THE OPTIMAL PATH PROBLEM

Ernesto de Queirós Vieira Martins
Marta Margarida Braz Pascoal
Deolinda Maria Lopes Dias Rasteiro
José Luis Esteves dos Santos

{*eqvm,marta,dml,zeluis*}*@mat.uc.pt*
Departamento de Matemática
Universidade de Coimbra
PORTUGAL

July 1998

Abstract:

In this paper the optimal path problem is studied from a global point of view. Emphasis is given to the optimality principle since it allows us to design a class of algorithms – the labeling algorithms – which determine an optimal path when the weak optimality principle is satisfied. This property is reinforced with two problems which are similar in their description and completly different in what concerns to their resolution.

Keywords: network, path, optimality principle, labeling algorithms.

1 Introduction

In the optimal path problem a function is given that assigns a value to each path that can be defined between a given pair of nodes of a given network. Among all the paths between that pair of nodes it has to be determined the one with the *best* value.

Two classes of optimal path problems can be considered: the unconstrained path problems and the constrained ones. While in the former no constraints are considered, in the last one the optimal path has to satisfy a given set of constraints, [1, 14]. As an example of the constrained optimal path problem we point out the classical traveling salesman problem, [11], in which only paths defined from some node to itself and passing through each node exactly once can be considered as feasible solutions.

Constrained optimal path problems differ between them only in the given function definition. Among the unconstrained problems we refer the classical shortest path and the maximum capacity path problems.

Different types of problems can also be considered. For example, ranking path problems, [13, 17], and multiobjective path problems, [18]. In the former, it is intended to determine not only the *best* path but also the K *best* paths by order of their values. In the multiobjective path problem a set of paths has to be determined, being all of then considered optimal.

2 Notation and definitions

Let $(\mathcal{N},\mathcal{A})$ denote a given network, where $\mathcal{N}=\{v_1,\dots,v_n\}$ is a finite set with n elements called nodes and $\mathcal{A}=\{a_1,\dots,a_m\}\subseteq\mathcal{N}\times\mathcal{N}$ is a finite set with m elements called arcs. Sometimes it will simply be used i to represent node v_i. Each arc $a_k \in \mathcal{A}$ can also be identified by a pair (i, j), where $i, j \in \mathcal{N}$.

Let i and j be two nodes of $(\mathcal{N}, \mathcal{A})$; if all pairs (i, j) in $(\mathcal{N}, \mathcal{A})$ are ordered, i.e., if all the arcs of the network are directed, $(\mathcal{N}, \mathcal{A})$ is said to be a directed network; if all the pairs (i, j) are not ordered, then $(\mathcal{N}, \mathcal{A})$ is said to be an undirected network.

Without loss of generality, in what follows it will be considered that $(\mathcal{N}, \mathcal{A})$ is a directed network; it will also be assumed that there is at most a single arc between each pair of nodes of the network and that there are no arcs of the form (i, i), where $i \in \mathcal{N}$.

Let s and t be two different nodes of $(\mathcal{N}, \mathcal{A})$, called initial and terminal node, respectively. A path from node s to node t in $(\mathcal{N}, \mathcal{A})$ is an alternating sequence of nodes and arcs, of the form $p = \langle s = v'_1, a'_1, v'_2, \ldots, a'_{\ell-1}, v'_\ell = t \rangle$, where $\ell \geq 2$, and:

- $v'_k \in \mathcal{N}$, for every $k \in \{1, \ldots, \ell\}$;
- $a'_k \equiv (v'_k, v'_{k+1}) \in \mathcal{A}$, for every $k \in \{1, \ldots, \ell - 1\}$.

For convenience, sometimes a single node is viewed as a path.

Given $i, j \in \mathcal{N}$, $\mathcal{P}_{ij}$ denotes the set of paths defined from i to j in $(\mathcal{N}, \mathcal{A})$. In order to simplify, $\mathcal{P}$ will denote the set $\mathcal{P}_{st}$. In what follows it will be assumed that $\mathcal{P}_{si}$ and $\mathcal{P}_{it}$, for every node i in $(\mathcal{N}, \mathcal{A})$, are assumed to be nonempty.

A loopless path from s to t is a path from s to t where all the nodes are different and a loop (or cycle) is a path from a node to itself, where all nodes are different except the first which is also the last.

Let $p \in \mathcal{P}_{ux}$ be a path from u to x and $q \in \mathcal{P}_{xv}$ a path from x to v. The concatenation of p and q, denoted by $p \diamond q$, is a path from u to v formed by p until its node x and followed by path q from x to v. Given a cycle $\mathcal{C}$ it can be defined the k-cycle concatenation by $\mathcal{C}^k = \mathcal{C}^{k-1} \diamond \mathcal{C}$, with $k \geq 1$ and $\mathcal{C}^0$ a single node of cycle $\mathcal{C}$.

A path is said to be finite if it has a finite number of nodes (and arcs).

3 The optimal path problem

In the optimal path problem a directed network $(\mathcal{N}, \mathcal{A})$ is given, where $r \geq 1$ real values, $c^1_{ij}, \ldots, c^r_{ij}$, are associated to each arc $(i, j) \in \mathcal{A}$. These values are denominated costs of (i, j). It will also be considered a real function $f : \mathcal{P} \longrightarrow \mathbb{R}$, denominated objective function, such that, for a given path p, $f(p)$ depends on the costs associated to the arcs of p.

It is intended to determine a path $p^* \in \mathcal{P}$ that optimizes some given function f in $\mathcal{P}$, i.e., such that $f(p^*) = \text{opt}\{f(p) : p \in \mathcal{P}\}$. Path p^* is denominated an optimal solution for such a problem.

3.1 Boundness and finiteness

One optimal path problem is said to be finite if there is an optimal path with a finite number of nodes; otherwise the problem is said to be non finite.

When one optimal path problem is non finite it is considered that its optimal solution is the limit of a sequence of paths, which is associated to a sequence of objective function values converging to the optimal value.

A non finite optimal path problem is bounded if for every optimal solution p^*, $f(p^*)$ converges to a finite value; otherwise the problem is said to be unbounded.

3.2 Examples of optimal path problems

In this section three examples of optimal path problems will be presented. In the first two cases it will be considered that $r = 1$, i.e., to each arc of the network is associated one single value; in the last problem, $r = 2$.

3.2.1 The shortest path problem

The shortest path problem is a classical network optimization problem, which has been intensively studied since the fifty's. Hundreds of papers can be found in specialized literature; with no purposes of selection the following are mentioned, [2, 3, 4, 5, 6, 7, 8, 9, 12, 15, 16, 19].

To each arc (i, j) of $(\mathcal{N}, \mathcal{A})$ it is associated the real value $c_{ij} \in \mathbb{R}$, called cost or distance of (i, j). It is still considered the objective function defined by

$$c : \mathcal{P} \longrightarrow \mathbb{R} : p \mapsto c(p) = \sum_{(i,j)\in p} c_{ij} = \sum_{p} c_{ij}.$$

In the shortest path problem it is intended to determine a path p^* from s to t in $(\mathcal{N}, \mathcal{A})$ such that $c(p^*)$ is minimum, i.e., such that $c(p^*) \leq c(p)$, for every path $p \in \mathcal{P}$.

In Theorem 1 it will be proved that the shortest path problem is finite if and only if there are no negative cycles (cycles with negative cost) in the given network.

Theorem 1 *Let us assume that $\mathcal{P}_{si}$ $\mathcal{P}_{it}$ are nonempty sets, for every $i \in \mathcal{N}$. The shortest path problem is finite if and only if $c(\mathcal{C}) \geq 0$, for every cycle $\mathcal{C}$ of $(\mathcal{N}, \mathcal{A})$.*

Proof: Let $\mathcal{C}$ be a cycle such that $c(\mathcal{C}) < 0$ and let i be a node of $\mathcal{C}$. Since $\mathcal{P}_{si}$ and $\mathcal{P}_{it}$ are nonempty, there exist p_1, path from s to i, and p_2, from i to t, in $(\mathcal{N}, \mathcal{A})$. Denoting by $p_1 \diamond \mathcal{C}^0 \diamond p_2$ the path $p_1 \diamond p_2$, we conclude that $p_k = p_1 \diamond \mathcal{C}^k \diamond p_2$, with $k \geq 0$, is a sequence of paths from s until t and $c(p_k) = c(p_1) + c(\mathcal{C}^k) + c(p_2)$. It follows that

$$\lim_{k\to\infty} c(p_k) = c(p_1) + c(\mathcal{C}) \lim_{k\to\infty} k + c(p_2) = -\infty.$$

Then, the optimal path is $p_\infty = \lim_{k\to\infty} p_k$, which means that p_∞ is not finite. Since c_{ij} is finite for every arc (i, j) we conclude that the shortest path problem is not finite.

Let us consider now that $c(\mathcal{C}) \geq 0$, for every cycle $\mathcal{C}$ of $(\mathcal{N}, \mathcal{A})$, and let p^* be an optimal path.

If p^* has no cycles then p^* is a loopless path; so it has no repeated nodes and it is finite too.

Considering now that p^* has at least one cycle, then $p^* = p_1 \diamond \mathcal{C} \diamond p_2$. If $c(\mathcal{C}) > 0$, $c(p^*) > c(p_1 \diamond p_2)$ and then p^* would not be optimal. So, it can be concluded that $c(\mathcal{C}) = 0$; but since $c(p^*) = c(p_1 \diamond p_2)$ then $p_1 \diamond p_2$ is also an optimal path, and p^* has one more cycle than $p_1 \diamond p_2$. Repeating this procedure it is possible to remove all cycles from p^*, and a loopless path is obtained, which is still an optimal solution of the problem. Once again it can be concluded that this is a finite solution. □

The above theorem yields the following results.

Corollary 1.1 *If the shortest path problem is not finite, then it is also unbounded.*

Proof: If the shortest path problem is not finite, then there is a negative cycle, $\mathcal{C}$, in $(\mathcal{N}, \mathcal{A})$. Let us consider $p_\infty = \lim_{k\to+\infty} p_k$, with $p_k = p_1 \diamond \mathcal{C}^k \diamond p_2$, where $p_1 \in \mathcal{P}_{si}$, $p_2 \in \mathcal{P}_{it}$ and where i is a node of $\mathcal{C}$. Then $c(p_\infty) = -\infty$ and the shortest path problem is unbounded. □

Corollary 1.2 *If $c(\mathcal{C}) \geq 0$, for every cycle $\mathcal{C}$ in $(\mathcal{N}, \mathcal{A})$, then there is a loopless path which is an optimal solution for the shortest path problem.*

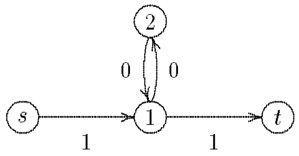

Figure 1: Network $(\mathcal{N}, \mathcal{A})$

Notice that, even if there are no negative cycles in $(\mathcal{N}, \mathcal{A})$, there can still exist non finite optimal paths, as long as there is a cycle with zero cost with a node belonging to a shortest path, as we can see in Figure 1.

In fact, the cost of the shortest path from s to t in $(\mathcal{N}, \mathcal{A})$ is 2, and there are several shortest paths in this network. Some of these paths are finite, e.g. $\langle s, 1, t\rangle$, which is also a loopless path, while $\lim_{k \to +\infty} \langle s, 1\rangle \diamond \mathcal{C}^k \diamond \langle 1, t\rangle$, where $\mathcal{C} = \langle 1, 2, 1\rangle$, is not finite, but since $c(\mathcal{C}) = 0$ still has cost 2.

3.2.2 The maximum capacity path problem

To each arc (i, j) of $(\mathcal{N}, \mathcal{A})$ is associated the value $u_{ij} \in \mathbb{R}^+$, called capacity of the arc (i, j). The objective function is defined by

$$u : \mathcal{P} \longrightarrow \mathbb{R}^+ : p \mapsto u(p) = \min_{(i,j) \in p} u_{ij} = \min_{p} u_{ij}.$$

In the maximum capacity path problem, [10], it is intended to determine a path p^* from s to t in $(\mathcal{N}, \mathcal{A})$, such that $u(p^*)$ is maximum, i.e., such that $u(p^*) \geq u(p)$, for every path $p \in \mathcal{P}$.

In Theorem 2 it will be proved that the maximum capacity path problem is always finite, which does not depend on the given network.

Theorem 2 *Let us assume that $\mathcal{P}_{si}$ and $\mathcal{P}_{it}$ are nonempty stes, for every $i \in \mathcal{N}$. The maximum capacity path problem is finite for every considered network $(\mathcal{N}, \mathcal{A})$.*

Proof: Let p^* be an optimal solution for the maximum capacity path problem in a given network $(\mathcal{N}, \mathcal{A})$. Then $u(p^*) \geq u(p)$, for every path $p \in \mathcal{P}$.

If p^* does not contain cycles then it is a loopless path and the problem is finite.

Let us assume now that p^* has at least one cycle, i.e., $p^* = p_1 \diamond \mathcal{C} \diamond p_2$. Then

$$u(p^*) = \min\{u(p_1), u(\mathcal{C}), u(p_2)\} \leq \min\{u(p_1), u(p_2)\} = u(p_1 \diamond p_2).$$

Since p^* is a maximum capacity path then $u(p^*) = u(p_1 \diamond p_2)$ and p^* has one more cycle than $p_1 \diamond p_2$; so $p_1 \diamond p_2$ is also an optimal path. Repeating this procedure it is possible to remove all cycles from p^*, and thus obtain an optimal solution that is a loopless path, which is finite. $\square$

The following result is now immediate.

Corollary 2.1 *Under the assumptions of Theorem 2, there is a loopless path that is an optimal solution for the maximum capacity path problem.*

Notice that similarly to the shortest path problem, non finite optimal solutions can exist for the maximum capacity path problem.

3.2.3 The shortest path per unit of time problem

To each arc (i,j) of $(\mathcal{N},\mathcal{A})$ it is associated a pair of values $(c_{ij},t_{ij}) \in \mathbb{R} \times \mathbb{R}^+$, denominated, respectively, cost and time of arc (i,j). In the shortest path per unit of time problem it is intended to determine a path that minimizes an objective function definided by

$$f : \mathcal{P} \longrightarrow \mathbb{R} : p \mapsto f(p) = \frac{\sum_p c_{ij}}{\sum_p t_{ij}}.$$

As Figure 2 shows, this problem can be no finite and bounded simultaneously.

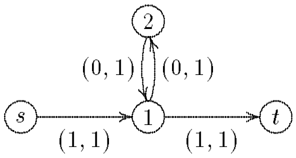

Figure 2: Network $(\mathcal{N},\mathcal{A})$

In fact, the optimal solution for the problem of the shortest path per unit of time from s to t in $(\mathcal{N},\mathcal{A})$ is $p_\infty = \lim_{k\to+\infty} p_k$, where $p_k = \langle s,1\rangle \diamond \langle 1,2,1\rangle^k \diamond \langle 1,t\rangle$, which is not finite. However, as $c(p_k) = \frac{1}{1+k}$, path $p_\infty = \lim_{k\to\infty} p_k$ has finite cost since $c(p_\infty) = \lim_{k\to\infty} \frac{1}{1+k} = 0$.

4 The optimality principle

Some optimal path problems verify one or two principles which are known as the strong and weak optimality principle and are stated bellow.

Strong optimality principle: *Every optimal path is formed by optimal subpaths.*

Weak optimality principle: *There is an optimal path formed by optimal subpaths.*

According to these principles, an optimal solution can be determined by finding successive optimal subsolutions. So, when an optimization problem verifies the strong optimality principle a labeling algorithm can be used to solve it. However, labeling algorithms can be applied if there is only one optimal path under those conditions, and they do determine this optimal path. Then, it can be concluded that if a problem verifies the weak optimality principle it still can be solved using a labeling algorithm.

It can be easily noted that an optimal path problem which verifies the strong optimality principle also satisfies the weak optimality principle.

In what follows conditions will be established in order to assure that the shortest path problem verifies these two principles (so it can be solved by a labeling algorithm).

Theorem 3 *The shortest path problem satisfies the strong optimality principle if and only if there are no negative cycles in $(\mathcal{N},\mathcal{A})$.*

Proof: Let us assume that there exists a negative cycle $\mathcal{C}$ in $(\mathcal{N},\mathcal{A})$. As it was refered in the proof of Theorem 1, $p_\infty = \lim_{k\to+\infty} p_k$, where $p_k = p_1 \diamond \mathcal{C}^k \diamond p_2$, is an optimal solution for this problem. Then, $q_k = p_1 \diamond \mathcal{C}^k$ is a subpath of p_∞, although it is not an optimal subpath once $c(q_k) > c(q_{k+1})$, and then in this case the shortest path problem does not satisfy the strong optimality principle.

Let us now consider that $c(\mathcal{C}) \geq 0$, for every cycle $\mathcal{C}$ in $(\mathcal{N},\mathcal{A})$. Then the shortest path problem is finite and we can choose an optimal path, p^*, and consider that it contains a non optimal subpath,

$q \in \mathcal{P}_{uv}$; i.e., $c(p^*) \leq c(p)$, for every path $p \in \mathcal{P}$, and $p^* = p_1 \diamond q \diamond p_2$. Let q^* be an optimal path from u until v; then $c(q^*) < c(q)$. Considering now the path $w = p_1 \diamond q^* \diamond p_2 \in \mathcal{P}$, since $c(w) < c(p^*)$ it can be concluded that p^* would not be an optimal solution as assumed. □

Corollary 3.1 *The shortest path problem satisfies the weak optimality principle if and only if there are no negative cycles in* $(\mathcal{N}, \mathcal{A})$.

It can also be noted that the maximum capacity path problem verifies the weak optimality principle (and therefore it can also be solved by a labeling algorithm).

Theorem 4 *The maximum capacity path problem satisfies the weak optimality principle.*
Proof: Let p^* be an optimal finite solution for the maximum capacity path problem, i.e., $u(p^*) \geq u(p)$, for every path $p \in \mathcal{P}$. Based on Corollary 2.1 we can assume that p^* is a loopless path. Let us suppose that p^* has a subpath, $q \in \mathcal{P}_{uv}$, which is not optimal, i.e., $p^* = p_1 \diamond q \diamond p_2$. Then, there is a maximum capacity path from u to v in $(\mathcal{N}, \mathcal{A})$, q^*, which is loopless, and

$$u(p_1 \diamond q^* \diamond p_2) = \min\{u(p_1), u(q^*), u(p_2)\} \geq \min\{u(p_1), u(q), u(p_2)\} = u(p^*).$$

This means that $p_1 \diamond q^* \diamond p_2$ is also an optimal solution for this problem. If all the subpaths of q^* are optimal the new path has one more optimal subpath than p^* and this process can be repeated until a path that contains only optimal subpaths is obtained. If q^* still contains non optimal subpaths it is also possible to remove them in a similar way. Since p^* is a loopless path and there is a finite number of loopless paths in $(\mathcal{N}, \mathcal{A})$ this process is finite too and when finished an optimal path will be obtained, containing only optimal subpaths. □

In general, the maximum capacity path problem does not satisfy the strong optimality principle which will be shown in the problem described in Figure 3.

The following figures exemplify these results. The first one, Figure 3, represents a network $(\mathcal{N}, \mathcal{A})$, where to each arc (i, j) is associated a pair $(c_{ij}, u_{ij}) \in \mathbb{R} \times \mathbb{R}^+$; that is, the first value of this pair represents the cost of (i, j) and the second one represents its capacity.

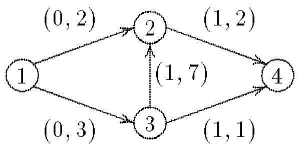

Figure 3: Network $(\mathcal{N}, \mathcal{A})$

The second figure, Figure 4, represents the tree of all the paths that can be defined from 1 to 4 in $(\mathcal{N}, \mathcal{A})$. To each node i in this tree is associated a pair where the first element represents the cost of the path in the tree from s to i, and the second its capacity.

Analysing the shortest path problem in this network it can be remarked that there are two optimal solutions, namely $\langle 1, 2, 4 \rangle$ and $\langle 1, 3, 4 \rangle$. As it can easily be verified, all the subpaths of this two paths are also optimal paths.

There are also two optimal solutions for the maximum capacity path in $(\mathcal{N}, \mathcal{A})$, namely paths $\langle 1, 2, 4 \rangle$ and $\langle 1, 3, 2, 4 \rangle$. Considering the second one, its subpath from 1 until 2 is $\langle 1, 2 \rangle$, which has 2 as capacity, while the capacity of $\langle 1, 3, 2 \rangle$ is 3. So, there is one optimal solution that is not formed only by optimal subpaths. However, every subpath of the optimal path $\langle 1, 2, 4 \rangle$ is optimal.

Finaly the third figure, Figure 5, represents the tree of all paths in $(\mathcal{N}, \mathcal{A})$ from 1 to 4, where to each node i it is associated the cost per unit of time – considering u_{ij} as the time for traversing (i, j) from i to j – of the path in the tree from s until i.

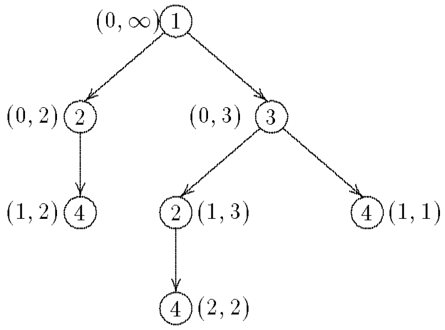

Figure 4: Tree of the paths from 1 to 4 in $(\mathcal{N}, \mathcal{A})$

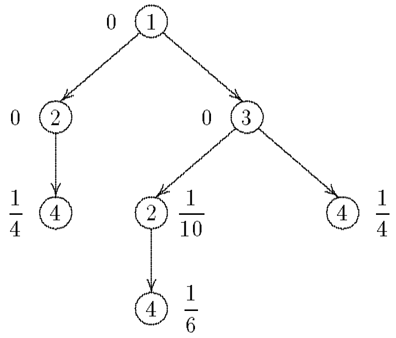

Figure 5: Tree of the paths from 1 to 4 in $(\mathcal{N}, \mathcal{A})$

Path $\langle 1, 3, 2, 4\rangle$ is the only optimal solution for the shortest path per unit of time problem. Its subpath from 1 until 2 is, $\langle 1, 3, 2\rangle$; however the optimal path between those two nodes is $\langle 1, 2\rangle$.

So, the unique optimal solution for this problem has one subpath that is not optimal, i.e, the shortest path per unit of time problem does not satisfy neither the strong nor the weak, optimality principle.

4.1 Labeling algorithms

In what follows only finite problems will be considered since only in this case it is possible to determine computationally a solution for an optimal path problem. Moreover the problem must satisfy the weak optimality principle. In fact, as remarked before, there is a class of algorithms, named labeling algorithms, which can solve some optimal path problems. These algorithms determine a tree rooted at s, where the path from s until i is optimal, for every node $i \in \mathcal{N}$. This tree is named optimal tree rooted at s and contains the optimal path from s until t that we intended to determine.

These algorithms try to construct this tree by determining optimal subpaths. With this purpose there is a label, π_i, associated to each node $i \in \mathcal{N}$, that represents the value of the *best* path determined from s to i at a certain moment. This path will be denoted by p_i^*. It is also used a set X, containing nodes of $(\mathcal{N}, \mathcal{A})$ with labels that can eventually be improved.[1]

Initially the set X contains only the initial node, s. The algorithm finishes when there are no more labels to improve, i.e., when X is empty. Whenever a node i is picked out from X and then analysed, the labels of other nodes in $(\mathcal{N}, \mathcal{A})$ are improved if possible.

After the execution of the algorithm p_i^* represents the optimal path from s to i and π_i^* represents its cost.

[1] Note that the concept of *best* depends on the optimal path problem that is being considered.

At the beginning of the algorithm, for every $i \in \mathcal{N} - \{s\}$, π_i^* should have a value that can be improved, i.e., an upper or lower bound for all the possible values that π_i^* can assume, which depends on the optimization problem. Thus, in the shortest path problem, π_i^* is initialized with the value $+\infty$, while in the maximum capacity path problem π_i^* is initialized with 0, for every $i \in \mathcal{N} - \{s\}$. Node s is initialized with the value $f(\langle s \rangle)$, i.e., the optimal value of a path from s to s.

This class of algorithms is decribed in Algorithm 1.

Labeling algorithms can be divided into two subclasses, namely, the label correcting algorithms and the label setting algorithms.

4.1.1 Label correcting algorithms

Labeling algorithms can be implemented in several different ways, which depends on the data structure used to represent the set X. This structure determines the node to be removed at each step of the algorithm and how X is manipulated.

When the node i to be picked out from X is not specified, its label can be improved later on and consequently i is placed again in X. In this case the algorithm is said to be a label correcting algorithm.

As refered before several versions of these algorithms can be implemented, by using different structures for X. The interested reader is adressed to [3, 4] for details.

Algorithm 1: *Label correcting algorithm for the optimal path problem*

`For` $(i \in \mathcal{N} - \{s\})$ `Do` $\pi_i^* \longleftarrow \pm\infty$;
$p_s^* \longleftarrow \langle s \rangle$;
$\pi_s^* \longleftarrow f(\langle s \rangle)$;
$X \longleftarrow \{s\}$;
`While` $(X \neq \emptyset)$ `Do`
  $i \longleftarrow$ node of X;
  $X \longleftarrow X - \{i\}$;
  `For` $((i,j) \in \mathcal{A}$ such that $f(p_i^* \diamond \langle i,(i,j),j \rangle)$ is *better* than $\pi_j^*)$ `Do`
    $p_j^* \longleftarrow p_i^* \diamond \langle i,(i,j),j \rangle$;
    $\pi_j^* \longleftarrow f(p_i^* \diamond \langle i,(i,j),j \rangle)$;
    `If` $(j \notin X)$ `Then` $X \longleftarrow X \cup \{j\}$;
  `EndFor`
`EndWhile`

4.1.2 Label setting algorithms

When a node is picked out from X the labels of several other nodes can be changed for a *better* value. So it is natural to think of choosing the element in X which has the *best* cost at each step of the labeling algorithm. In this way, instead of just removing an element of X, an element i will be removed such that π_i^* is *better* than π_j^*, for every $j \in X$.

A labeling algorithm where each node is chosen in this way is said to be a label setting algorithm. This means that once a node is picked out from set X, its label is permanent, i.e., it will not be improved again. In order to assure that this happens it will be assumed that all the nodes labeled from i have a label that is not *better* that π_i^*. In Theorem 5 will be proved that in this case the label of every node removed from X is permanent.

Theorem 5 *Let us assume that the label of every node obtained from $i \in \mathcal{N}$ is not* better *than the label of i. Then, whenever a node is picked out from set X, its label is permanentu when a label setting algorithm is used.*

Proof: Let us assume that i is a node removed from X and that its label is not permanent, i.e., there is a node j picked out from X, such that, π_i^* can be improved from π_j^*.

When i was chosen in X, if j was in X, then π_j^* was not *better* than π_i^*. Otherwise, if in that moment j was in X, then j was labelled, directly or indirectly, from a node x in X in that step, which means, once again, that π_x^* was not *better* than π_i^*, which proves the theorem. □

Algorithm 2: *Label setting algorithm for the optimal path problem*

```
For (i ∈ 𝒩 − {s}) Do π_i* ⟵ ±∞;
p_s* ⟵ ⟨s⟩;
π_s* ⟵ f(⟨s⟩);
X ⟵ {s};
While (X ≠ ∅) Do
   i ⟵ node of X such that π_i* is better than π_j*, for every j ∈ X;
   X ⟵ X − {i};
   For ((i, j) ∈ 𝒜 such that f(p_i* ◇ ⟨i, (i, j), j⟩) is better than π_j*) Do
      p_j* ⟵ p_i* ◇ ⟨i, (i, j), j⟩;
      π_j* ⟵ f(p_i* ◇ ⟨i, (i, j), j⟩);
      If (j ∉ X) Then X ⟵ X ∪ {j};
   EndFor
EndWhile
```

5 Two optimal path problems

To reinforce the importance of the optimality principle in the resolution of optimal path problems, two new problems will be described.

Although similar in their description, these problems are completly different in what concerns the determination of the optimal solution. They are based on two of the problems presented before – the shortest path problem and the maximum capacity path problem; in these two new problems it is intended to determine paths from s to t which are optimal in a subset of $\mathcal{P}$.

As refered above, although both problems – shortest path and maximum capacity path problem – satisfy the weak optimality principle, only the first one satisfies the strong optimality principle. In the following sections it will be proved that this property influences the fact of the new problems to be described satisfying or not these principles.

5.1 The problem of the maximum capacity path in the set of the shortest paths

Let $\mathcal{P}^*$ denote the set of the shortest paths from s to t, i.e., $\mathcal{P}^* = \{p^* \in \mathcal{P} : c(p^*) \leq c(p), \forall p \in \mathcal{P}\}$. It is immediate that $\mathcal{P}^* \subseteq \mathcal{P}$.

In the problem of the maximum capactity path in the set of the shortest paths it is intended to determine a maximum capacity path in $\mathcal{P}^*$, i.e., it is intended to determine a path $\bar{p} \in \mathcal{P}^* \subseteq \mathcal{P}$ such that $u(\bar{p}) \geq u(p)$, for every $p \in \mathcal{P}^* \subseteq \mathcal{P}$. The path $\bar{p}$ is a shortest path from s to t and it is also a maximum capacity path among all the paths of $\mathcal{P}^*$.

Notice that as the maximum capacity path problem is finite this problem is also finite since there is a finite shortest path. In this case, there is also an optimal path which is loopless.

Theorem 6 *Let us assume that $\mathcal{P}_{si}$ and $\mathcal{P}_{it}$ are nonempty sets, for every $i \in \mathcal{N}$ and that the shortest path problem is finite. The problem of the maximum capacity path in the set of the shortest paths is finite.*

Proof: Let $\bar{p}$ be an optimal path for the problem of the maximum capacity path in the set of the shortest paths, that is, $\bar{p} \in \mathcal{P}^*$ and $u(\bar{p}) \geq u(p)$, for every $p \in \mathcal{P}^*$.

If $\bar{p}$ does not contain cycles, then it is a loopless path and this problem has a finite optimal solution.

Let us now assume that $\bar{p}$ has a cycle $\mathcal{C}$, i.e., $\bar{p} = p_1 \diamond \mathcal{C} \diamond p_2$. Once the shortest path problem is finite, $c(\mathcal{C}) \geq 0$, and then

$$c(\bar{p}) = c(p_1) + c(\mathcal{C}) + c(p_2) \geq c(p_1 \diamond p_2).$$

Since $\bar{p}$ is a shortest path $c(\bar{p}) = c(p_1 \diamond p_2)$, and then $p_1 \diamond p_2 \in \mathcal{P}^*$. Besides,

$$u(\bar{p}) = \min\{u(p_1), u(\mathcal{C}), u(p_2)\} \leq u(p_1 \diamond p_2).$$

Analogously, once $\bar{p}$ is a maximum capacity path in $\mathcal{P}^*$, then $u(\bar{p}) = u(p_1 \diamond p_2)$. Since $p_1 \diamond p_2$ has less cycles than $\bar{p}$ it can be concluded that it is possible to remove all the cycles in $\bar{p}$, obtaining then an optimal loopless path. Thus, the problem of the maximum capacity path in the set of the shortest paths is finite. □

Similarly to other problems, from this theorem the result presented in Corollary 6.1 follows.

Corollary 6.1 *Under the assumptions of Theorem 6, there is a loopless path that is an optimal solution for the problem of the maximum capacity path in the set of the shortest paths.*

It can now be proved that the problem of the maximum capacity path in the set of the shortest paths verifies the weak optimality principle.

Theorem 7 *Let $(\mathcal{N}, \mathcal{A})$ be a network without negative cycles. The problem of the maximum capacity path in the set of the shortest paths satisfies the weak optimality principle.*

Proof: Let us consider a network $(\mathcal{N}, \mathcal{A})$ without negative cycles. Then the shortest path problem in $(\mathcal{N}, \mathcal{A})$ is finite and satisfies the strong optimality principle. In what follows will be proved that there exists a maximum capacity path in $\mathcal{P}^*$ for which all loopless paths are also optimal.

Let $\bar{p}$ be a maximum capacity loopless path in the set of all the shortest paths, i.e., $u(\bar{p}) \geq u(p)$, for every $p \in \mathcal{P}^*$. Let us assume that $\bar{p} = p_1 \diamond q \diamond p_2$, where $q \in \mathcal{P}_{uv}$ is not an optimal path. Since the shortest path problem satisfies the strong optimality principle, q is a shortest path from u to v in $(\mathcal{N}, \mathcal{A})$, i.e., $q \in \mathcal{P}^*_{uv}$. Then, there is another shortest loopless path, $\bar{q} \in \mathcal{P}^*_{uv}$, which has maximum capacity in that set, i.e., $c(\bar{q}) = c(q)$ and $u(\bar{q}) \geq u(q')$, for every $q' \in \mathcal{P}^*_{uv}$. So, it can be considered a path from s to t in $(\mathcal{N}, \mathcal{A})$, $p_1 \diamond \bar{q} \diamond p_2$. This is a shortest path since

$$c(p_1 \diamond \bar{q} \diamond p_2) = c(p_1) + c(\bar{q}) + c(p_2) = c(p_1) + c(q) + c(p_2) = c(\bar{p});$$

$p_1 \diamond \bar{q} \diamond p_2$ is also a maximum capacity path in $\mathcal{P}^*$ since $\bar{p}$ is an optimal loopless path and

$$u(p_1 \diamond \bar{q} \diamond p_2) = \min\{u(p_1), u(\bar{q}), u(p_2)\} \geq \min\{u(p_1), u(q), u(p_2)\} = u(\bar{p})..$$

It is then possible to remove from $\bar{p}$ every subpath that is not optimal and once the number of loopless paths in $(\mathcal{N}, \mathcal{A})$ is finite, this process is also finite allowing us to obtain an optimal path containing only optimal subpaths. □

Algorithm 3: *Algorithm for the maximum capacity path problem in the set of the shortest paths*

For $(i \in \mathcal{N} - \{s\})$ Do $\pi_i^* \longleftarrow (+\infty, -\infty)$;
$p_s^* \longleftarrow \langle s \rangle$;
$\pi_s^* \longleftarrow (0, +\infty)$;
$X \longleftarrow \{s\}$;
While $(X \neq \emptyset)$ Do
  $i \longleftarrow$ node of X;
  $X \longleftarrow X - \{i\}$;
  For $((i,j) \in \mathcal{A})$ Do
    If $(\pi_j^1 > \pi_i^1 + c_{ij})$ Then
      If $(j \notin X)$ Then $X \longleftarrow X \cup \{j\}$;
      $p_j^* \longleftarrow p_i^* \diamond \langle i, (i,j), j \rangle$;
      $\pi_j^* \longleftarrow (\pi_i^1 + c_{ij}, \min\{\pi_i^2, u_{ij}\})$;
    Else
      If $(\pi_j^1 = \pi_i^1 + c_{ij}$ and $\pi_j^2 < \min\{\pi_i^2, u_{ij}\})$ Then
        If $(j \notin X)$ Then $X \longleftarrow X \cup \{j\}$;
        $p_j^* \longleftarrow p_i^* \diamond \langle i, (i,j), j \rangle$;
        $\pi_j^2 \longleftarrow \min\{\pi_i^2, u_{ij}\}$;
      EndIf
    EndIf
  EndFor
EndWhile

Once the problem of the maximum capacity path in the set of the shortest paths satisfies the weak optimality principle it can be solved by a labeling algorithm, after taking into account certain adaptations. These adaptations are presented in Algorithm 3. Note that whenever a tie for the cost of two paths from s to i happens, the path with better capacity is chosen.

Let us still consider the network $(\mathcal{N}, \mathcal{A})$, represented in Figure 3, and its tree of all the paths defined from 1 to 4, in Figure 4. After using Algorithm 3 it is obtained the tree represented in Figure 6.

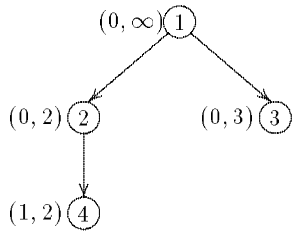

Figure 6: Tree obtained with a labeling algorithm for the maximum capacity path problem in the set of the shortest paths

5.2 The problem of the shortest path in the set of the maximum capacity paths

In this section $\bar{\mathcal{P}}$ will be used to denote the set of the maximum capacity paths from s to t (once again $\bar{\mathcal{P}} \subseteq \mathcal{P}$).

In the problem of the shortest path in the set of the maximum capacity paths, it is intended to determine a shortest path in the set $\bar{\mathcal{P}}$, i.e., a path $p^* \in \bar{\mathcal{P}} \subseteq \mathcal{P}$, such that $c(p^*) \leq c(p)$, for every path $p \in \bar{\mathcal{P}} \subseteq \mathcal{P}$. Then p^* is a maximum capacity path and it is a shortest path among all the paths of $\bar{\mathcal{P}}$. However, since the maximum capacity path does not satisfy the strong optimality principle (but only the weak optimality principle), it is not possible to assure that all the subpaths in p^* have maximum capacity. This is the reason why, in general, the problem of the shortest path in the set of the maximum capacity paths does not satisfy the weak optimality principle.

As an example the network represented in Figure 3 and the tree of all the paths from 1 to 4 in $(\mathcal{N}, \mathcal{A})$ in Figure 4 can be used again. Path $p^* = \langle 1, 2, 4\rangle$ is the unique optimal solution for this problem; however, its subpath $\langle 1, 2\rangle$, with cost 0 and capacity 2, is not an optimal path from 1 to 2 since $\langle 1, 3, 2\rangle$ has a higher capacity, 3.

Since this problem does not satisfy neither the strong nor the weak optimality principle, then it can not be solved using a labeling algorithm. This is shown in Figure 7, representing the tree constructed by a labeling algorithm, trying to determine the shortest path problem in the set of the maximum capacity paths from 1 to 4 in network $(\mathcal{N}, \mathcal{A})$.

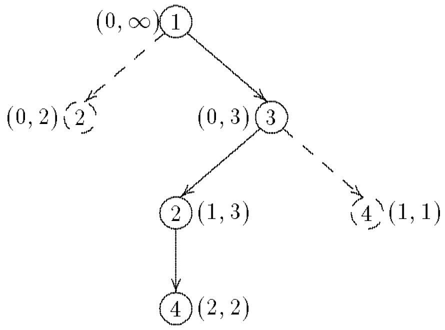

Figure 7: Tree obtained with a labeling algorithm, analogous to Algorithm 3 for the shortest path problem in the set of the maximum capacity paths

Although it can not be solved by a labeling algorithm, the problem of the shortest path in the set of the maximum capacity paths can be solved easily such as follows.

Let $\bar{u} = u(\bar{p})$, where $\bar{p} \in \mathcal{P}$ is a maximum capacity path in $(\mathcal{N}, \mathcal{A})$ from s to t.

It can be noted that the arcs in $(\mathcal{N}, \mathcal{A})$ whose capacity is lower than $\bar{u}$ will not be a part of any optimal solution for this problem. Therefore, after the value $\bar{u}$ is known, those arcs of $(\mathcal{N}, \mathcal{A})$ can be ignored. This is stated in Theorem 8.

Theorem 8 *Let $\bar{u} = u(\bar{p})$, where $\bar{p} \in \bar{\mathcal{P}}$, and a let p be other maximum capacity path in $(\mathcal{N}, \mathcal{A})$ from s to t. Then $u_{ij} \geq \bar{u}$, for every arc $(i, j) \in p$.*

Proof: Let $p \in \bar{\mathcal{P}}$ and let us assume that $p = p_1 \diamond \langle i, (i, j), j\rangle \diamond p_2$, where $u_{ij} < \bar{u}$. Then

$$u(p) = \min\{u(p_1), u_{ij}, u(p_2)\} < \bar{u},$$

and p would not be a maximum capacity path in $(\mathcal{N}, \mathcal{A})$ as it was assumed. $\square$

Based on this theorem it can be concluded that the shortest path in $\bar{\mathcal{P}}$ does not contain any arc with a capacity lower than $\bar{u}$. Moreover, let $\bar{\mathcal{A}}$ be the set obtained from $\mathcal{A}$ removing these arcs, and let $\bar{\mathcal{P}}(\bar{\mathcal{A}})$ denote the set of the maximum capacity paths from s to t in $(\mathcal{N}, \bar{\mathcal{A}})$. The following result, which asserts that $\bar{\mathcal{P}} = \bar{\mathcal{P}}(\bar{\mathcal{A}})$, is immediate and it will be presented without proof.

Theorem 9 *The set of the maximum capacity paths from s to t in $(\mathcal{N}, \mathcal{A})$ coincides with the set of the maximum capacity paths from s to t in $(\mathcal{N}, \bar{\mathcal{A}})$, where $\bar{\mathcal{A}} = \{(i, j) \in \mathcal{A} : u_{ij} \geq \bar{u}\}$.*

Changing the initial network, $(\mathcal{N}, \mathcal{A})$, according to Theorem 9 it is then possible to determine the shortest path in $\bar{\mathcal{P}}$, which is simply the shortest path from s to t in $(\mathcal{N}, \bar{\mathcal{A}})$. Therefore, this problem can be solved in three phases:

1. Determining $\bar{u}$, the maximum capacity of every path from s to t in $(\mathcal{N}, \mathcal{A})$.

2. Removing from $(\mathcal{N}, \mathcal{A})$ the arcs with capacity lower than $\bar{u}$.

3. Determining the shortest path from s to t in the resulting network.

These three phases are resumed and presented in Algorithm 4.

Algorithm 4: *Algorithm for the shortest path problem in the set of the maximum capacity paths*

For $(i \in \mathcal{N} - \{s\})$ Do $\pi_i \longleftarrow -\infty$;
$p_s^* \longleftarrow \langle s \rangle$;
$\pi_s \longleftarrow +\infty$;
$X \longleftarrow \{s\}$;
While $(X \neq \emptyset)$ Do
  $i \longleftarrow$ node of X;
  $X \longleftarrow X - \{i\}$;
  For $((i,j) \in \mathcal{A}$ such that $\pi_j < u(p_i^* \diamond \langle i, (i,j), j \rangle))$ Do
    $p_j^* \longleftarrow p_i^* \diamond \langle i, (i,j), j \rangle$;
    $\pi_j \longleftarrow u(p_i^* \diamond \langle i, (i,j), j \rangle) = \min\{\pi_i, u_{ij}\}$;
    If $(j \notin X)$ Then $X \longleftarrow X \cup \{j\}$;
  EndFor
EndWhile
$u^* \longleftarrow \pi_t$;
For $(i \in \mathcal{N} - \{s\})$ Do $\pi_i \longleftarrow +\infty$;
$p_s^* \longleftarrow \langle s \rangle$;
$\pi_s \longleftarrow 0$;
$X \longleftarrow \{s\}$;
While $(X \neq \emptyset)$ Do
  $i \longleftarrow$ node of X;
  $X \longleftarrow X - \{i\}$;
  For $((i,j) \in \mathcal{A}$ such that $u_{ij} \geq u^*$ and $\pi_j < c(p_i^* \diamond \langle i, (i,j), j \rangle))$ Do
    $p_j^* \longleftarrow p_i^* \diamond \langle i, (i,j), j \rangle$;
    $\pi_j \longleftarrow c(p_i^* \diamond \langle i, (i,j), j \rangle) = \pi_i + c_{ij}$;
    If $(j \notin X)$ Then $X \longleftarrow X \cup \{j\}$;
  EndFor
EndWhile

Acknowledgments:
Research developed within **CISUC** ("Centro de Informática e Sistemas da Universidade de Coimbra") and partly supported by the **PRAXIS XXI** Project of **JNICT** ("Junta Nacional de Investigação Científica e Tecnológica").

References

[1] Y.P. Aneja and K.P.K. Nair. The constrained shortest path problem. *Naval Research Logistics Quarterly*, 25:549–555, 1978.

[2] R.E. Bellman. On a routing problem. *Quarterly Applied Mathematics*, 1:425–447, 1958.

[3] B.V. Cherkassky, A.V. Goldberg, and T. Radzik. Shorthest paths algorithms: Theory and experimental evaluation. *Mathematical Programming*, 73:129–196, 1996.

[4] E.V. Denardo and B.L. Fox. Shortest route methods: reaching, pruning and buckects. *Operations Research*, 27:161–186, 1979.

[5] N. Deo and C. Pang. Shortest paths algorithms: taxonomy and annotation. *Networks*, 9:275–323, 1979.

[6] R. Dial, G. Glover, D. Karney, and D. Klingman. A computational analysis of alternative algorithms and labelling techniques for finding shortest path trees. *Networks*, 9:215–348, 1979.

[7] E. Dijkstra. A note on two problems in connection with graphs. *Numerical Mathematics*, 1:395–412, 1959.

[8] S.E. Dreyfus. An appraisal of some shortest-path algorithms. *Operations Research*, 17:395–412, 1969.

[9] G. Gallo and S. Pallotino. Shortest path methods: a unifying approach. *Mathematical Programming Study*, 26:38–64, 1986.

[10] P. Hansen. Bicriterion path problems. in *Multiple Criteria Decision Making: Theory and Applications*, editors: G. Fandel and T. Gal, Lectures Notes in Economics and Mathematical Systems, 177, 109-127, Springer Heidelberg, 1980.

[11] J. K. Lenstra, E. L. Lawler, and A. H. G. Rinnooy Kan (eds.). *The Traveling Salesman Problem.* John Wiley & Sons, Inc., 1983.

[12] E.Q.V. Martins. Determinação de caminhos óptimos em redes orientadas, 1984. (Dissertação de Doutoramento; Departamento de Matemática; Universidade de Coimbra).

[13] E.Q.V. Martins, M.M.B. Pascoal, and J.L.E. Santos. The k shortest paths problem. Submitted, 1998.

[14] V. V. Masch. The cyclic method of solving the transshipment with an additional constraint. *Networks*, 10:17–31, 1980.

[15] E.F. Moore. The shortest path through a maze. Proceedings of the International Symposium on the Theory of Switching, (1959), Harvard University Press, 285–292.

[16] U. Pape. Implementation and efficiency of moore algorihms for the shortest root problem. *Mathematical Programming*, 7:212–222, 1974.

[17] M.M.B. Pascoal. Algoritmos para a enumeração dos k trajectos mais curtos, 1997. (Dissertação de Mestrado; Departamento de Matemática; Universidade de Coimbra).

[18] J.L.E. Santos. O problema do trajecto óptimo multiobjectivo, 1997. (Dissertação de Mestrado; Departamento de Matemática; Universidade de Coimbra).

[19] J.Y. Yen. Shortest path network problems. (Mathematical Systems in Economics, Heft 18), Hain, (1975), Meisennheim am Glan.