Web Algorithms

Eng. Fabio Persia, PhD

Algorithmic techniques: local search

Characteristics

- We define for every feasible solution *y* a subset of feasible "neighbor" solution, called the neighborhood of *y* or simply *neighborhood(y)*.

- Starting from an initial solution, we repeatedly switch to a better solution in the current neighborhood, until possible.

Scheme of a local search algorithm

- Fix an initial solution y feasible for input x (usually trivial one)
- While ($\exists\ y' \in$ *neighborhood*(y) better than y)
  let $y=y'$
- Return y

In order to define a local search algorithm for a given problem it is thus sufficient to define:

1. the initial solution
2. the neighborhood of feasible solutions

Complexity

In order to get polynomial time:

1. The initial solution must be determined in polynomial time

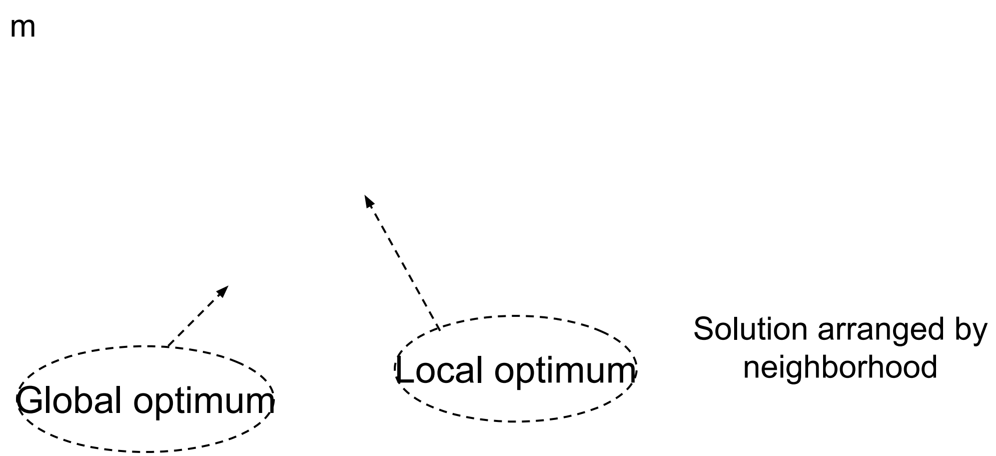
1. The test of the guard condition of the while with the eventual consequent determination of a better solution in the neighborhood must be done in polynomial time

   WARNING: the neighborhood may have exponential cardinality with respect to the input size!

3. The number of while iterations must be polynomial

Approximation

The returned solution *y* has not any better one in its neighborhood, that is it is a local optimum.



In order to bound the approximation ratio it is sufficient to bound the ratio between the value of any local optimum with the one of measure of an optimal (global) solution.

Neighborhood definition

Neighborhood(y):

- Sufficiently “rich” to obtain good solutions / local optima
- Sufficiently “poor” to guarantee a polynomial time complexity

Extremal cases …

- *Neighborhood(y)= Ø:*
  - running time polynomial (if initial solution determined in polynomial time))
  - bad approximation, since every solution is a local optimum

- *Neighborhood(y)= S(x),* that is the set of all possible feasible solutions of *x:*
  - good approximation, since every local optimum is also a global optimum
  - running time not polynomial (if problem is NP-hard)

Max Cut

- INPUT: Graph $G=(V,E)$

- SOLUTION: Partition of V in two subsets V_1 and V_2, that is such that $V_1 \cup V_2 = V$ and $V_1 \cap V_2 = \emptyset$

- MEASURE: Cardinality of the cut, that is number of edges with an endpoint in V_1 and the other endpoint in V_2, i.e.

$$| \{ \{u,v\} \mid u \in V_1 \text{ and } v \in V_2 \} |$$

Algorithm

In order to define the local search algorithm, it suffices to determine:

1. Initial solution: $V_1=V$ and $V_2=\emptyset$

1. Neighborhood: given $V=\{v_1,\ldots,v_n\}$ and V_1,V_2, the neighbor solutions of (V_1,V_2) are all the pairs (V_{1i},V_{2i}) with $1\leq i\leq n$ that can be obtained moving a node v_i from V_1 to V_2 or vice versa, that is such that

    if $v_i\in V_1$ $V_{1i} = V_1 \setminus \{v_i\}$ and $V_{2i} = V_2 \cup \{v_i\}$

    else $v_i\in V_2$ $V_{1i} = V_1 \cup \{v_i\}$ and $V_{2i} = V_2 \setminus \{v_i\}$

Complexity

1. Initial solution trivially obtained in polynomial time

1. Test of the while guard and eventual determination of a better neighbor solution performed in polynomial time as follows:
   - For each of the n neighbor solutions (n iterations), check whether it is improves (n^2 iterations) → $O(n^3)$

2. While iterations at most $|E|=O(n^2)$, since every iteration improves the current solution, that is increases at least of one the number of edges in the cut, and there are at $|E|$ edges in the cut

Thus the algorithm has time complexity $O(n^3 \cdot n^2)=O(n^5)$.

Approximation

Let us see a useful property for showing the approximation ratio of the algorithm:

Fact: Given a graph G=(V,E), let δ_i the degree of a generic node $v_i \in V$. Then,

$$\sum_{i=1}^{n} \delta_i = 2\,|E|$$

Proof: Trivially true, since each edge is counted twice in the summation, that is it increments the summation by 2.

□

Theorem: The local search algorithm is ½-approximating

Proof:

Let us show that any local optimum (V_1, V_2) has measure $m \geq \frac{|E|}{2}$.

This implies $\frac{m}{m^*} \geq \frac{\frac{|E|}{2}}{|E|} = \frac{1}{2}$

since $m^* \leq |E|$.

Given a local optimum (V_1, V_2) let us denote by h the number of *internal* edges, that is with both endpoints in V_1 or both in V_2

Clearly $m+h=|E|$.

For every node $v_i \in V$ let us define the internal and external degrees of the node as follows:

- δ_i^{int}=number of edges that v_i has towards nodes in its partition, that is

  $\delta_i^{int} = |\ \{v_k \mid \{v_i, v_k\} \in E \ \ and \ \ (v_i, v_k \in V_1) \ or \ (v_i, v_k \in V_2)|$

- δ_i^{ext}=number of edges that v_i has towards nodes in the other partition, that is

  $\delta_i^{ext} = |\ \{v_k \mid \{v_i, v_k\} \in E \ \ and \ \ (v_i \in V_1 \ , \ v_k \in V_2) \ or \ (v_i \in V_2 \ , \ v_k \in V_1)\ \}$

Since the neighbor solution (V_{1i}, V_{2i}) has measure not greater than the one of (V_1, V_2) (the local optimum), we have

$$m - \delta_i^{ext} + \delta_i^{int} \leq m \quad \text{and thus } \delta_i^{int} - \delta_i^{ext} \leq 0$$

Summing up over all the nodes v_i we have that

$$\Sigma_{vi\in V}\,\delta_i^{int} - \Sigma_{vi\in V}\,\delta_i^{est} = \Sigma_{vi\in V}\,\delta_i^{int} - \delta_i^{est} \leq 0$$

From the previous fact $\Sigma_{vi\in V}\,\delta_i^{int} = 2h$

(because it is like summing the degrees of the nodes of the graph containing only the internal edges)

and $\Sigma_{vi\in V}\,\delta_i^{est} = 2m$

(because it is like summing the degrees of the nodes of the graph containing only the external edges)

Thus $0 \geq \Sigma_{vi\in V}\,\delta_i^{int} - \Sigma_{vi\in V}\,\delta_i^{est} = 2h - 2m$, that is $m \geq h$

So that, adding m to both sides and dividing by 2

$$m \geq (m+h)/2 = |E|/2 .$$

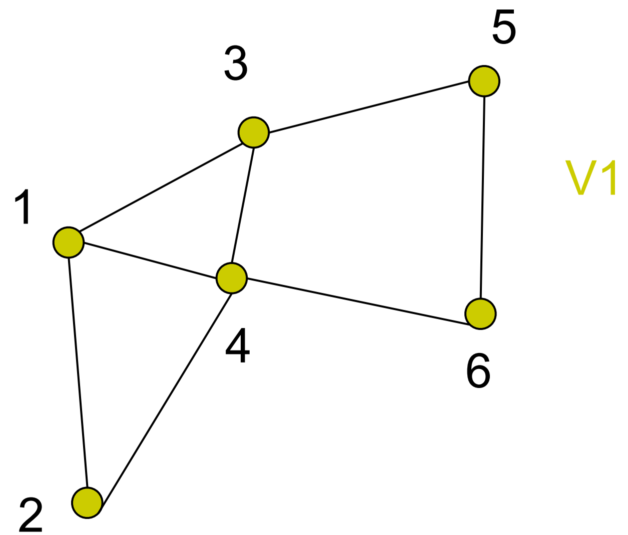
□

Example of execution

Step 1

Current solution:

V2=Ø



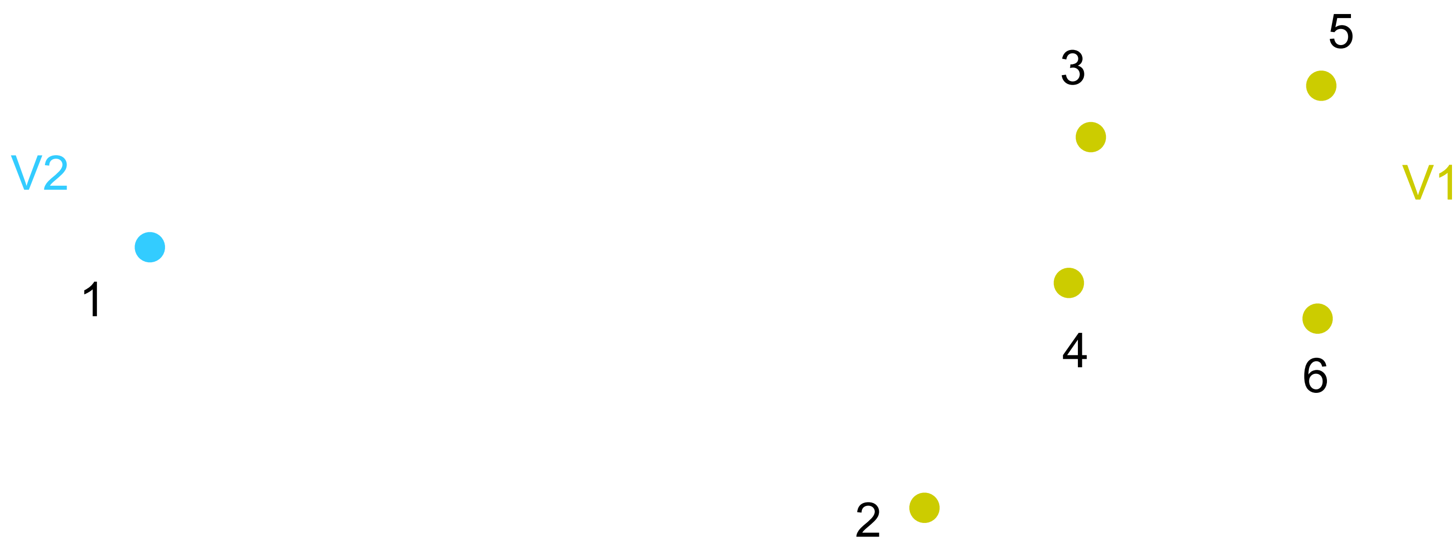
Measure: 0

MEASURE: Cut cardinality, that is number of edges with an endpoint in V_1 and the other in V_2

Example of execution

Step 2

Current solution

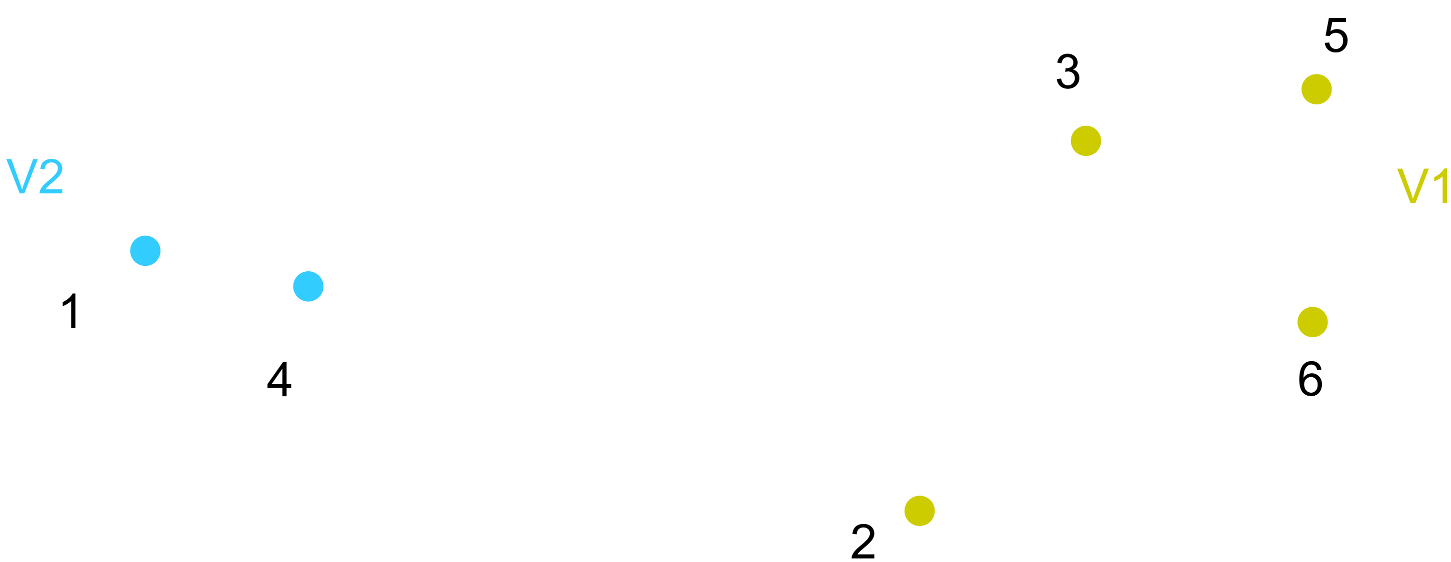


Measure: 3

Example of execution

Step 3

Current solution

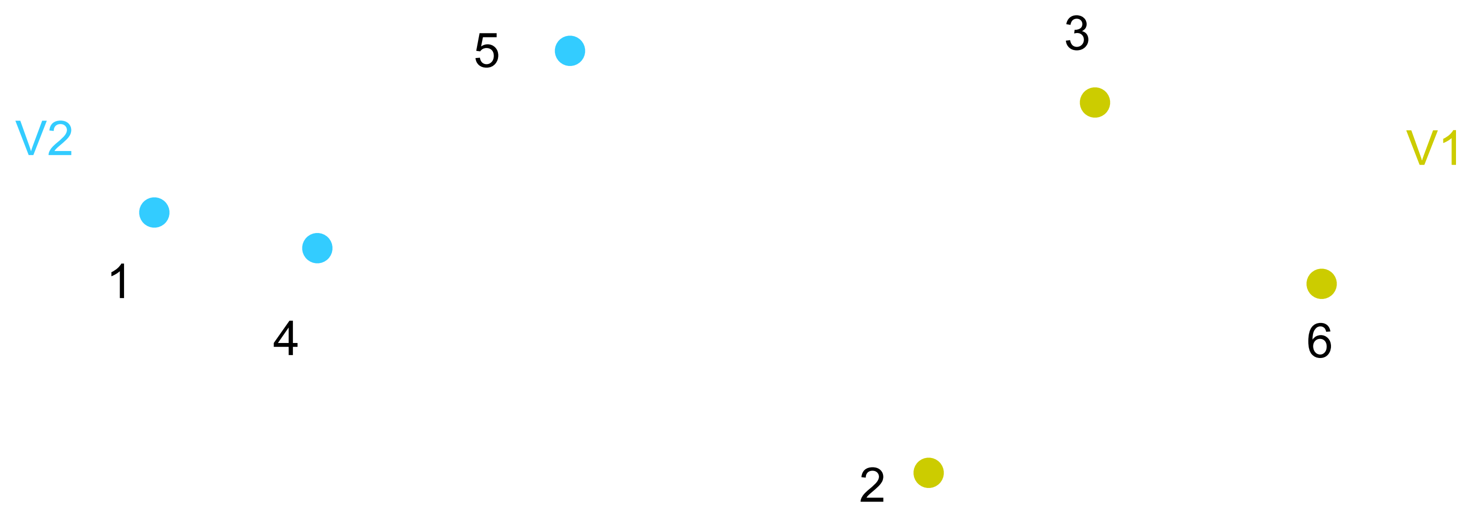


Measure: 5

Example of execution

Step 4

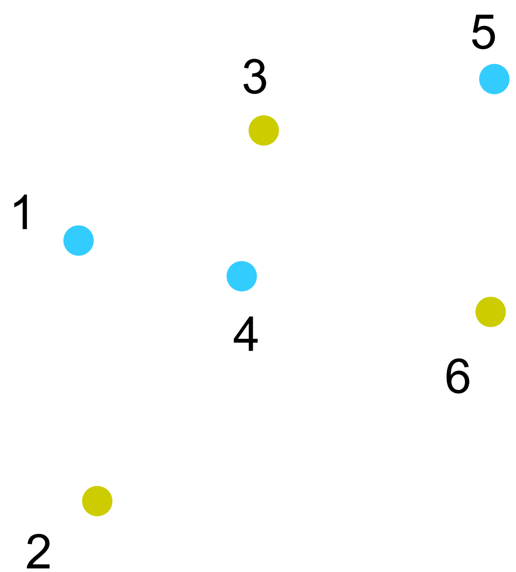
Current solution



Measure: 7

Example of execution

Local optimum



Measure: 7

Conclusions on local search

As greedy algorithms, local search algorithms have good performance in practice and lead to the determination of good heuristics (algorithms performing well in practice but that usually do not have guaranteed performances in terms of time or approximation).