Heuristic Algorithms

Master's Degree in Computer Science/Mathematics

Roberto Cordone

DI - Università degli Studi di Milano

Schedule:	Thursday 14.30 - 16.30 in classroom 503
	Friday 14.30 - 16.30 in classroom 503
Office hours:	on appointment
E-mail:	roberto.cordone@unimi.it
Web page:	https://homes.di.unimi.it/cordone/courses/2024-ae/2024-ae.html
Ariel site:	https://myariel.unimi.it/course/view.php?id=4466

Constructive heuristics

In Combinatorial Optimization every solution x is a subset of B

A constructive heuristic updates a subset $x^{(t)}$ step by step

1. start from an empty subset: $x^{(0)} = \emptyset$
   (*obviously a subset of any optimal solution*)
2. stop when a termination condition holds
   (*the following subsets cannot be optimal solutions*)
3. select the "best" element $i^{(t)} \in B \setminus x$ among the "acceptable" ones at the current step t
   (*try and keep $x^{(t)}$ within a feasible and optimal solution*)
4. add $i^{(t)}$ to the current subset $x^{(t)}$: $x^{(t+1)} := x^{(t)} \cup \{i^{(t)}\}$
   (*the selection can never be undone!*)
5. go back to point 2

Such processes admit a nice modelling tool

The construction graph

Every construction heuristic A defines a construction graph

- the node set $\mathcal{F}_A \subseteq 2^B$ (search space) is the collection of all subsets $x \subseteq B$ acceptable for A
- the arc set is the collection of all pairs $(x, x \cup \{i\})$ such that $x \in \mathcal{F}_A$, $i \in B \setminus x$ and $x \cup \{i\} \in \mathcal{F}_A$

The arcs represent the elementary extensions of the acceptable subsets

The construction graph is by definition acyclic

Each possible execution of A is a maximal path of the construction graph

- from the empty subset $x = \emptyset$
- to a subset x that cannot be acceptably extended

$$\Delta_A^+(x) = \{i \in B \setminus x : x \cup \{i\} \in \mathcal{F}_A\} = \emptyset$$

Termination condition

A constructive heuristic A terminates when

- the current subset $x^{(t)}$ has no outgoing arc

$$\Delta_A^+\left(x^{(t)}\right) = \left\{ i \in B \setminus x^{(t)} : x^{(t)} \cup \{i\} \in \mathcal{F}_A \right\} = \emptyset$$

- that is, extending $x^{(t)}$ implies to leave the search space $\mathcal{F}_A$

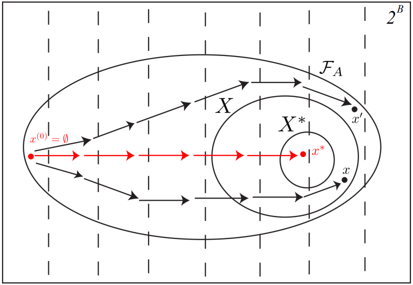
$$x^{(t)} \cup \{i\} \notin \mathcal{F}_A \text{ for each } i \in B \setminus x^{(t)}$$

Different behaviours are possible

- sometimes all visited subsets are feasible (e.g., KP)
- often the last subset is the only feasible solution
- $x^{(t)}$ could even move in and out of X (or X^*)

  (*but this is uncommon*)
- the path can visit or not X and X^*

Exact constructive algorithms

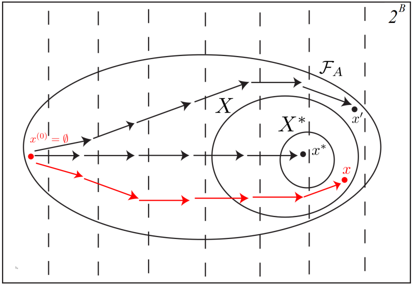


The algorithm visits a sequence of subsets $\emptyset = x^{(0)} \subset \ldots \subset x^{(t_f)}$ terminating

- in an optimal solution $x^* \in X^*$
- in a nonoptimal feasible solution $x \in X$
- in an unfeasible subset x'

Example: *MST* problem, both with $\mathcal{F}_{\text{Kruskal}}$ and $\mathcal{F}_{\text{Prim}}$

Heuristic constructive algorithms (1)

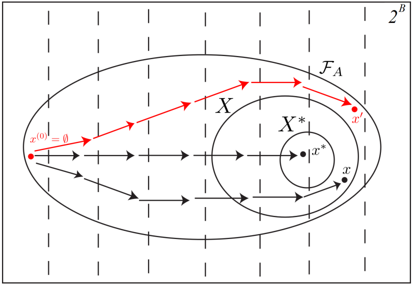


The algorithm visits a sequence of subsets $\emptyset = x^{(0)} \subset \ldots \subset x^{(t_f)}$ terminating

- in an optimal solution $x^* \in X^*$
- in a nonoptimal feasible solution $x \in X$
- in a unfeasible subset x'

Example: *KP*, *MDP*, etc...

Heuristic constructive algorithms (2)



The algorithm visits a sequence of subsets $\emptyset = x^{(0)} \subset \ldots \subset x^{(t_f)}$ terminating

- in an optimal solution $x^* \in X^*$
- in a nonoptimal feasible solution $x \in X$
- in an unfeasible subset x'

Example: *TSP* on a noncomplete graph

Pseudocode

A constructive heuristic (for minimization problems) can be described as

Algorithm Greedy(I)

$x := \emptyset$; $x^* := \emptyset$;

If $x \in X$ *then* $f^* := f(x)$ *else* $f^* := +\infty$;

While $\Delta_A^+(x) \neq \emptyset$ *do*

  $i := \arg \min_{i \in \Delta_A^+(x)} \varphi_A(i, x)$;

  $x := x \cup \{i\}$;

  If $x \in X$ and $f(x) < f^*$ *then* $x^* := x$; $f^* := f(x)$;

Return (x^*, f^*);

The path (sequence of subsets) visited by the algorithm is determined by

- the set $\Delta_A^+(x) \subseteq B \setminus x$, that is derived from the construction graph
- the selection criterium $\varphi_A : B \times \mathcal{F}_A \to \mathbb{R}$ used to select the element i to add to the current subset $x^{(t)}$ to generate $x^{(t+1)}$, that can be seen as a weight function on the arcs $(x, x \cup \{i\})$

The solution returned is the best visited during the execution

(*usually, the last one*)

Definition of the construction graph

Ideally, the search space $\mathcal{F}_A$ should include

- the empty subset: $\emptyset \in \mathcal{F}_A$ (*A starts from $\emptyset$*)
- all feasible solutions: $X \subseteq \mathcal{F}_A$
  (*maybe excluding provably nonoptimal solutions*)
- only subsets accessible from $\emptyset$ (*inaccessible subsets are useless*)

Using $\mathcal{F}_A$ requires a fast inclusion test to answer the decision problem

- "is subset $x^{(t)}$ acceptable?" ($x^{(t)} \in \mathcal{F}_A$?)

or at least a fast update test: if $x^{(t)} \in \mathcal{F}_A$, is $x^{(t)} \cup \{i\} \in \mathcal{F}_A$?

$\mathcal{F}_A = \{x' \subseteq B : \exists x \in X : x' \subseteq x\}$ (subsets of feasible solutions, i. e. partial solutions) is a natural candidate, but its inclusion test

- "is subset $x^{(t)}$ a partial solution?" ($\exists x \in X : x^{(t)} \subseteq x$?)

generalises the feasibility problem ($\exists x \in X : \emptyset \subseteq x$?)

- "is there any feasible solution?" ($\exists x \in X$?)

and could be $\mathcal{NP}$-complete

In that case, one needs to relax the search space

A natural selection criterium

If the objective function can be extended from X to $\mathcal{F}_A$,
it looks natural to use the objective function as the selection criterium

$$\varphi_A(i, x) = f(x \cup \{i\})$$

Algorithm Greedy(I)
$x := \emptyset$; $x^* := \emptyset$;
If $x \in X$ *then* $f^* := f(x)$ *else* $f^* := +\infty$;
While $\Delta_A^+(x) \neq \emptyset$ *do*
  $i := \arg \min_{i \in \Delta_A^+(x)} f(x \cup \{i\})$;
  $x := x \cup \{i\}$;
  If $x \in X$ and $f(x) < f^*$ *then* $x^* := x$; $f^* := f(x)$;
Return (x^*, f^*);

The fractional knapsack problem (*FKP*)

Select from a set of objects of identical volume a maximum value subset which could be contained in a knapsack of limited capacity

In the *FKP* the capacity simply imposes a cardinality constraint: the feasible solutions are those with $|x| \leq \lfloor V/v \rfloor$

Algorithm GreedyFKP(I)
$x := \emptyset$; $x^* := \emptyset$;
If $x \in X$ *then* $f^* := f(x)$ *else* $f^* := +\infty$;
While $|x| < \lfloor V/v \rfloor$ *do*
  $i := \arg\max_{i \in B \setminus x} \phi_i$;
  $x := x \cup \{i\}$;
  If $x \in X$ and $f(x) > f^*$ *then* $x^* := x$; $f^* := f(x)$;
Return (x^*, f^*);

- Define $\mathcal{F}_A = X$: subset x can be extended as long as $|x| < \lfloor V/v \rfloor$
- Any or no element of $B \setminus x$ extends x feasibly ($\Delta_A(x) = B \setminus x$ or $\emptyset$)
- The objective function is additive, and therefore

$$f(x \cup \{i\}) = f(x) + \phi_i \Rightarrow \arg\max_{i \in B \setminus x} f(x \cup \{i\}) = \arg\max_{i \in B \setminus x} \phi_i$$

- The last subset visited is the best solution found

Example: the fractional knapsack problem

B	a	b	c	d	e	f
ϕ	7	2	4	5	4	1

$$v_i = 1 \text{ for each } i \in B$$

$$V = 4$$

The algorithm performs the following steps:

1. $x := \emptyset$;
2. since $|x| = 0 < 4$, evaluate $i := a$ and update $x := \{a\}$;
3. since $|x| = 1 < 4$, evaluate $i := d$ and update $x := \{a, d\}$;
4. since $|x| = 2 < 4$, evaluate $i := c$ and update $x := \{a, c, d\}$;
5. since $|x| = 3 < 4$, evaluate $i := e$ and update $x := \{a, c, d, e\}$;
6. since $|x| = 4 \not< 4$, terminate

This algorithm always finds the optimal solution

But why?

The knapsack problem

Select from a set of objects of different volume a maximum value subset which could be contained in a knapsack of limited capacity

Algorithm GreedyKP(I)
$x := \emptyset$; $x^* := \emptyset$; $f^* := 0$;
While $\exists i \in B \setminus x : v_i \leq V - \sum_{j \in x} v_j$ *do*
  $i := \arg \max_{i \in B \setminus x : v_i \leq V - \sum_{j \in x} v_j} \phi_i$;
  $x := x \cup \{i\}$;
Return $(x, f(x))$;

- Define $\mathcal{F}_A = X$: only some elements of $B \setminus x$ extend x feasibly

$$\Delta_A^+(x) = \{i \in B \setminus x : \sum_{j \in x} v_j + v_i \leq V\}$$

- The objective function is additive, and therefore

$$f(x \cup \{i\}) = f(x) + \phi_i \Rightarrow \arg \max_{i \in \Delta_A^+(x)} f(x \cup \{i\}) = \arg \max_{i \in \Delta_A^+(x)} \phi_i$$

- The last subset visited is the best solution found

Example: the knapsack problem

B	a	b	c	d	e	f
ϕ	7	2	4	5	4	1
v	5	3	2	3	1	1

$$V = 8$$

The algorithm performs the following steps:

1. $x := \emptyset$;
2. since $\Delta_A^+(x) \neq \emptyset$, select $i := a$ and update $x := \{a\}$;
3. since $\Delta_A^+(x) \neq \emptyset$, select $i := d$ and update $x := \{a, d\}$;
4. since $\Delta_A^+(x) = \emptyset$, terminate

This algorithm does not find the optimal solution $x^* = \{a, c, e\}$

But why?

Example: the *MDP*

Select from a set of points a subset of k points with the maximum sum of the pairwise distances

Algorithm GreedyMDP(I)

$x := \emptyset$;

While $|x| < k$ *do*

$\quad i := \arg\max_{i \in B \setminus x} \sum_{j \in x} d_{ij}$;

$\quad x := x \cup \{i\}$;

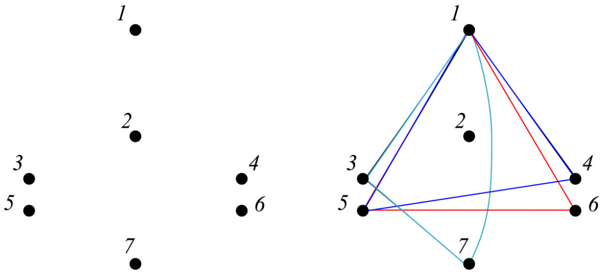
Return $(x, f(x))$;

- Define $\mathcal{F}_A$ as the set of all partial solutions
- The subset x can be extended as long as $|x| < k$
- Any element of $B \setminus x$ extends x in a feasible way
- The objective function is quadratic, and therefore

$$f(x \cup \{i\}) = f(x) + 2\sum_{j \in x} d_{ij} + d_{ii} \Rightarrow \arg\max_{i \in B \setminus x} f(x \cup \{i\}) = \arg\max_{i \in B \setminus x} \sum_{j \in x} d_{ij}$$

- The last subset visited is the best (and only feasible) solution found

Example: the Maximum Diversity Problem



The algorithm has two strong drawbacks

1. at the first step, all points are equivalent ($f(\{i\}) = 0$ for all $i \in B$)
2. the final result is nonoptimal even if
   - the first step selects the pair of farthest points (that is, $(1, 7)$)
   - the algorithm is repeated selecting each point as the first (e.g., 5)

But why?

The Travelling Salesman Problem

Given a directed graph and a cost function defined on the arcs, find a minimum cost circuit visiting all the nodes of the graph

Define $\mathcal{F}_A$ as the collection of all subsets of arcs that form no subtour and keep a degree ≤ 1 in all nodes, (it is a superset of the partial solutions)

The selection criterium is the objective function (it is additive, therefore equivalent to the cost of the new arc)

Algorithm GreedyTSP(I)

$x := \emptyset$; $x^* := \emptyset$;

$f^* := +\infty$;

While $\Delta_A^+(x) \neq \emptyset$ *do*

  $i := \arg \min_{i \in \Delta_A^+(x)} c_i$;

  $x := x \cup \{i\}$;

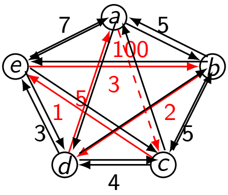
If $x \in X$ *then* $x^* := x$; $f^* := f(x)$;

Return (x^*, f^*);

Only the last subset visited can be feasible (if any!)

Example: the Travelling Salesman Problem

For the sake of simplicity consider a symmetric graph



The algorithm performs the following steps:

1. $x := \emptyset$;
2. since $\Delta_A^+(x) \neq \emptyset$, select $i := (c, e)$ and update x;
3. since $\Delta_A^+(x) \neq \emptyset$, select $i := (b, d)$ and update x ($(e, c) \notin \Delta_A^+(x)$);
4. since $\Delta_A^+(x) \neq \emptyset$, select $i := (e, b)$ and update x ($(d, b) \notin \Delta_A^+(x)$);
5. since $\Delta_A^+(x) \neq \emptyset$, select $i := (d, a)$ and update x:
   notice that (b, e), (d, e), (e, d), (c, d) and $(d, c) \notin \Delta_A^+(x)$!
6. since $\Delta_A^+(x) = \emptyset$, terminate

The algorithm does not find a feasible solution

Adding arc (a, c) with cost 100, it finds a feasible, but nonoptimal, solution

But why?

In summary

A constructive heuristic A finds

- the optimum when $\Delta_A^+\left(x^{(t)}\right)$ and $\varphi_A(i,x)$ guarantee that the current subset $x^{(t)}$ is always included in at least one optimal solution
- a feasible solution when $\Delta_A^+\left(x^{(t)}\right)$ guarantees that the current subset $x^{(t)}$ is always included in at least one feasible solution
- a general subset when these properties are lost at some step t

An ideal constructive algorithm always keeps one open way to the optimal solution

In practice, some of these properties is usually lost at some step of the algorithm

Relevant features

What features allow a constructive algorithm to find the optimum?

- A search space identical to the feasible region ($\mathcal{F} = X$)?
  (*No, because this holds for both the fractional and general KP*)
- A cardinality-constrained problem?
  (*It would explain failing on the KP, but not on the MDP and TSP*)
- An additive objective function?
  (*It does not explain failing on the TSP*)

There is no general characterization of the problems solved exactly by constructive algorithms

But there are characterisations for wide classes of problems

A characterization in the additive case

Assume that

1. the objective function be additive

$$\exists \phi : B \rightarrow \mathbb{N} : f(x) = \sum_{i \in x} \phi_i$$

2. the solutions be the bases (maximal subsets) of the search space

$$X = \mathcal{B}_{\mathcal{F}} = \{Y \in \mathcal{F} : \nexists Y' \in \mathcal{F} : Y \subset Y'\}$$

It is a very frequent case (*KP*, *MAX-SAT*, *TSP*, but not *MDP*, *SCP*)

In this case, the constructive algorithm always finds the optimal solution if and only if $(B, \mathcal{F})$ is a *matroid embedding*

Since the definition of *matroid embedding* is rather complex,
let us focus on some important structures

1. greedoids (necessary condition)
2. matroids or greedoids with the strong exchange property (sufficient)

Greedoids

A greedoid $(B, \mathcal{F})$ with $\mathcal{F} \subseteq 2^B$ is a pair such that

- trivial axiom: $\emptyset \in \mathcal{F}$ *The empty set is acceptable*
- accessibility axiom:
  if $x \in \mathcal{F}$ and $x \neq \emptyset$ then $\exists i \in x : x \setminus \{i\} \in \mathcal{F}$
  Any acceptable subset can be built adding elements in suitable order
- exchange axiom: if $x, y \in \mathcal{F}$ with $|x| = |y| + 1$,
  then $\exists i \in x \setminus y$ such that $y \cup \{i\} \in \mathcal{F}$
  Any acceptable subset can be extended with a suitable element of any other acceptable subset of larger cardinality

The exchange axiom implies that all bases have the same cardinality

All of these conditions

- hold in the fractional *KP*, *MST* problem (both Kruskal and Prim),...
- do not hold in the general *KP*, *TSP*...
- hold in the *MDP*, but the objective function is not additive

Greedoids make greedy algorithms possible, but not necessarily exact

Matroids

A matroid is a set system $(B, \mathcal{F})$ with $\mathcal{F} \subseteq 2^B$ such that

- trivial axiom: $\emptyset \in \mathcal{F}$
- hereditarity axiom: if $x \in \mathcal{F}$ and $y \subset x$ then $y \in \mathcal{F}$
  Any acceptable subset can be built adding its elements in any order
- exchange axiom: if $x, y \in \mathcal{F}$ with $|x| = |y| + 1$,
  then $\exists i \in x \setminus y$ such that $y \cup \{i\} \in \mathcal{F}$
  Any acceptable subset can be extended with a suitable element of any other subset of larger cardinality

The hereditarity axiom is a stronger version of accessibility

- it holds in Kruskal's search space for the *MST* problem
- it does not hold in Prim's search space for the *MST* problem

We already know some examples of matroids

Uniform matroid: fractional and general knapsack

$$\mathcal{F} = \{x \subseteq B : |x| \leq \lfloor V/v \rfloor\}$$

- Trivial axiom: the empty set respects the cardinality constraint
- Hereditarity axiom: if x respects the cardinality constraint, all of its subsets also respect it
- Exchange axiom: if x and y respect the cardinality constraint and $|x| = |y| + 1$, one can always add a suitable element of x to y without violating the cardinality (*in fact, any element of* x)

For the general KP the first two axioms hold, but the third one does not

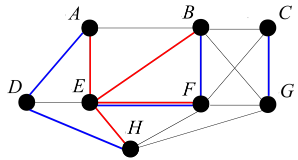
Example:

If $V = 6$ and $v = [\,3\;3\;2\;2\;1\,]$, the subsets $x = \{3, 4, 5\}$ and $y = \{1, 2\}$ are in $\mathcal{F}$, but no element of x can be added to y

Graphic matroid: minimum spanning tree

$$\mathcal{F} = \{x \subseteq B : x \text{ forms no cycles } \}$$

- Trivial axiom: the empty set of edges forms no cycles
- Hereditarity axiom: if x is acyclic, all of its subsets are acyclic
- Exchange axiom: if x and y are acyclic and $|x| = |y| + 1$, one can always add a suitable edge of x to y without forming any cycle
  (*not all edges of x work*)



$x = \{(A, D), (D, H), (E, F), (B, F), (C, G)\}$

$y = \{(A, E), (B, E), (E, F), (E, H)\}$

(A, D), (D, H) and (C, G) can be added to y

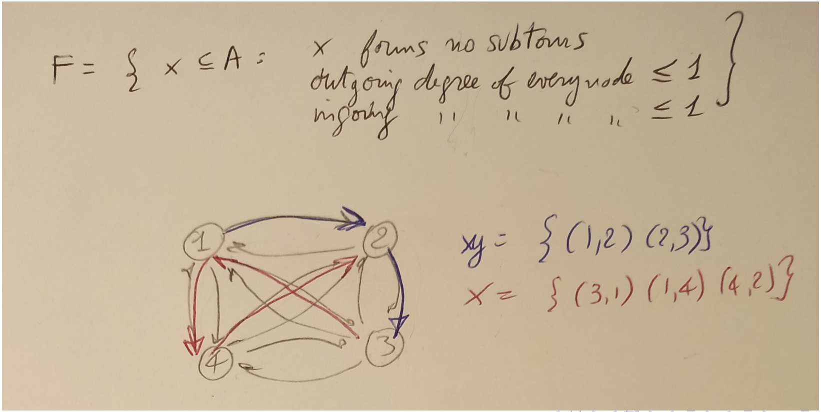
Travelling Salesman Problem

For the *TSP* the first two axioms hold

- the empty set has no subtours and degrees ≤ 1
- any proper subset of a set $\in \mathcal{F}$ (no subtours and degrees ≤ 1) also belongs to $\mathcal{F}$

but the third axiom is violated

Example: $y = \{(1,2),(2,3)\}$ and $x = \{(3,1),(1,4),(4,2)\}$



No arc of x can be added to y remaining in $\mathcal{F}$

Greedoids with the strong exchange axiom

The optimality of the greedy algorithm can be proved for greedoids (weaker second axiom) if the exchange axiom is strengthened

- strong exchange axiom:

$$\begin{cases} x \in \mathcal{F}, y \in \mathcal{B}_{\mathcal{F}} \text{ such that } x \subseteq y \\ i \in B \setminus y \text{ such that } x \cup \{i\} \in \mathcal{F} \end{cases} \Rightarrow \exists j \in y \setminus x : \begin{cases} x \cup \{j\} \in \mathcal{F} \\ y \cup \{i\} \setminus \{j\} \in \mathcal{F} \end{cases}$$

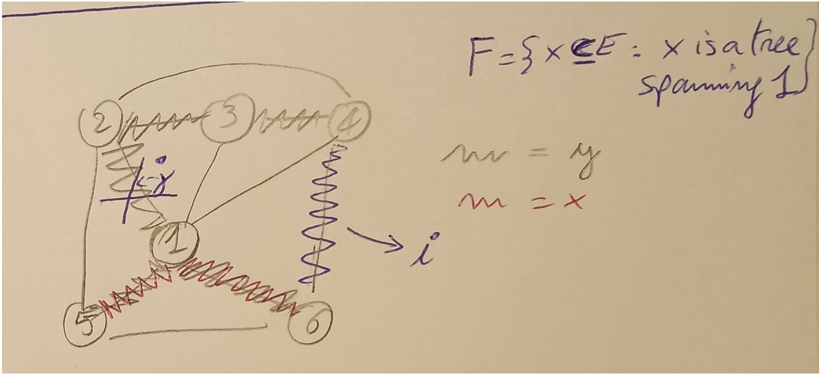
Given a basis and one of its subsets (from which the basis is accessible), if there is an element that "leads astray" the subset from the base, there must be another one which keeps it on the right way and it must be feasible to exchange the two elements in the basis

Greedoids with the strong exchange axiom: *MST*

A classical example of greedoid with strong exchange axiom is given by

- B = edge set of a graph
- $\mathcal{F}$ = collection of the trees including a given vertex v_1

that yields Prim's algorithm for the *MST* problem



The trivial and the accessibility axiom hold (the hereditary one does not)
The exchange axiom holds in the strong form

Optimal constructive algorithms

Notice that the optimality of a constructive algorithm A depends on

- the properties of the problem (e. g., additive objective function, bases as feasible solutions)
- the properties of the search space $\mathcal{F}_A$ (that is, of the algorithm)