Heuristic Algorithms

Master's Degree in Computer Science/Mathematics

Roberto Cordone

DI - Università degli Studi di Milano

Schedule:	Thursday 14.30 - 16.30 on MS-Teams
	Friday 14.30 - 16.30 on MS-Teams
Office hours:	on appointment
E-mail:	`roberto.cordone@unimi.it`
Web page:	`https://homes.di.unimi.it/cordone/courses/2020-ae/2020-ae.html`
Ariel site:	`https://rcordoneha.ariel.ctu.unimi.it`

Effectiveness of a heuristic algorithm

A heuristic algorithm is useful if it is

1. efficient: it "costs" much less than an exact algorithm
2. effective: it "frequently" returns a solution "close to" an exact one

Let us now discuss the effectiveness of heuristic algorithms:

- closeness of the solution obtained to an optimal one
- frequency of hitting optimal or nearly optimal solutions

These features can be combined into a

- frequency distribution of solutions more or less close to the optimum

The effectiveness of a heuristic algorithm can be investigated with a

- theoretical analysis (*a priori*), proving that the algorithm finds always or with a given frequency solutions with a given guarantee of quality
- experimental analysis (*a posteriori*), measuring the performance of the algorithm on sampled benchmark instances to show that it occurs

Effectiveness of a heuristic algorithm

The effectiveness of a heuristic optimization algorithm A is measured by the difference between the heuristic value $f_A(I)$ and the optimum $f^*(I)$

- absolute difference:

$$\tilde{\delta}_A(I) = |f_A(I) - f^*(I)| \geq 0$$

  rarely used, and only when the objective is a pure number

- relative difference:

$$\delta_A(I) = \frac{|f_A(I) - f^*(I)|}{f^*(I)} \geq 0$$

  frequent in experimental analysis (*usually as a percent ratio*)

- approximation ratio:

$$\rho_A(I) = \max\left[\frac{f_A(I)}{f^*(I)}, \frac{f^*(I)}{f_A(I)}\right] \geq 1$$

  frequent in theoretical analysis: the first form is used for minimization problems, the second one for maximization problems

Theoretical analysis (in the worst case)

To obtain a compact measure, independent from I, find the worst case
(*as for efficiency, that is complexity*)

The difference between $f_A(I)$ and $f^*(I)$ is in general unlimited,
but for some algorithms it is limited:

- absolute approximation:

$$\exists \tilde{\alpha}_A \in \mathbb{N} : \tilde{\delta}_A(I) \leq \tilde{\alpha}_A \text{ for each } I \in \mathcal{I}$$

  A (rare) example is Vizing's algorithm for *Edge Coloring* ($\tilde{\alpha}_A = 1$)

- relative approximation:

$$\exists \alpha_A \in \mathbb{R}^+ : \rho_A(I) \leq \alpha_A \text{ for each } I \in \mathcal{I}$$

Factor α_A ($\tilde{\alpha}_A$) is the relative (absolute) approximation guarantee

For other algorithms, the guarantee depends on the instance size

$$\rho_A(I) \leq \alpha_A(n) \text{ for each } I \in \mathcal{I}_n, n \in \mathbb{N}$$

Effectiveness can be independent from size (contrary to efficiency)

How to achieve an approximation guarantee?

For a minimization problem, the aim is to prove that

$$\exists \alpha_A \in \mathbb{R} : f_A(I) \leq \alpha_A f^*(I) \text{ for each } I \in \mathcal{I}$$

1. find a way to build an underestimate $LB(I)$

$$LB(I) \leq f^*(I) \qquad I \in \mathcal{I}$$

2. find a way to build an overestimate $UB(I)$, related to $LB(I)$ by a coefficient α_A

$$UB(I) = \alpha_A \, LB(I) \qquad I \in \mathcal{I}$$

3. find an algorithm A whose solution is not worse than $UB(I)$

$$f_A(I) \leq UB(I) \qquad I \in \mathcal{I}$$

Then $f_A(I) \leq UB(I) = \alpha_A LB(I) \leq \alpha_A f^*(I)$, for each $I \in \mathcal{I}$

$$f_A(I) \leq \alpha_A f^*(I) \text{ for each } I \in \mathcal{I}$$

A 2-approximated algorithm for the *VCP*

Given a undirected graph $G = (V, E)$ find the minimum cardinality vertex subset such that each edge of graph is incident to it

A matching is a set of nonadjacent edges

Maximal matching is a matching such that any other edge of the graph is adjacent to one of its edges (*it cannot be enlarged*)

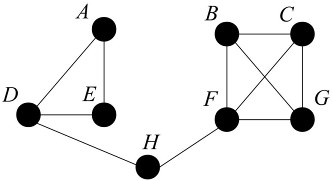
Matching algorithm:

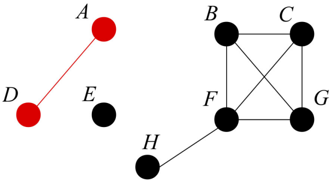
1. Build a maximal matching $M \subseteq E$ scanning the edges of E and including in M those not adjacent to M
   ($now\ every\ edge\ of\ E \setminus M\ is\ adjacent\ to\ an\ edge\ of\ M$)
2. The set of extreme vertices of the matching edges is a *VCP* solution

$$x_A := \bigcup_{(u,v) \in M} \{u, v\}$$

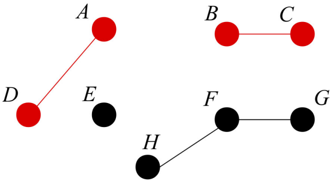
and it can be improved removing the redundant vertices

Example

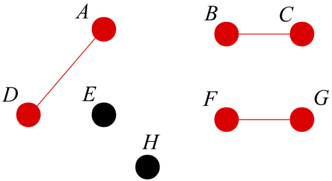




$|M| = 1$



$|M| = 2$



$f_A = 2 \cdot |M| = 6$ (and G is redundant)

The optimum is $f^* = 5$

Proof

The matching algorithm is 2-approximated

1. The cardinality of matching M is an underestimate $LB(I)$
   - the cardinality of an optimal covering for any subset of edges $E' \subseteq E$ does not exceed that of an optimal covering for E
     $$|x^*_{E'}| \leq |x^*_E|$$
     (*it costs more to cover all edges than only the matching*)
   - the optimal covering of a matching M has cardinality $|M|$
     (*each edge of the matching requires exactly one different vertex*)
2. Including both the extremes of each edge of the matching yields
   - an overestimate (*it covers both the matching and the adjacent edges*)
   - of value $UB(I) = 2LB(I)$ (*two different vertices for each edge*)
3. The matching algorithm returns solutions of value $f_A(I) \leq UB(I)$
   (*possibly removing redundant vertices*)

This implies $f_A(I) \leq 2f^*(I)$ for each $I \in \mathcal{I}$, that is $\alpha_A = 2$

... and the bound is tight!

Since α_A relates $UB(I)$ and $LB(I)$, $f_A(I)$ and $f^*(I)$ could be closer

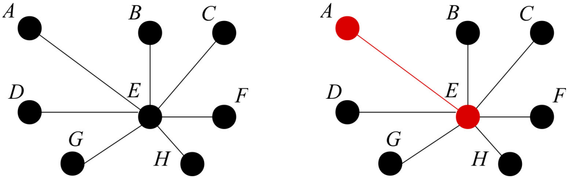
Actually, for many instances $\rho_A(I)$ is much better than α_A

Are there instances $\bar{I}$ for which $f_A(\bar{I}) = \alpha_A f^*(\bar{I})$? How are they like?

The study of these instances is useful to

- evaluate whether they are rare or frequent
- introduce *ad hoc* modifications to improve the algorithm

In the literature the typical expression "*and the bound is tight*" introduces the description of instances exhibiting the worst case



Patching all worst cases improves the approximation ratio

The *TSP* under the triangle inequality

Consider the *TSP* with the additional (rather common) assumptions that

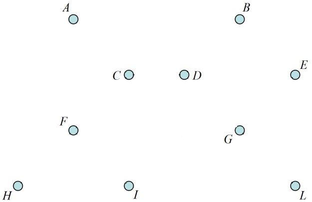
- graph $G = (N, A)$ is complete
- function c is symmetric and satisfies the triangle inequality

$$c_{ij} = c_{ji} \quad \forall i, j \in N \qquad \text{and} \qquad c_{ij} + c_{jk} \geq c_{ik} \quad \forall i, j, k \in N$$

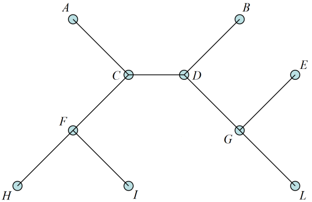
Double-tree algorithm

1. Consider the complete undirected graph corresponding to G
2. Build a minimum cost spanning tree $T^* = (N, X^*)$
3. Make a pre-order visit of T^* and build two lists of arcs:
    a. x' lists the arcs used both by the visit and the backtracking: this is a circuit visiting each node, possibly several times
    b. x lists the arcs linking the nodes in pre-order ending with the first: this is a circuit visiting each node exactly once

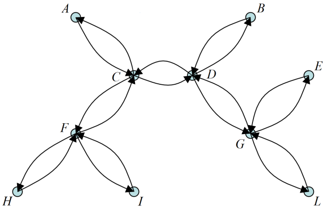
Example



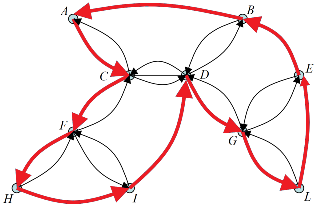
1) Complete graph G (arcs omitted)



2. Minimum spanning tree T^*



3.a) $x' = (A, C, F, H, F, I, F, C, D, G, L, G, E, G, D, B, D, C, A)$



3.b) $x' = (A, C, F, H, I, D, G, L, E, B, A)$

Proof

The double-tree algorithm is 2-approximated

1. the cost of the minimum spanning tree is an underestimate $LB(I)$
   - deleting an arc from a hamiltonian circuit yields a hamiltonian path that is cheaper
   - a hamiltonian path is a spanning tree (usually not of minimum cost)
2. the cost of circuit x' is
   - an overestimate $UB(I)$ (*it is a hamiltonian circuit, but not minimum*)
   - equal to $2LB(I)$ (*two arcs correspond to each edge*)
3. the cost of circuit x is $f_A(I) \leq UB(I)$

   (*a single direct arc replacing a sequence decreases the cost*)

This implies that $f_A(I) \leq 2f^*(I)$ for each $I \in \mathcal{I}$, that is $\alpha_A = 2$

Notice: x' is used in the approximation proof, but needs not be computed

Inapproximability

For an inapproximable problem, all approximated algorithms are exact

Consider this family of *TSP* instances violating the triangle inequality:

- $c_{ij} = 0$ for $(i, j) \in A_0 \subset A$
- $c_{ij} = 1$ for $(i, j) \in A \setminus A_0$

The optimum of any such instance $\bar{I}$ is:

$$\begin{cases} f^*\left(\bar{I}\right) = 0 \text{ if } A_0 \text{ contains a hamiltonian circuit} \\ f^*\left(\bar{I}\right) \geq 1 \text{ otherwise} \end{cases}$$

(*in the latter case, the optimal solution contains at least an arc* $\notin A_0$)

Assume that a polynomial algorithm A provide a guarantee α_A

$$f_A(I) \leq \alpha_A f^*(I) \ \forall I \in \mathcal{I}$$

Then $f^*\left(\bar{I}\right) = 0 \Leftrightarrow f_A\left(\bar{I}\right) = 0$

Whenever the subgraph $G(N, A_0)$ has a hamiltonian circuit, A finds it, solving an $\mathcal{NP}$-complete problem in polynomial time $(\mathcal{P} = \mathcal{NP})$

(*unlikely*)

Approximation schemes

For hard problems

- exact algorithms provide the best approximation guarantee ($\alpha_A = 1$), but require exponential time T_A
- approximated algorithms provide a worse guarantee ($\alpha_A > 1$), but could require polynomial time T_A

Some problems admit a whole range of different compromises between efficiency ed effectiveness

- better and better approximation guarantees: $\alpha_{A_1} > \ldots > \alpha_{A_r}$
- worse and worse computational complexities: $T_{A_1} < \ldots < T_{A_r}$

Approximation scheme is a parametric algorithm A_α allowing to choose α

(Example: the *KP*)

Beyond the worst case

As usual, the worst-case approach is rough:
some algorithms often have a good performance, though sometimes bad

The alternative approaches are similar to the ones used for complexity

- parametrization: prove an approximation guarantee that depends on other parameters of the instances besides the size n
- average-case: assume a probability distribution on the instances and evaluate the expected value of the approximation factor
(*the algorithm could have a bad performance only on rare instances*)

but there is at least another approach

- randomization: the operations of the algorithm depend not only on the instance, but also on pseudorandom numbers, so that
the solution becomes a random variable which can be investigated
(*the time complexity could also be random, but usually is not*)

Randomized approximation algorithms

For a randomized algorithm A, $f_A(I)$ and $\rho_A(I)$ are random variables

A randomized approximation algorithm has an approximation ratio whose expected value is limited by a constant

$$E[\rho_A(I)] \leq \alpha_A \text{ for each } I \in \mathcal{I}$$

Max-SAT problem: given a CNF, find a truth assignment to the logical variables that satisfy a maximum weight subset of formulae

Purely random algorithm:

Assign to each variable x_j ($j = 1, \ldots, n$)

- value *False* with probability $1/2$
- value *True* with probability $1/2$

What is the expected value of the solution?

Randomized approximation for the *MAX-SAT*

Let $\mathcal{C}_x \subseteq \{1, \ldots, m\}$ be the subset of formulae satisfied by solution x

The objective value $f(x) = f_A(I)$ is the total weight of the formulae in $\mathcal{C}_x$ and its expected value is

$$E[f_A(I)] = E\left[\sum_{i \in \mathcal{C}_x} w_i\right] = \sum_{i \in \mathcal{C}} (w_i \cdot Pr[i \in \mathcal{C}_x])$$

Let k_i be the number of literals of formula $i \in \mathcal{C}$ and $k_{\min} = \min_{i \in \mathcal{C}} k_i$

$$Pr[i \in \mathcal{C}_x] = 1 - \left(\frac{1}{2}\right)^{k_i} \geq 1 - \left(\frac{1}{2}\right)^{k_{\min}} \text{ for each } i \in \mathcal{C}$$

$$\Rightarrow E[f_A(I)] \geq \sum_{i \in \mathcal{C}} w_i \cdot \left[1 - \left(\frac{1}{2}\right)^{k_{\min}}\right] = \left[1 - \left(\frac{1}{2}\right)^{k_{\min}}\right] \sum_{i \in \mathcal{C}} w_i$$

and since $E[\rho_A(I)] = f^*(I) / E[f_A(I)]$ and $f^*(I) \leq \sum_{i \in \mathcal{C}} w_i$ for each $I \in \mathcal{I}$

one obtains

$$E[\rho_A(I)] \leq 1 / \left[1 - \left(\frac{1}{2}\right)^{k_{\min}}\right] \leq 2$$