

New Algorithm Permitting the Construction of an Effective Spanning Tree

Salima Nait Belkacem*, Moncef Abbas

Abstract. In this paper, we have done a rapid and very simple algorithm that resolves the multiple objective combinatorial optimization problem. This, by determining a basic optimal solution, which is a strong spanning tree constructed, according to a well-chosen criterion. Consequently, our algorithm uses notions of Bellman's algorithm to determine the best path of the network, and Ford Fulkerson's algorithm to maximise the flow value. The Simplex Network Method that permits to reach the optimality conditions manipulates the two algorithms. In short, the interest of our work is the optimization of many criteria taking into account the strong spanning tree, which represents the central angular stone of the network. To illustrate that, we propose to optimize a bi-objective distribution problem.

Keywords: Bi-objective minimum cost flow problem, Network flows, Linear programming, Minimum spanning tree, Efficient extreme point.

1. Introduction

Combinatorial optimization plays a central role within many applications in the real case: seeing that it is a moving, an exciting and a dynamic domain. Its presence in the different problems has led to the creation of a variety of technique and methods of the multi-objective minimum cost flow problems' resolution.

In particular, our method presented in this paper, consist in finding the optimal spanning tree of a flow network problem following a well-chosen criterion. This one will be the central angular stone of the network, and thank to it we will be able to optimize the other criteria to obtain the different results to improve such as the time (duration), the cost and the distance.

* (Corresponding author) Département de Recherche Opérationnelle, Faculté de Mathématiques, Université des sciences et de la Technologie Houari Boumediene, Bab-Ezzouar (Algérie). Works at Faculté des Hydrocarbures et de la Chimie, Université M'Hamed Bougara, Boumerdes (Algérie), email : naitsali@yahoo.fr

Moncef Abbas, USTHB, Faculté des Mathématiques, / Laboratoire « AMCD&RO »,

As these minimum cost flow problems exist in almost all industries including agriculture, energy and transportations, the best way to understand the basic idea of our method is via an example.

Supposing that a company has P factories with known demands, it will identify flows that are going to satisfy the warehouses demands, and this from available supplies in the factories all by minimizing the dispatch cost e.g. [1].

The problem of this company represents clearly a particular case belonging to the minimum cost flow problems also said: « Transport problem ».

As the central concept subjacent to the network is the spanning tree notion, and as this one plays an important role in the domain of flow network problems, we have to define it: Each spanning tree of the graphic $G = (V, A)$ has $n-1$ arcs, $n = |V|$ nodes, $m = |A|$ arcs and a length or a cost arc associated to each arc $(i, j) \in A$; when the spanning tree has the smallest cost, it is called « minimum spanning tree » e.g. [1].

We remark that there exist many models (shapes) of spanning trees. For example, the spanning tree used in the simplex method is a « spanning tree solution » (it should be noted that the minimum cost flow problems have always spanning tree solutions). On the other hand, when we use the Simplex Network Method that is an adaptation of the Simplex Method, known in linear programming, we use what we call a « spanning tree structure » e.g. [1-2, 5].

The selection of the optimal solution has to take into account several objectives; these aimed objectives could be for example the minimisation of the cost and the duration or the maximization of the security, the debit, etc. The study of these multi-objective minimum cost flow problems becomes then more interesting.

In our work, the considered goal is to find the optimal spanning tree similar to the strong spanning tree structure existing in the Simplex Network Method. (The spanning tree structure satisfies the optimality conditions) e.g. [1].

To reach our objective, we use Bellman and Ford Fulkerson's algorithms, manipulated by notions of the Simplex Network Method. Bellman's algorithm has for goal to determine the best path between all the different points of the Network, starting by a node source S to a node receiver P . Ford Fulkerson's algorithm is going to improve this road, and thanks to the Simplex Network Method Notions, we will be able to reach the optimality conditions, then make the spanning tree stronger. Otherwise, the arcs changing that leads to a manipulation in an iterative way, permits to variate the cycles at each step to reach the optimal spanning tree (already constructed from the combination of the classical algorithms), from here, we can optimize other criteria taking into account this one [8, 12].

The article is constructed as follows:

In « Section 2 », we give some definitions and formulations of the mono and bi objective flow network problem. After that, we propose in « Section 3 » an algorithm that allows us to find the efficient spanning tree, illustrated by an example. Moreover, in « section 4 » is our conclusion.

First, we give some necessary definitions to follow up.

2. Definitions

2.1. Definition

A graph $G' = (V', A')$ is a subgraph of $G = (V, A)$: If $V' \subseteq V$ and $A' \subseteq A$ we say that $G' = (V', A')$ is the subgraph of G , induced by V' , if A contains each arc of A' with both end points in V' .

A graph $G' = (V', A')$ is a spanning subgraph of $G = (V, A)$ if $V' = V$ and $A' \subseteq A$ [1].

2.2. Definition

A tree is a connected graph that contains no cycles [1].

Example 1; Figure 2 shows a tree of the graph shown in Figure 1.

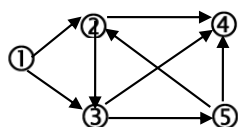


Figure 1. Network example

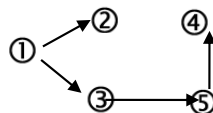


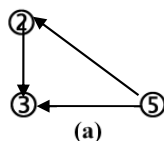
Figure 2. Tree example

Every spanning tree connected to n nodes graph G has $(n-1)$ arcs.

2.3. Definition

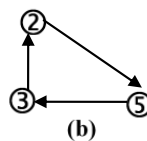
A directed cycle is a directed $i_1-i_2-\dots-i_n$ together with the arc (i_n, i_1) [1].

Example 2; the graph shown in Figure 3 (a) is a cycle, but not a directed one. The graph shown in Figure 3 (b) is a directed cycle.



(a)

Figure 3. Examples of cycles



(b)

2.4. Formulation of the bi objective network flow problem

Let us first introduce the following notations:

Let $G = (V, A)$ be an oriented network, $V = \{1, 2, 3, \dots, n\}$ a set of nodes and A a set of arcs. For each node $i \in V$, let the integer b_i represent the supply/ demand of the node i . For each arc $(i, j) \in A$, u_{ij} and l_{ij} respectively denote the upper and lower bounds of the flow passing through the arc $(i, j) \in A$. c'_{ij} is the cost of one unit of the flow on the arc (i, j) at the objective function t^{th} ($t = 1, 2$). x_{ij} is the flow through the arc (i, j) ; $\text{Suc}(i) = \{j \in V / (i, j) \in A\}$, $\text{Pred}(i) = \{j \in V / (j, i) \in A\}$ e.g. [8-9].

The Bi objective Network Flow (BNF) problem can be formulated in the following way:

$$(PL_1) \quad \left\{ \begin{array}{ll} \text{Min } f_1(x) = \sum_{i \in V} \sum_{j \in \text{Suc}(i)} c_{ij}^1 x_{ij} & (1) \\ \text{Min } f_2(x) = \sum_{i \in V} \sum_{j \in \text{Suc}(i)} c_{ij}^2 x_{ij} & (2) \\ \text{St } \sum_{j \in \text{Suc}(i)} x_{ij} - \sum_{j \in \text{Pred}(i)} x_{ji} = b_i \quad \forall i \in V & (1') \\ l_{ij} \leq x_{ij} \leq u_{ij} & \forall (i, j) \in A \quad (2') \end{array} \right.$$

2.5. Formulation of the mono objective flow problem

The mono objective minimum cost flow problem can be formulated as a linear mathematical program as follows:

$$(PL_2) \quad \left\{ \begin{array}{ll} \text{Min } f(x) = \sum_{i \in V} \sum_{j \in \text{Suc}(i)} c_{ij} x_{ij} & (1) \\ \sum_{j \in \text{Suc}(i)} x_{ij} - \sum_{j \in \text{Pred}(i)} x_{ji} = b_i \quad \forall i \in V & (1') \\ l_{ij} \leq x_{ij} \leq u_{ij} & \forall (i, j) \in A \quad (2') \end{array} \right.$$

In general, there is no feasible solution for the BNF problems that simultaneously minimizes both of the objectives. For this reason, the solutions of these problems are searched among the set of efficient points, satisfying the following definitions:

*The set of feasible solution or the decision space is denoted by X and its image through $\{(f_1(x), f_2(x)) / x \in X\}$ is called objective space.

2.6. Definition

Let $K \subset \mathbb{R}^2$ be the feasible region in the criterion space. Then, $z \in K$ if and only if, there exist $x \in X$ such $z = (f_1(x), f_2(x))$.

2.7. Definition

A point (or a solution) $x \in Z$ is efficient if and only if, its criterion vector

$z = (f_1(x), f_2(x))$ is non-dominated.

Augmenting path

Supposing that we have established a flow in a network. We consider a path in this network from a source node s to a well node p . If we want to increase the flow value in the network, we have to modify x_{ij} of (i, j) arcs of the considered path, without affecting the arcs of the flow that are not in this path.

If the arc (i, j) goes in the same direction of the path, then we add the value δ to the flow x_{ij} , and if the arc goes in the opposite direction, in this case, the value δ should be subtracted from the flow x_{ij}

$$\text{Where } \delta = \min \left[\min_{(i,j) \in C^+} [u_{ij} - x_{ij}]; \min_{(i,j) \in C^-} x_{ij} \right]$$

C^+ : the set of arcs in the same direction of the path;

C^- : the set of arcs in the inverse direction of the path;

Thus, we determined an increasing path, whose flow value δ is increasing.

We did increase the flow x_{ij} to δ for (i, j) arcs of C^+ and decreased x_{ij} to δ for (i, j) arcs of C^- e.g. [3] and algorithm in Annex.

The function of Bellman and Ford

The Bellman algorithm uses the principle dynamic programming.

We give an algorithm, which finds the distances between the source node S and a maximum of other nodes i see in annex.

For each node u of a graph

$d[u] = +\infty$

$d[s] = 0$

For $k = 1$ to number of nodes **do**

 For each arc (u, p) of graph **do**

$d[p] = \text{minimum} (d[p], \{d[u] + \text{values}(u, p)\})$

Return to $d[u]$

Example 3

We consider a network with 6 nodes, 10 arcs and many criteria, each criterion represents a distance. We apply Bellman's algorithm for each distance. We obtain then three different paths from a source node $S=1$ to a shaft node $P=6$ e.g. Figure 4. We choose the spanning tree corresponding to the third distance. Next, we apply Ford Fulkerson's algorithm for the chosen spanning tree in order to improve it. [For more details and a better comprehension, see Example 4 and Table 1].

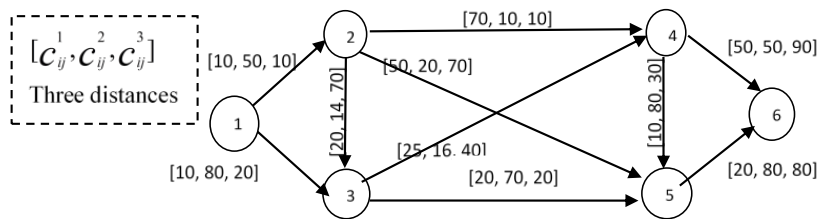
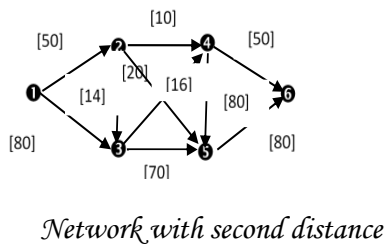
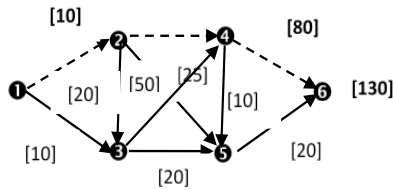
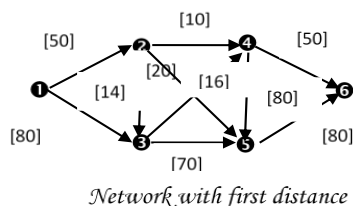
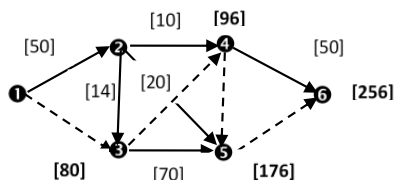


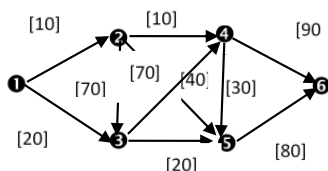
Figure 4. Network example

Work:

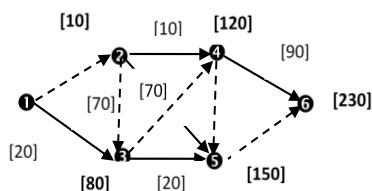




(b) 1-3-4-5-6. Path corresponding to the second distance



Network with third distance



(c) 1-2-3-4-5-6. Path corresponding to the third distance

Figure 5. Different paths determined by the algorithm of Bellman algorithm

In Figure 4 (c) we have the results maximum distance of the path 1-2-3-4-5-6 / $d(1) = 0$; $d(2) = 10$; $d(3) = 80$; $d(4) = 120$; $d(5) = 150$; $d(6) = 230$.

3. Introduction of the algorithm constructing the spanning tree of the network

In the first step, once the initial spanning tree corresponding to only one criterion is obtained by the algorithm of Bellman. We use its arcs as the set B . The other arcs that do not belong to the spanning tree B are noted as set H . We choose L as the set of arcs that have its flows at

the inferior borne, and H , the set of arcs that have its flows at the superior borne. The flow of the spanning tree B will be improved by Ford Fulkerson's algorithm.

Optimizing this criterion, the extreme point $f(x)$ is obtained using Bellman's algorithm. This calculation of the potential nodes and the reduced costs, $\bar{c}_{ij}^1 = c_{ij}^1 - \pi_i^1 + \pi_j^1$ for all $(i, j) \in A$, allows us to optimize the spanning tree B taking into account the considered criteria and the arcs changing between the sets B , L and H until reaching the following optimality conditions e.g. Ahuja [1].

$$\text{Optimality conditions: } \left\{ \begin{array}{ll} l_{ij} \leq x_{ij} \leq u_{ij} & \text{and} \quad \bar{c}_{ij} = 0 \quad \forall (i, j) \in B \\ x_{ij} = l_{ij} & \text{and} \quad \bar{c}_{ij} \geq 0 \quad \forall (i, j) \in L \\ x_{ij} = u_{ij} & \text{and} \quad \bar{c}_{ij} \leq 0 \quad \forall (i, j) \in U \end{array} \right.$$

At the end, we stop the algorithm when all the arcs of the network satisfy all the optimality conditions.

In other words, all the arcs that haven't satisfied the optimality conditions are the arcs belonging to the set $H = \{(i, j) \in L, \bar{c}_{ij} < 0\} \cup \{(i, j) \in U, \bar{c}_{ij} > 0\}$. The selected arcs each time are those that reach the minimal value

$$\alpha = \min \{ \bar{c}_{ij} / \{(i, j) \in L, \bar{c}_{ij} < 0\} \cup \{(i, j) \in U, \bar{c}_{ij} > 0\} \}.$$

The arcs changing that is done each time between the sets B , L and H permits to have the optimum by the end.

Which means, once arrived to the end, we will not have any arc that does not satisfy the optimality conditions.

In the second step, we start from the same hypothesis given in the first step and we optimize the second criterion taking into consideration the first one.

To realize that, once the initial spanning tree is selected by Bellman's algorithm, we choose it as a first criterion. The other criteria take other positions; they will be optimized according to the spanning tree of the network.

Thus, we optimize the last criterion (determined by the algorithm of Bellman) by Sedenonoda and Gonzalez-Martin's algorithm [8-9, 11-12] using the formula:

$$\alpha' = \min \left\{ \frac{\bar{c}_{ij}^2}{\bar{c}_{ij}^1} : \bar{c}_{ij}^2 < 0; \forall (i, j) \in L; \frac{\bar{c}_{ij}^2}{\bar{c}_{ij}^1} : \bar{c}_{ij}^2 > 0; \forall (i, j) \in U \right\}$$

Then, the set H contains all the variables out of the base that reach the value α' e.g. [2, 6, 12].

The process stops when all the arcs of this network verify the optimality conditions, this permits us to have all the effective extreme solutions of the problem in the objective space, in a simple and a rapid way. See table 2 and annex.

3.1. Theorem

If the objective function of a minimum cost flow problem is limited inferiorly in the feasible area, the problem has always an optimal spanning tree solution. [1]

3.2. Resolution algorithm

We propose an algorithm that uses Bellman's algorithm and allows us to find the following spanning tree corresponding to one criterion.

Then, this spanning tree is improved using the algorithm of Ford Fulkerson. The aimed goal is to use that spanning tree to resolve the multi-objective minimum cost flow problems, which means, we optimize the second criterion taking into account the spanning tree, this one corresponds to the first criterion already determined by Bellman's algorithm, see annex [1, 12] and example 4.

- x^0 represent an initial solution-spanning tree found by Bellman algorithm.
- π^1, π^2 denote the potential node to compute for each criterion.
- Depth (i) is the path length of the spanning tree connecting a source node s to a sink node P.
- Thread (i) is a path of the set of all descendants of node i.
- B is the set arcs of spanning tree generated by Bellman algorithm.
- L is the set arcs not belong to spanning tree B with a flow restricted in value on lower bounds.
- U is the set arcs not belong to spanning tree B with a flow restricted in value on upper bounds.
- (B^0, L^0, U^0) the initial spanning tree and two sets associated with the starting efficient solution.

begin

B is a spanning tree corresponding to one criterion, which is found using the Bellman algorithm;

Let L the set of the arcs have their flow on lower bound and U the set of arcs have their flow on upper bound;

Let $\{x^0, \pi^1, B, L, U\}$ the first efficient solution ;

Let Pred, Depth and Thread three indices corresponding to the spanning tree B ;

$\bar{c}_{ij} = c_{ij}^1 - \pi_i^1 + \pi_j^1 \quad \forall (i, j) \in A$, reduced cost corresponding to the first criterion;

Put $x = x^0$ an efficient extreme point corresponding to the spanning tree found by Bellman algorithm;

As long as B is not optimal spanning tree **do**

```

begin
  Let  $\{ \mathcal{X}, B, L, U, \pi^1 \}$  the first efficient point;
  Compute _ Entering_ Arcs  $(L, U, \pi^1, H, \bar{c}_{ij}^{-1}, \mathcal{X})$ ;
  Compute _ New_ Points  $(\mathcal{X}, B, L, U, \pi^1, Pred, Depth, H)$ 
end
end.

```

Procedure 1 Compute _ Entering_ Arc $(L, U, \pi^1, H, \bar{c}_{ij}^{-1}, \mathcal{X})$;

```

begin
- Input: Reduced costs  $\bar{c}_{ij}^{-1} = c_{ij}^1 - \pi_i^1 + \pi_j^1 \quad \forall (i, j) \in A$  derived from current efficient
  solution  $\mathcal{X}$  /
   $\mathcal{X}$  efficient point obtained at several iteration.
  Let  $\alpha = \min \left\{ \bar{c}_{ij}^{-1} / \{ \{(i, j) \in L, \bar{c}_{ij}^{-1} < 0 \} \cup \{(i, j) \in U, \bar{c}_{ij}^{-1} > 0 \} \} \right\}$ 
  Let  $H$  a set of arcs where the last minimum is reached
- Output: News reduced costs
end;

```

Procedure 1 Compute _ New_ Point $(\mathcal{X}, B, L, U, \pi^1, Pred, Depth, Thread, H)$;

```

begin
While  $H \neq \emptyset$  do
  begin
    Let  $(i, j)$  the first arc of  $H$ ;  $H := H - (i, j)$ ;
    If  $(i, j)$  did not verify the optimality conditions of first objective then
      Let  $\delta_{ij}$  the value of flow in arc  $(i, j)$  allows modified the flow  $\mathcal{X}$  all along the spanning
      tree  $B$  and
      let  $(p, q)$  arc leaving out ;
      If  $\delta_{ij} > 0$ , then
        begin
          Send the value of flow throwing the pivot cycle define by arc  $(i, j)$ ;
          Let  $N\mathcal{X}$  the new flow;
          Update  $\mathcal{X}, B, L, U, \pi^1, Pred, Depth, Thread$ ;
          Store  $NB, NL, NU, N\pi^1$ , and  $N\mathcal{X}$  as efficient extreme point
        end
      end
  end

```

end ;

Example 4

A distribution company proposes merchandise delivery between towns noted by 1, 2, 3, 4, 5 and 6, in which the debit = 8 ; where $b_1 = (\text{source}) = 8$ and $(\text{well- shaft}) = -8$, and $b_i = 0$ ($i = 2, 3, 4, 5$). Each arc is equipped with a lower bound, an upper bound and four criteria.

To supply and satisfy all its customers, it constructs a network where the towns represent nodes and the connexions between them will be presented by arêtes see Figure 5.

To realise the objective of this company, we are going to use Bellman's algorithm to determine the path that touches the maximum of the nodes (towns), as well as, to represent the maximal duration of this distribution project. Evidently, by applying this algorithm on all the criteria, we will be able to choose the best of them which will construct the spanning tree of the network (we note it then 1st criterion).

Then, we calculate the potential nodes and the reduced costs of the network. See Figure 7 ; where $B = \{(1,2) ; (2,3) ; (3,4) ; (4,5) ; (5,6)\}$, $L = \{(1,3) ; (2,4) ; (2,5) ; (3,5) ; (4,6)\}$ the arcs which have an inferior limit and $U = \{(1,3) ; (2,4) ; (2,5) ; (3,5) ; (4,6)\}$ the arcs which have a superior limit.

The following operation is the application of Ford Fulkerson's algorithm along the spanning tree by the modification of the cycles each time. The exchange movement of arcs candidate to go in or to go out between the sets B , L and U , pursues in many interaction and ends when we reach the optimality conditions (the optimum).

The last obtained spanning tree is constructed according to the 1st criterion, it will represent the optimal strong spanning tree of this Network and its « stop test » is determined by our algorithm, considering it, we will be able to optimize other criteria.

It means that: when reaching the optimum, all the arcs satisfy the optimality conditions (So, there is no arc in the set H). The arcs belonging to set U have a negative reduced cost.

Finally, in the case of this company, we have proceeded according to two stapes. The first one consist in building a spanning tree. The second one, optimizes many criteria taking into account the spanning tree constructed by the two algorithms and the Simplex Network Method [8-9, 11-12].

All the efficient extreme points in the objective space can be seen in table 2.

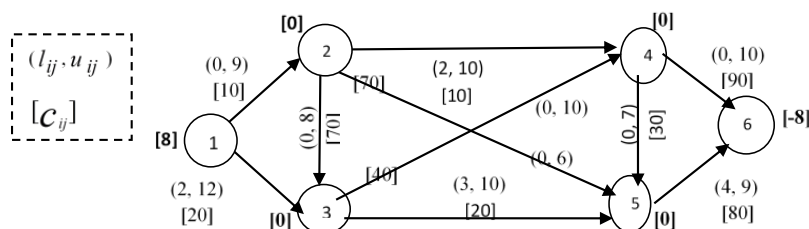


Figure 6. Example of a network with one criterion

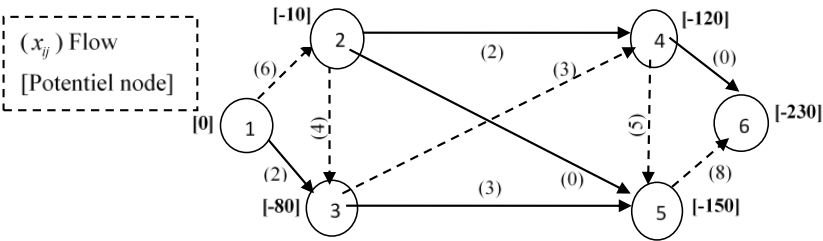


Figure 7. Spanning tree determined by Bellman’s algorithm

Table 1. All the efficient extreme points in the objective space for a mono objective problem.

	(1,2)	(1,3)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)	(4,6)	(5,6)	f ₁
x^1	⑥	2	④	2	0	③	③	⑤	0	⑧	1370
x^2	⑥	2	①	⑤	0	0	③	⑤	0	⑧	1070
x^3	⑥	2	①	⑤	0	0	③	①	④	4	990
x^4	⑤	③	0	⑤	0	0	③	①	④	4	930
x^5	④	④	0	④	0	0	④	0	④	4	920

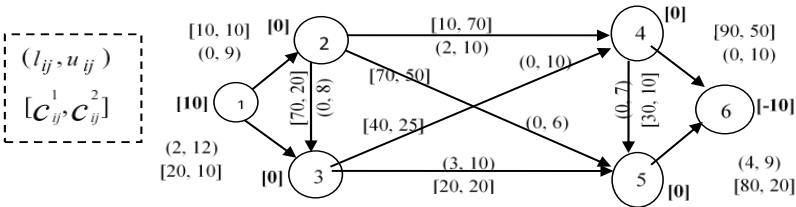


Figure 8. Example of a network with two criteria

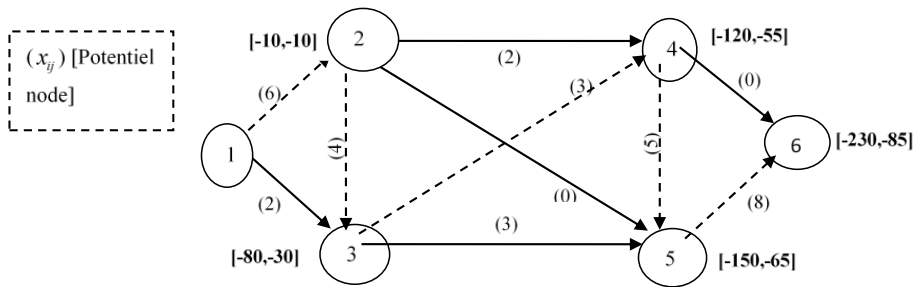


Figure 9. Optimization of the second criterion taking into account the initial spanning tree.

Table 2. All the efficient extreme points in the objective space for a bi objective problem.

	(1,2)	(1,3)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)	(4,6)	(5,6)	f_1	f_2
x^1	⑥	2	④	2	0	③	3	⑤	0	⑧	1370	645
x^2	⑥	2	①	2	③	0	③	②	0	⑧	1160	630
x^3	⑤	③	0	2	③	0	3	②	0	⑧	1100	610
x^4	②	⑥	0	2	0	③	3	⑤	0	⑧	110	565
x^5	②	⑥	0	2	0	0	⑥	②	0	⑧	980	520

4. Conclusion

In this paper, we presented a method that allows us to choose among several criteria, the criterion that is suitable for building an optimal spanning tree of the network. Choosing this criterion, is determined by our method that is characterized by the algorithm of Bellman and Ford-Fulkerson, the Network simplex method, in addition to simple traditional tools well known in graph theory. Subsequently, taking into account this spanning tree, we determined all the efficient extreme points in the objective space of the bi objective minimum cost flow network problem. In conclusion, we hope to see our work programmed by many softwares and applied on different problems in the reel case. e.g. [1, 3, 5- 6, 9-10].

References

- [1] Ahuja R K., Magnanti T L., Orlin J B., Network flows: Theory, algorithm, and applications. Prentice-Englewood Cliffs, 1993. .
- [2] Cunningham W H., A network simplex method, Mathematical Programming, **11**, 1976, 105-116.
- [3] Eusébio A., Figueira J R., Ehrgott M., On finding representative non-dominated points for the bi objective integer network flow problems, Computers & Operations Research, **48**, 2014, 1-10.
- [4] Geoffrion A.M., "Proper efficiency and the theory of vector maximization", Journal of Mathematical Analysis and Application, **22**, 1968, 618-630.
- [5] Hamacher H W., Pedersen C R., Ruzika S., Multiple objective minimum cost flow problems: a review, European Journal of Operational Research, **176**, 2007, 1404-1422.
- [6] Lee H., Pulat S., Bi criteria network flow problems: continuous case, European Journal of Operational Research, **51**, 1991, 119-126
- [7] Lee H., Pulat S., Bicriteria network flow problems; Integer case, European Journal of Operational Research, **66**, 1993, 1488-157.
- [8] Nait Belkacem S., An Algorithm for Choosing Ordering a New Criteria of a Bi Objective Flow Problem, Foundations of Computing and Decision Sciences, **46**, 2021, 11-26. DOI: 10.2478/fcds- 2021-0002.
- [9] Nait Belkacem S., Contribution à la Résolution du Problème de Flot Multi Objectif URI, Mémoire de Magister, Mathématique, Université des Sciences et Technologie Houari Boumedienne Algérie, 2007, 81. <http://hdl.net/123456789/3195>.
- [10] Raith A., Ehrgott M., A two-phase algorithm for the bi objective integer minimum cost flow problem, Computers and Operations Research, **36**, 2009, 1945-1954.
- [11] Sedenó-Noda A., Gonzalez-Martin C., An algorithm for the bi-objective integer minimum cost flow problem, Computers and Operations Research, **28**, 2001, 139-156.
- [12] Sedenó-Noda A., Gonzalez-Martin C., The bi-objective minimum cost flow problem, European Journal of Operational Research, **124**, 2000, 591-600

Received 12.06.2022, Accepted 10.03.2023

ANNEX

Bellman algorithm e.g. [1]

Let $G = (V, A)$ directed graph, the path of node u to node v define by the nodes sequence $\langle\langle s_0, s_1, s_2, \dots, s_k \rangle\rangle$ which $u = s_0$, $v = s_k$ and $(s_{i-1}, s_i) \in A$ for all $i \in \{1, 2, 3, \dots, n\}$; the

distance of s to p is $d[p, |V| - 1]$ with: $d[p, k]$ is a distance of the source node s to sink node p . The path has k arcs.

Function Bellman Ford

For each node u of a graph
 $d[u, 0] = +\infty$
 $d[s, 0] = 0$
 For $k = 1$ to nodes number -1 do
 For node p do
 $d' = d[p, k-1]$
 For each arc (u, p) of a graph do
 $d' = \max(d', d(u, k-1) + \text{value}(u, p))$
 $d[p, k] = d'$
 Return $d[\cdot, |V| - 1]$

Procedure Simplex_Pivot (i, j) ; [1] and [10]

begin:

Input: select the arc (i, j) candidate out by choosing the value α ;
 Let w denote the basic cycle containing (i, j) ;

Let $\delta_{ij} = \min \{ r_{ij} / (i, j) \in w \}$;

Send δ_{ij} value of flow around w ;

Let (p, q) denote the arc of w that is pivoted out;

Output: news basic.

end

The simplex pivot is introducing arc (i, j) into the basis and removing the leaving arc (p, q) from the basis. The reduced costs may change while there are arcs remaining in H with have values of α the process repeats.

Augmenting path algorithm [1]

begin

$x := 0$

While $G(x)$ contains a directed path from node s to node p do

begin

Identify an augmenting path P from node s to node p ;

$\delta = \min \{ r_{ij} : (i, j) \in P \}$

Augment δ unit of flow along P and update $G(x)$;

end;

end;

Network simplex algorithm [1]

The network simplex algorithm maintains a feasible spanning tree at the each iteration and successively transforms it into an improved spanning tree structure until it becomes optimal.

Begin

Determine an initial feasible spanning tree (B, L, U) ;

Let $\mathcal{X}$ be the flow and π be the node potentials associated with this spanning tree;

While some non-tree arc violates the optimality, conditions **do**

begin

Select an entering arc (k, l) violating its optimality condition;

Add arc (k, l) to the spanning tree and determine the leaving arc (p, q) ;

Perform a spanning tree update and update the solutions $\mathcal{X}$ and π ;

end;

end;

Resolution algorithm for the bi objective flow problem**begin**

B is a spanning tree corresponding to one criterion, which is found using the Bellman algorithm;

Let L the set of the arcs have their flow on lower bound and U the set of arcs have their flow on upper bound [12];

Let $\{x^0, \pi^1, \pi^2, B, L, U\}$ the first efficient solution ;

Let Pred, Depth and Thread three indices corresponding to the spanning tree B ;

$c_{ij} = c_{ij}^1 - \pi_i^1 + \pi_j^1 \quad \forall (i, j) \in A, \quad c_{ij} = c_{ij}^2 - \pi_i^2 + \pi_j^2 \quad \forall (i, j) \in B$ reduced cost ;

Put $\mathcal{X} = x^0$ an efficient extreme point corresponding to the spanning tree found by Bellman algorithm;

As long as B is not optimal spanning tree **do**

begin

Let $\{\mathcal{X}, B, L, U, \pi^1, \pi^2\}$ the first efficient point;

Compute _entering_ arcs (L, U, π^1, π^2, H') ;

Compute_new_points ($\mathcal{X}$, B , L , U , π^1 , π^2 , Pred, Depth, H')

end

end.