nature it can get stuck in a local optima as well.

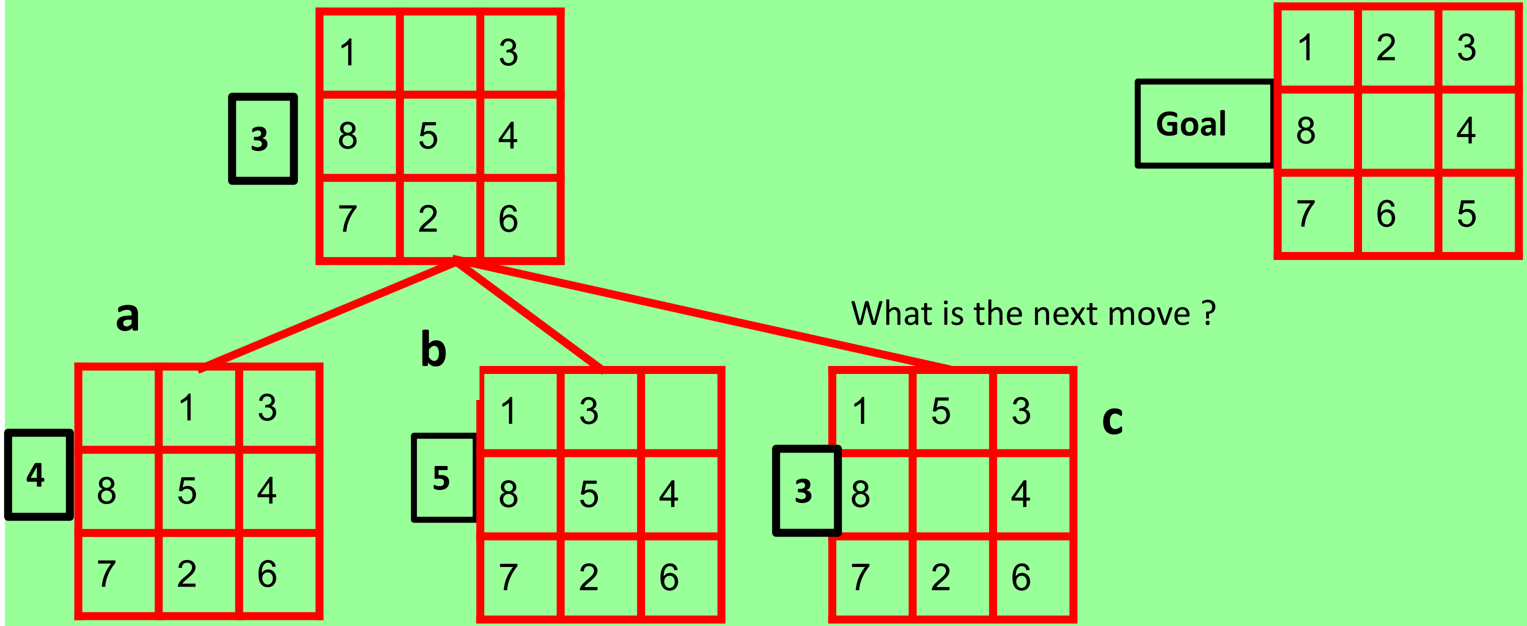


Local Optima

- The search may get stuck in a ridge or plateau .
- As this is a blind search it cannot see further than its region.
- It is also a single point search.



Hill-Climbing Search



What is the next move ?

Start (3):

1		3
8	5	4
7	2	6

Goal:

1	2	3
8		4
7	6	5

a (4):

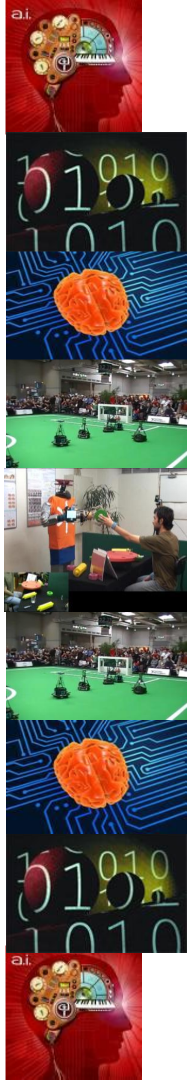
	1	3
8	5	4
7	2	6

b (5):

1	3	
8	5	4
7	2	6

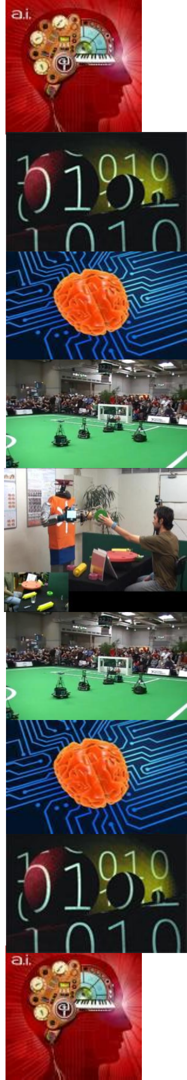
c (3):

1	5	3
8		4
7	2	6



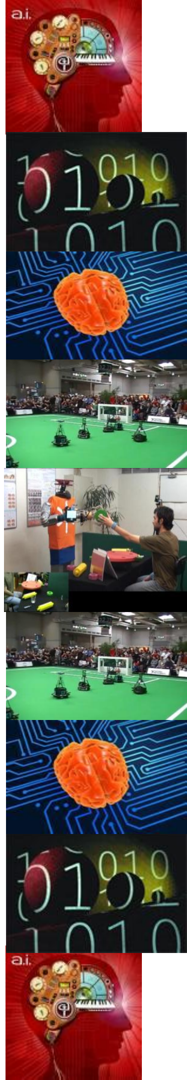
Combinatorial Problems

- These type of problems involve combining components to arrive at a solution.
- Components are objects of a solution.
- Obtained from a finite set.



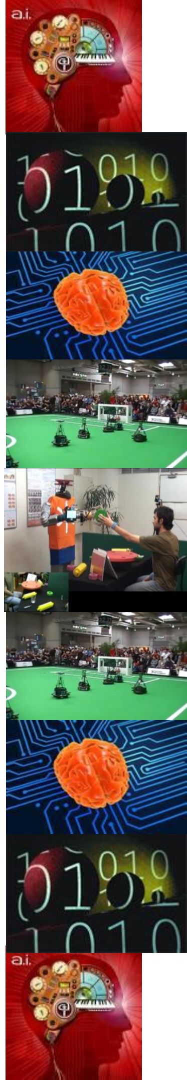
Optimisation

- Optimisation aims to find an optimal solution.
- TSP
- Knapsack problem
- Graph colouring



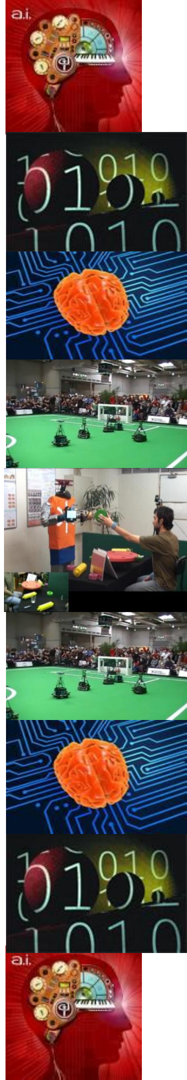
Problem and Solutions

- Problem.
- Problem instances.
- Constructive heuristic.
- Perturbative heuristic.
- Diversification.
- Intensification.
- Metaheuristic.



Systematic and Local Search

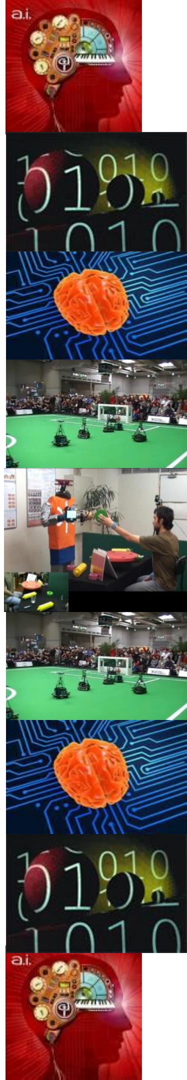
- Systematic- global and complete
- Local - neighbourhood and incomplete



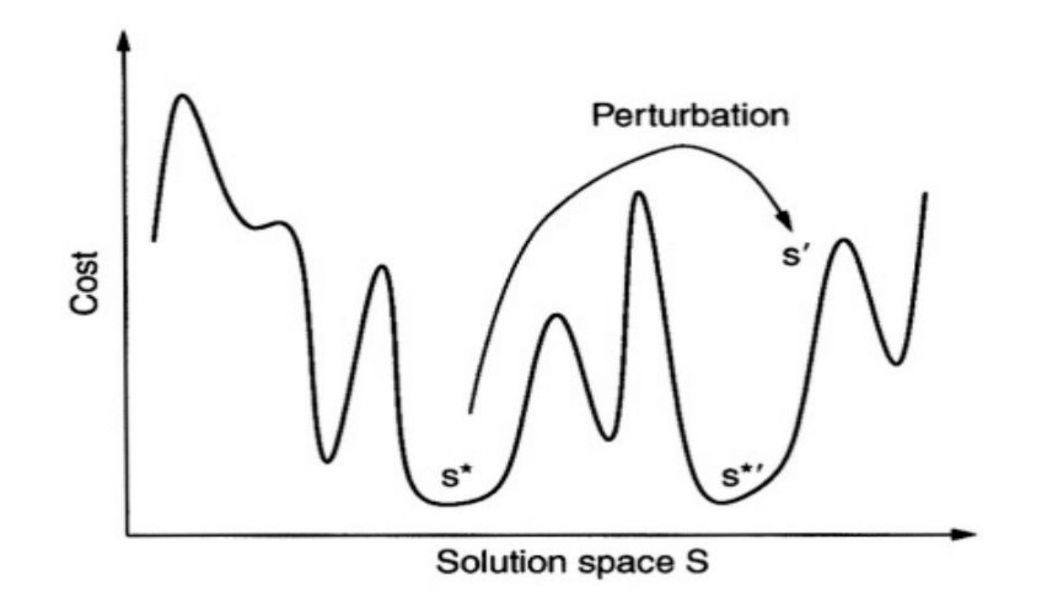
Iterated Local Search

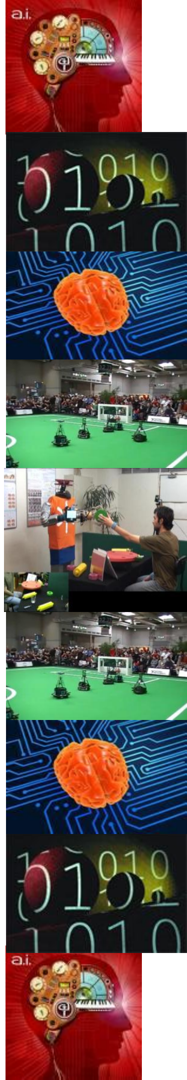
Generally four basic steps:

1. Generate InitialSolution,
2. LocalSearch,
3. Perturbation, and
4. Acceptance Criterion.



ILS

- 



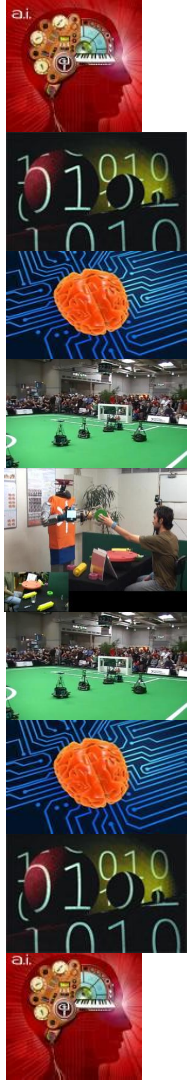
ILS

i) one can start with a random solution or one returned by some greedy construction heuristic;

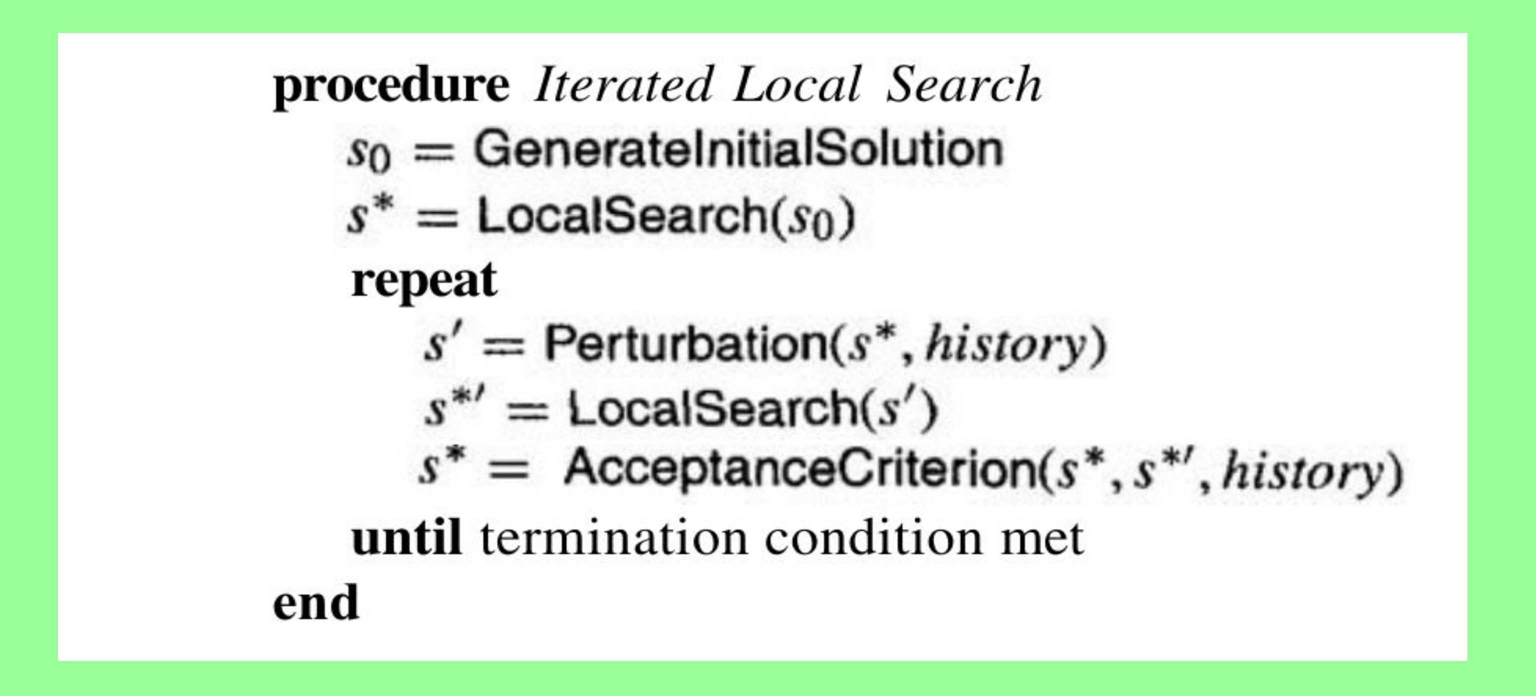
(ii) for most problems a local search algorithm is readily available;

iii) for the perturbation, a random move in a neighborhood of higher order than the one used by the local search algorithm can be surprisingly effective; and

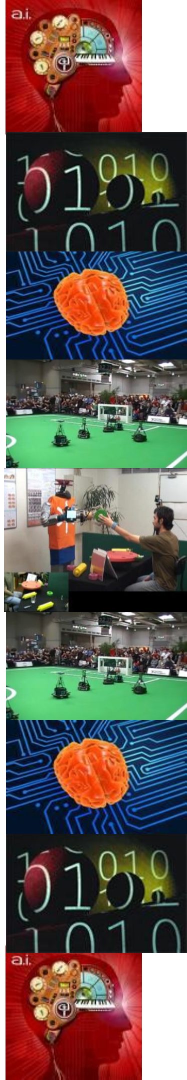
iv) a reasonable first guess for the acceptance criterion is to force the cost to decrease.



ILS- Algorithm

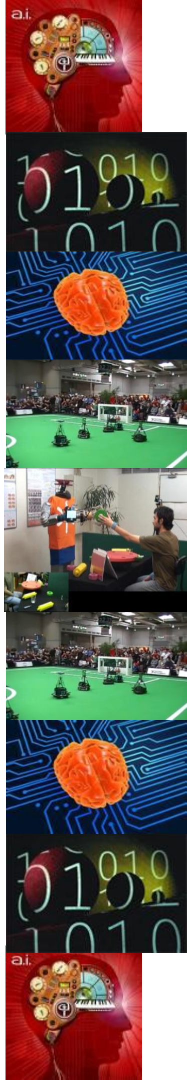


```
**procedure** *Iterated Local Search*
    $s_0$ = GenerateInitialSolution
    $s^*$ = LocalSearch($s_0$)
    **repeat**
        $s'$ = Perturbation($s^*$, $history$)
        $s^{*\prime}$ = LocalSearch($s'$)
        $s^*$ = AcceptanceCriterion($s^*$, $s^{*\prime}$, $history$)
    **until** termination condition met
**end**
```



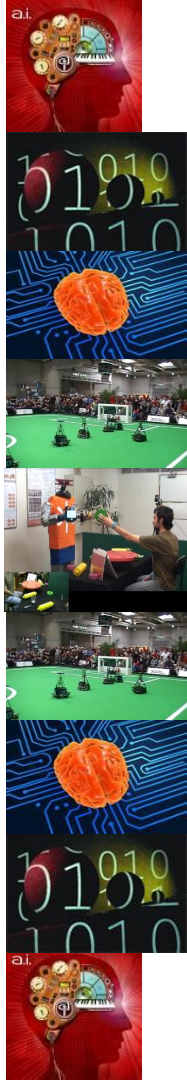
Tabu Search

- The Tabu search is a simple search.
- Usually underrated.
- But is very effective



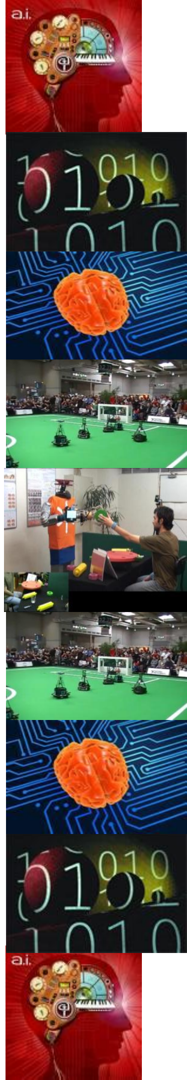
Tabu Search

- It maintains a list of previously visited positions(nodes) in the search space.
- Taboo list prohibits revisits
- It accepts worsening moves if no improvements are available.

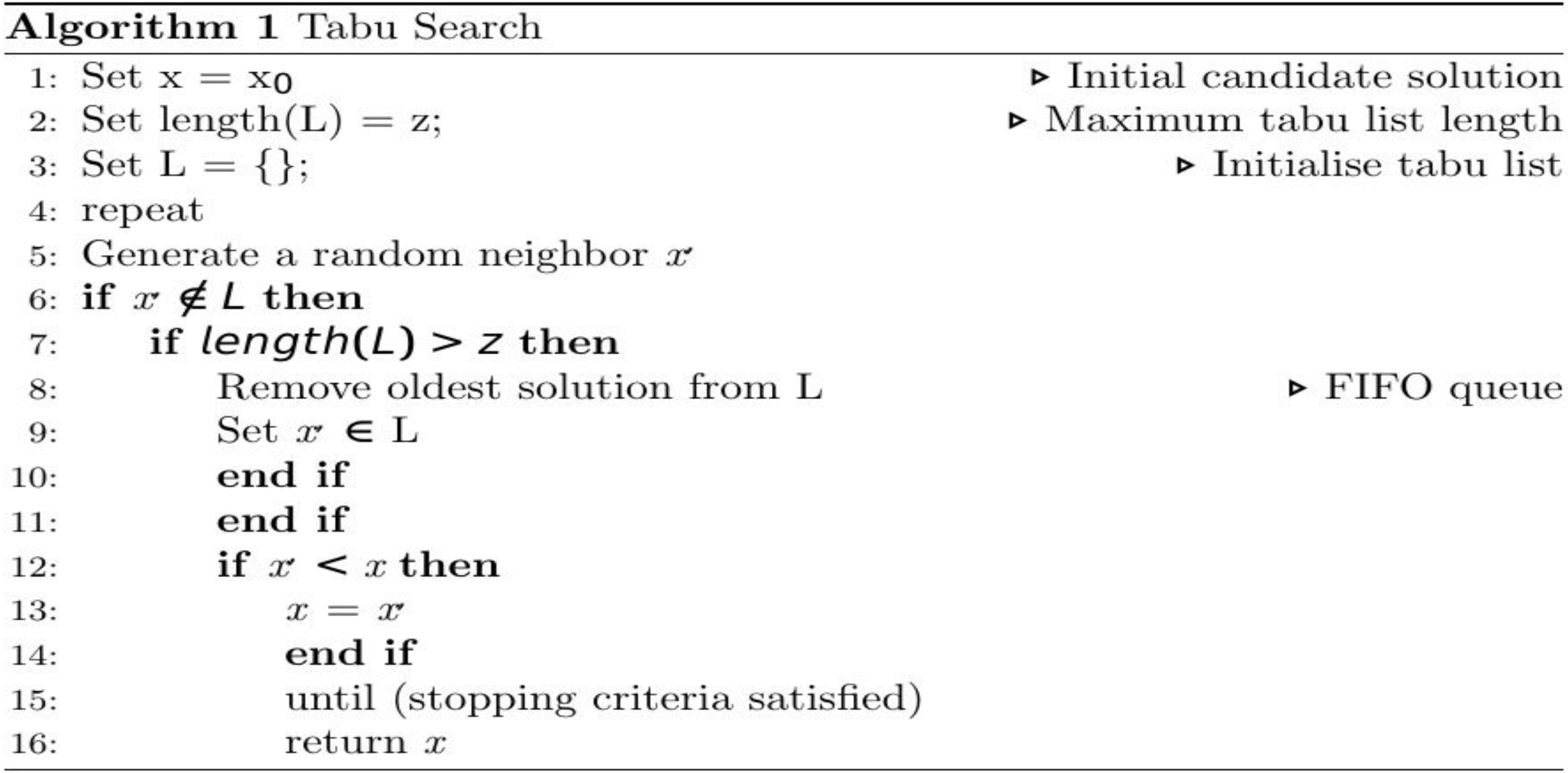


Tabu Search

- The tabu search prohibits going to the local optima once it has been visited.
- Forces the search to leave the local optima and accept worse moves by going away from the local optima.

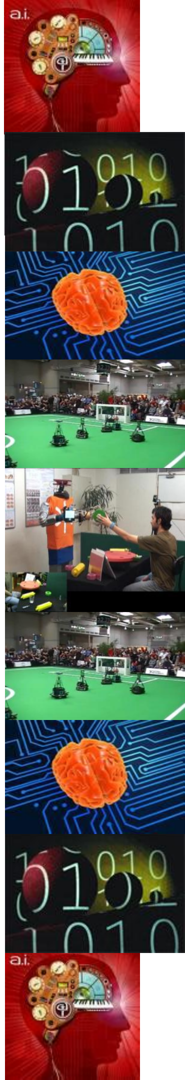


Tabu Algorithm



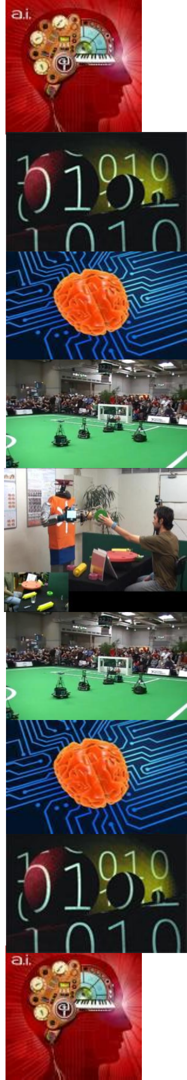
Algorithm 1 Tabu Search

```
Set x = $x_0$                                        ▷ Initial candidate solution
Set length(L) = z;                                   ▷ Maximum tabu list length
Set L = {};                                          ▷ Initialise tabu list
repeat
Generate a random neighbor $x$
if $x \notin L$ then
    if $length(L) > z$ then
        Remove oldest solution from L                ▷ FIFO queue
        Set $x \in$ L
        end if
        end if
        if $x < x$ then
            $x = x$
            end if
            until (stopping criteria satisfied)
            return $x$
```

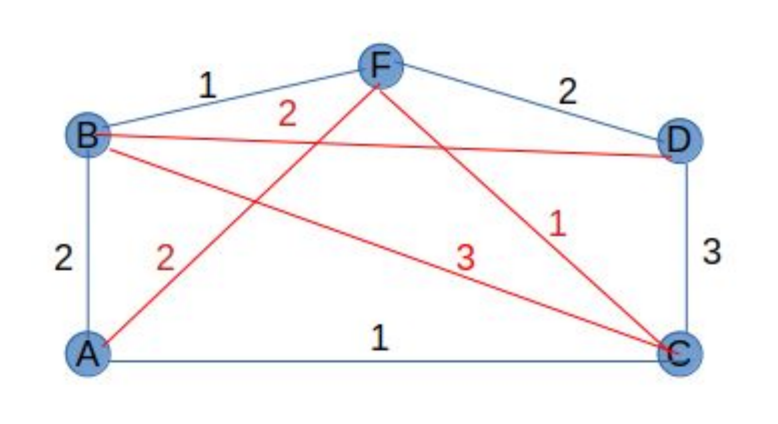


Tabu Search

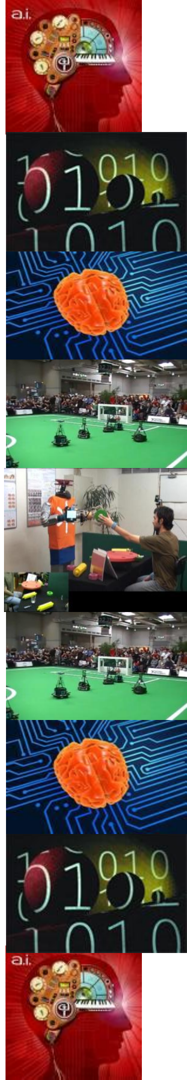
- The key feature of the tabu search is the use of memory.
- It accepts the best solution in the neighbourhood even if its worse than the current solution.
- To prevent loops it “taboos” going to previously visited solutions.



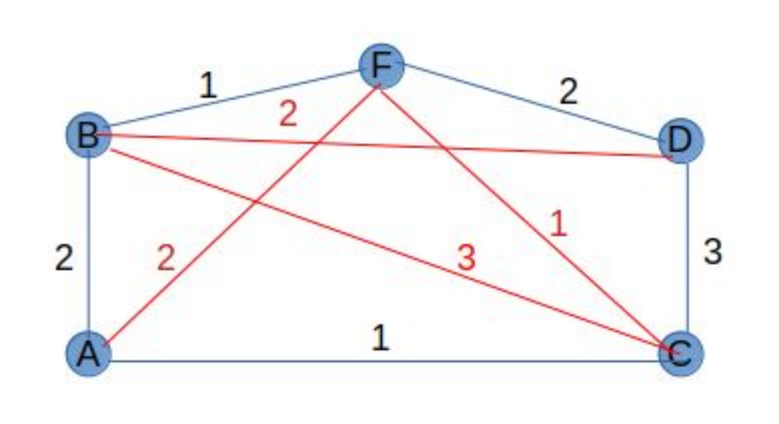
Tabu Search- Example



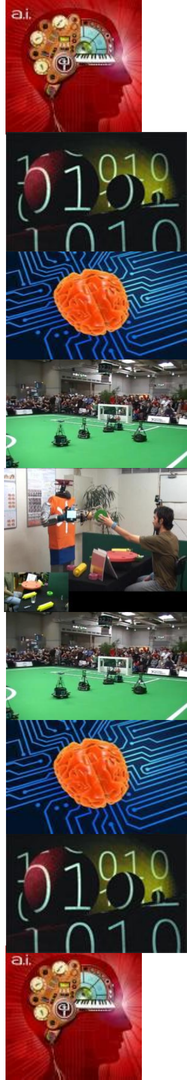
Given the following TSP problem and the following states.



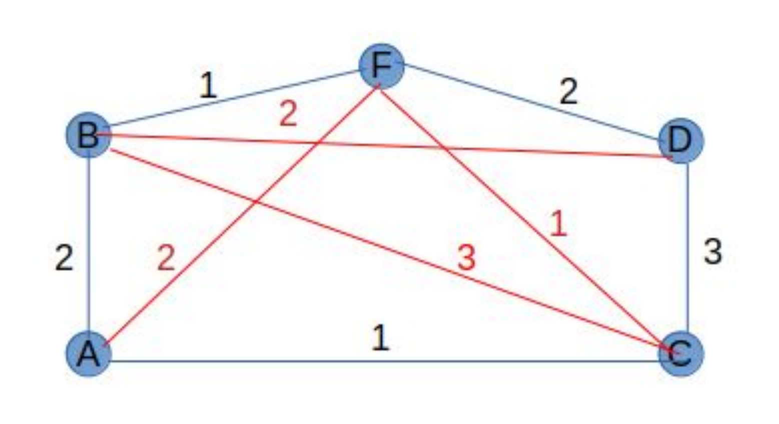
Tabu Search- Example



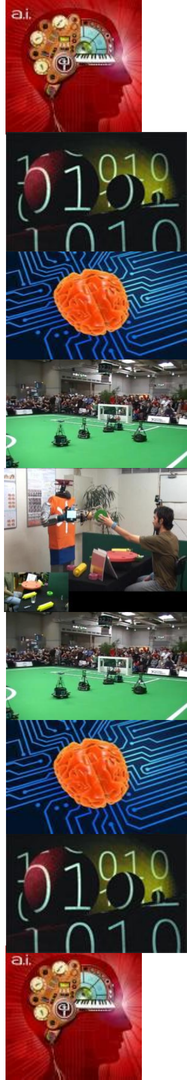
- State= {F,B,A,C,D,F} ={1+2+1,3,2} = 9



Tabu Search- Example

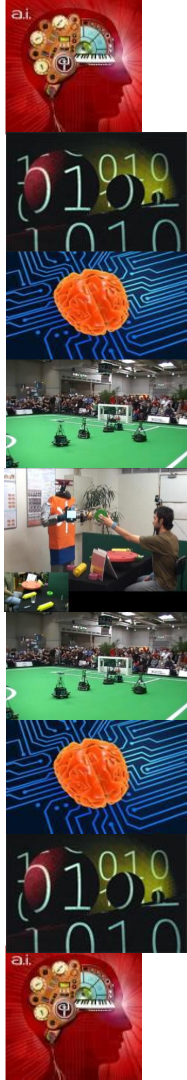


- State= {F,B,A,C,D,F} ={1+2+1,3,2} = 9
- s’ = {F,A,B,C,D,F} ={2+2+3+3+2} =12



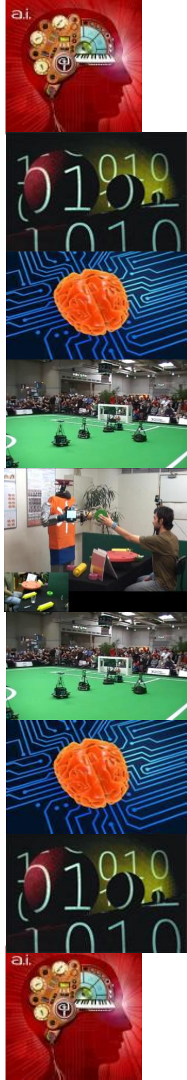
Tabu Search

- State= {F,B,A,C,D,F} ={1+2+1,3,2} = 9.
- Tabu search will generate the following (will hill-climbing ?)
- State = {F,A,B,C,D,F} ={2+2+3+3+2} =12. (accepted or not ? insert in L)
- State = {F,C,A,B,D,F}={1+1+2+2+2}=8



Disadvantages Tabu Search

- Too many parameters to be considered.
- Number of iterations could be large
- Global optimal may not be found and depends on configuration.



QUESTIONS