Radboud University Nijmegen

Faculty of Science

Optimization and Allocation in Minimum Cost Spanning Tree Problems with Time

Master Thesis Mathematics

Supervisors Tilburg University:
Prof. dr. P.E.M. Borm
J. Schouten MSc

Author:
D. de Moor

Supervisor Radboud University:
dr. W. Bosma

April 17, 2019

Acknowledgements

First of all, I would like to thank my supervisors Peter Borm and Jop Schouten from Tilburg University for the opportunity to write my thesis under their guidance. I am grateful for their effort in guiding me during my thesis and their time. Additionaly, their feedback, precision and insights motivated me and made this process a joyfull experience. I would also like to thank my supervisor Wieb Bosma from Radboud University for the opportunity to write my thesis at Tilburg University and the support during this process.

Secondly, I would like to thank my friends, especially Marrit de Jong, Clara Kappen and Evalien Koenen for giving me advice, motivating me and studying with me at the University Library in Nijmegen. Also outside our studying hours, they have supported me and were always there with me and for me.

To my parents, my parents in law, my sister Mila de Moor and my partner Burak Deniz, thank you for always being their for me, for supporting me and encouraging me, not only during the writing of my thesis but also throughout my life in general.

Contents

Chapter 1

Introduction

Consider the following problem, introduced by Bergantinos and Lorenzo (2004): a village consists of several houses. The houses in the village do not have a common water supply, but each house in the village has a private well. Because not every private well provides enough water in some periods of the year, the village authority decides to construct a water deposit by building a dam. Through pipelines, every house can be connected to the water deposit. This can be a direct connection or the houses can connect to the water deposit via other houses. The construction of the water deposit is payed by the village authority and acces to the water deposit is free for every house in the village. The only cost the villagers have to incur is the cost of constructing the pipelines to connect the houses to the water deposit. Because villagers pay the cost of constructing the pipelines, the villagers tell the village authority which pipelines to construct and which pipelines not to construct. The villagers now wonder which pipelines they need to construct in order to connect all houses to the water deposit at minimum cost. Furthermore, they wonder how to divide this minimal cost among the houses in a fair way.

Finding an optimal construction in which every house is connected to the common water supply lies in the field of operations research. In this area, the main issues explored are providing efficient algorithms and the computational complexity of these algorithms. A closely related discipline is game theory. Whenever the actions of several involved parties (called agents) are interdependent, game theory enters the scene. Modern day game theory is based on the work of John von Neumann and Oskar Morgenstern (1944), who published the book called "Theory of Games and Economic Behavior" and can be divided into two parts: cooperative game theory and non-cooperative game theory. In cooperative game theory, agents can make binding agreements and can form coalitions – a group of cooperative agents – in order to minimize the joint cost. In non-cooperative game theory however, there are no binding agreements and agents make their decisions independently rather than to cooperate. In the example illustrated above, the problem of fairly dividing the minimal total cost of constructing the pipelines among the houses is a problem cooperative game theory is concerned with.

The subject of this thesis is concerned with a problem which deals with both operations research and cooperative game theory and is called the *Minimum Cost Spanning Tree (mcst) problem.* The mcst problem can be explained as follows: consider a group of agents located at different places. They all need to be connected to the same source, where the source stands for some particular supplier of a service. This can be a direct connection, or the agents can connect to the source via other agents. Every possible connection is associated with some known nonnegative cost. In the previous example, the houses represent the agents and the water supply represents the source. The problem is to first find a network of minimum cost that connects all agents to the source. It turns out that this network of minimum cost is a minimum spanning tree, referred to as a *minimum cost spanning tree (mcst).* Secondly, when the minimum cost of the network is known, the problem is to divide the total cost among the agents in a fair way.

There are many algorithms that find an mcst in the mcst problem. Graham and Hell (1985) published a historic overview of algorithmic solutions of the problem. The most commonly used algorithms are Kruskal's algorithm (Kruskal, 1956) and Prim's algorithm (Prim, 1957). Another algorithm worth mentioning for this thesis is the vertex oriented construct procedure (Çiftçi and Tijs, 2009).

Claus and Kleitman (1973) were the first to look at how the total cost of the constructed network – an mcst – should be divided among the different agents in a fair way. Bird (1976) was the first to study this problem using game theory. This part of the mcst problem can be formulated as a cooperative game, and is called the *mcst game*, where a distinction can be made between *pessimistic mcst games* and *optimistic mcst games.* A core element is considered as a fair cost allocation. The core consists of all allocations that are efficient, meaning the sum of the costs allocated to the agents is equal to the cost of an mcst, and for which no coalition has an incentive to deviate.

Instead of looking at mcst games, one can also look at the algorithms used for constructing an mcst to answer the question how to fairly divide the total cost of the optimal network among the agents. Examples of allocation methods that make use of known algorithms for constructing an mcst are the *Bird rule* (Bird, 1976) which relies on Prim's algorithm, the *equal remaining obligations rule (ERO)* (Feltkamp, Tijs and Muto, 1994) which can rely on both Kruskal's algorithm as on the vertex oriented construct procedure and the *vertex oriented construct and charge procedure* (voccp) (Çiftçi and Tijs, 2009) which relies on the vertex oriented construct procedure. Bird (1976) and Granot and Huberman (1981) showed that the allocation obtained from the Bird rule is always an element of the core and therefore the core of every mcst game is non-empty. It turns out that also every allocation obtained from the voccp is an element of the core. Careful attention has been payed in this thesis to prove this claim because remarkably, this proof is nowhere explicitly found in the literature.

Extensive research is done concerning mcst problems. However, there is little to no research done in mcst problems in which connections can be constructed at two points in time, called the *Minimum Cost Spanning Tree with Time (mcstt) problem.* Again consider the mcst problem in which houses need to be connected to a common water supply. Villagers now get the option to connect to the water supply in a lesser time span. The cost of the construction of the pipelines will then increase, because of an increase in labour cost for example. In contrast, if villagers choose to wait for the moment in which the pipelines are finished as planned to connect to the water supply, they incur an *additional cost* – extra costs on top of the construction costs for waiting until the second point in time to connect to the water supply. These additional costs differ per house. The higher an additional cost, the higher is the need for that particular house to connect to the source in a faster time span, i.e., at the first point in time.

The purpose of this thesis is to explore the mcstt problem. Again, the problem is to first find a network of minimum cost that connects all agents to the source either at the first or the second point in time. This network of minimum cost is called a *minimum cost spanning tree with time (mcstt).* Other than in the mcst problem, in the mcstt problem it is important to know at which time the connections between agents or an agent and the source are constructed. Also, connections between agents can only be constructed at the first point in time if those agents are also connected to the source by some other connection at the first point in time. Furthermore, at the second point in time, all agents need to be connected to the source. The properties an mcstt meets, are refered to as the shape of an mcstt.

The difficulty of finding an mcstt in an mcstt problem lies in finding which agents connect to the source at which point in time. Once we know which agents connect to the source at the first point in time, we can use any combination of Prim's algorithm, Kruskal's algorithm and the vertex oriented construct procedure at the two points in time to obtain an algorithm that finds an mcstt in the mcstt problem.

When the cost of the mcstt is known, the problem is to divide this cost among the agents in a fair way. In order to obtain a fair cost allocation we look at the so called *mcstt games.* Where we could use Prim's algorithm, Kruskal's algorithm and the vertex oriented construct procedure to obtain an mcstt in the mcstt problem, unfortunately we can not use the combination of the corresponding allocation rules to obtain a cost allocation vector that is an element of the core. Based on various examples of mcstt problems, it seems reasonable to conjecture that the corresponding mcstt games have a non-empty core. However, an allocation method that constructs an allocation vector that is an element of the core for any mcstt problem is not yet found.

A special class of mcstt problems are the so called *Minimum Cost Spanning Tree Problems with Time with Constant Reduced Edges (mcsttr problems)* in which we assume that the costs of the connections between the agents and between the agents and the source are all reduced by a constant amount at the second point in time. It turns out that for an mcstt problem with three agents, the core of the associated mcstt game is nonempty. By making a distinction between which agents connect to the source at which point in time, for each case a fair allocation method is found for the mcsttr problem with three agents.

Other special classes of mcstt problems are for example the mcstt problem in which the cost of all connections between the agents and between the agents and the source at the second point in time have equal cost or the mcstt problem in which the cost of all connections between agents and between the agents and the source at the first point in time are reduced by a certain factor at the second point in time. Because at first sight, finding an allocation method for these problems also does not appear to be straight forward, this thesis only pays particular attention to mcsttr problems.

This thesis is structured in the following way. In Chapter 2, some basic notions of graph theory and cooperative game theory are given that are needed for this thesis. Chapter 3 considers the mcst problem and is divided into 3 sections. Section 3.1 explores the construction of a network of minimum cost in mcst problems and some efficient algorithms are given. In section 3.2, mcst games are considered. Section 3.3 explores fair cost allocation methods for the mcst problem. Chapter 4 considers the mcstt problem and is divided into 4 sections. Section 4.1 explores the shape of an optimal network in the mcstt problem. In section 4.2, an algorithm for the mcstt problem is provided in case we know which agents connect to the source at which point in time. In section 4.3 the mcstt game is defined and its properties are explored and section 4.4 tries to find a fair cost allocation method for the mcstt problem. Chapter 5 considers the mcsttr problem and is also divided into 4 sections. In section 5.1 the mcsttr problem is introduced and some properties of the associated mcsttr games are given. Section 5.2 tries to find an answer to the question how to find which agents connect to the source at which point in time. In section 5.3 a comparison is made between an mcstt in the mcsttr problem and an mcst in the mcst problem at the first point in time in order to know more about the shape of an mcstt in the mcsttr problem. In section 5.4 a fair cost allocation method is given for the mcsttr problem with three agents.

Chapter 2

Preliminaries

This chapter gives an overview of the concepts used in this thesis. In section 2.1, basic notions from graph theory are given. Section 2.2 consists of basic notions from cooperative game theory that are necessary to explore for this thesis.

2.1 Basic notions from graph theory

A *graph* G is a pair (V, E), where V is a finite set of *vertices* and E a finite set of *edges*. Each edge $e \in E$ connects a vertex $u \in V$ with a vertex $v \in V$ and is denoted by the pair $\{u, v\}$. Here, u and v are called the *endpoints* of e.

A graph is called *complete* if it contains an edge for every pair of distinct vertices, i.e., there exists an edge $e \in E$ for every pair of vertices $u, v \in V$ with $u \neq v$, and is denoted by (V, E_V).

The graph $G' = (V', E')$ is called a *subgraph* of the graph $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E_{V'} \cap E$, meaning each edge $e \in E'$ has the same endpoints in G' as in G.

We may denote the set of vertices and the set of edges of a graph G by $V(G)$ and $E(G)$ respectively. If E is a set of edges, we denote the set of vertices corresponding to the endpoints of the edges in E by $N(E)$.

Let $G = (V, E)$ be a graph and let $< v_1, \ldots, v_{k+1} >$ be a sequence of vertices where $v_1, \ldots, v_{k+1} \in V$. Let $e_i = \{v_i, v_{i+1}\}$ be the edge that connects vertex v_i with vertex v_{i+1} for every $i \in \{1, \ldots, k\}$. We call the set of those edges $\{e_1, \ldots, e_k\}$ a *path* P of size k. Because the edges in this path P connect the vertex v_1 with vertex v_{k+1} through $v_2, \ldots, v_k$, this path is also called a path from v_1 to v_{k+1}. A *cycle* is a path where $v_{k+1} = v_1$.

A graph $G = (V, E)$ is called *connected* if there exists a path from u to v for every $u, v \in V$. A subgraph G' of G is called a *component* of G if G' is connected and if there is no edge that connects a vertex in $V(G')$ to a vertex in $V(G) \setminus V(G')$. A subgraph G' of G is called a tree if it is connected and it does not contain any cycles. A tree G' containing all vertices of G is called a *spanning tree.* Note that a spanning tree G' has exactly $V(G') - 1$ edges.

2.2 Basic notions from cooperative game theory

Let $N = \{1, \ldots, n\}$ be a set of agents. The collection of all subsets of N, also called the *powerset* of N, is denoted by 2^N. A subset $S \in 2^N$ of agents is called a *coalition.* A *(cost) game* is a pair (N, c) where $N = \{1, \ldots, n\}$ is the set of agents and $c : 2^N \to \mathbb{R}$ is a mapping called the *characteristic function* with $c(\emptyset) = 0$. $c(S)$ is called the *cost* of the coalition S and represents the cost coalition S makes if the agents in that coalition would cooperate.

Example 2.2.1. *Consider the cost game* (N, c) *presented in Table 2.2.1. The cost that for example agent 1 incurs if she operates alone would be* 10 *and the cost that agent 2 incurs if she operates alone would be* 6*, while if agent 1 and agent 2 were to cooperate, the total cost would be* 11*.*

S	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	N
$c(S)$	0	10	6	8	11	12	9	13

Table 2.2.1: A cost game

A game (N, c) is called *monotone* if $c(S) \leq c(T)$ for all $S, T \in 2^N$ with $S \subseteq T$. A game (N, c) is called *subadditive* if $c(S) + c(T) \geq c(S \cup T)$ for all $S, T \in 2^N$ with $S \cap T = \emptyset$. In other words, there is no point in breaking up a coalition into parts, because the cost of the two parts together would be higher than the cost of the coalition itself. A cost game (N, c) is called *submodular* if $c(S) + c(T) \geq c(S \cap T) + c(S \cup T)$ for every $S, T \in 2^N$. A cost game (N, c) is called *supermodular* if $c(S) + c(T) \leq c(S \cap T) + c(S \cup T)$ for every $S, T \in 2^N$.

An *allocation* or *cost vector* $x \in \mathbb{R}^N$ assings a cost x_i to every agent $i \in N$. An allocation is called *efficient* if $\sum_{i \in N} x_i = c(N)$. The set of efficient allocations for which no coalition $T \subseteq S$ has an incentive to leave S is called the *core* of a cost game (N, c) and is defined by

$$\text{Core}(c) = \left\{ x \in \mathbb{R}^N \,\middle|\, \sum_{i \in N} x_i = c(N), \sum_{i \in S} x_i \leq c(S) \text{ for all } S \subseteq N \right\}$$

Example 2.2.1 (continued)**.** *The core of (N,c) is given in figure 2.2.1. We have the following equations:*

$$\begin{aligned} x_1+x_2+x_3 &= 13, \\ x_1 &\leq 10, \\ x_2 &\leq 6, \\ x_3 &\leq 8, \\ x_1+x_2 \leq 11 &\implies x_3 \geq 2, \\ x_1+x_3 \leq 12 &\implies x_2 \geq 1, \\ x_2+x_3 \leq 9 &\implies x_1 \geq 4. \end{aligned}$$

Now $x_1 \leq 10$, $x_2 \leq 6$ and $x_3 \leq 8$ are reflected by the dotted line segments, while $x_1 \geq 4$, $x_2 \geq 1$ and $x_3 \geq 2$ are reflected by the black line segments. So

$$\text{Core}(c) = \text{conv}\{(10,1,2),(4,1,8),(4,6,3),(5,6,2)\}.$$

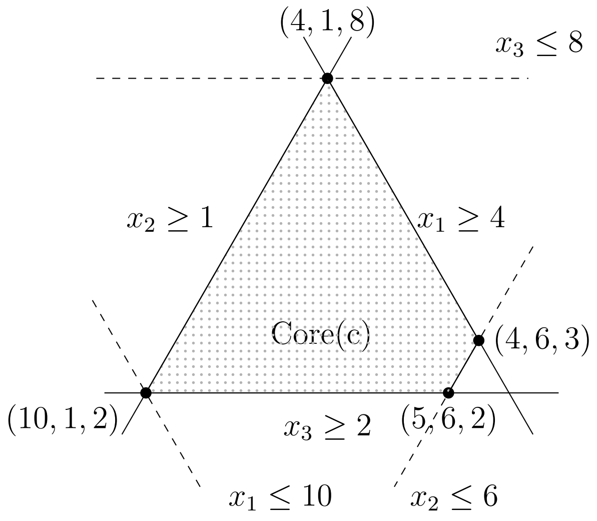


Figure 2.2.1: The core of the cost game presented in table 2.2.1

It is well known that a subadditive 3-agent game (N,c) with $N=\{1,2,3\}$ has a nonempty core if and only if $c(\{1,2\})+c(\{1,3\})+c(\{2,3\} \geq 2c(N)$.

Let $\Pi(N)$ denote the set of all bijections $\sigma : \{1, \dots, n\} \to N$ and let $\sigma(k)$ represent the i-th agent such that agent i is in the k-th position with respect to σ. A *marginal vector* $m^\sigma(c)$ with respect to the ordering σ of agents is defined as $m^\sigma(c) = (m^\sigma_{\sigma(1)}(c), \dots, m^\sigma_{\sigma(n)}(c))$ where

$$m^\sigma_{\sigma(k)}(c) = c(\{\sigma(1), \dots, \sigma(k)\}) - c(\{\sigma(1), \dots, \sigma(k-1)\})$$

for $k \in \{1, \dots, n\}$. It represents the extra cost (or value) if agent $\sigma(k)$ would join the coalition $\{\sigma(1), \dots, \sigma(k-1)\}$. It is the marginal contribution of an agent with respect to σ. The convex hull of the marginal vectors is called the *Weber set* and is denoted by $W(c)$ (Quant, Borm, Reijnierse and van Velzen, 2005), i.e.

$$W(c) = \text{conv}\{m^\sigma(c) | \sigma \in \Pi(N)\}.$$

It turns out that the Weber set of a cost game (N, c) is equal to the core of (N, c) if and only if (N, c) is submodular.

Theorem 2.2.2. *Let (N, c) be a cost game. Then $W(c) = Core(c)$ if and only if (N, c) is submodular.*

Proof. A proof can be found in Shapley (1971) and Ichiishi (1981). □

The *Shapley value* denoted by Φ is defined as the average of the marginal vectors over all possible orders σ in $\Pi(N)$, i.e.

$$\Phi(c) = \frac{1}{n!} \sum_{\sigma \in \Pi(N)} m^\sigma(c)$$

Example 2.2.1 (continued)**.** *The marginal vectors with respect to the ordering $\sigma \in \Pi(N)$ of the cost game (N, c) are given in table 2.2.2.*

σ	$m^\sigma_1(c)$	$m^\sigma_2(c)$	$m^\sigma_3(c)$	$m^\sigma(c)$
(123)	10	1	2	$(10, 1, 2)$
(132)	10	1	2	$(10, 1, 2)$
(213)	5	6	2	$(5, 6, 2)$
(231)	4	6	3	$(4, 6, 3)$
(312)	4	1	8	$(4, 1, 8)$
(321)	4	1	8	$(4, 1, 8)$

Table 2.2.2: Marginal vectors of the cost game presented in table 2.2.1

We have $W(c) = \text{conv}\{(10,1,2),(4,1,8),(4,6,3),(5,6,2)\}$. *Because* $W(c) = \text{Core}(c)$*, this game is submodular. The Shapley value is now equal to*

$$\begin{aligned}\Phi(c) &= \left(\frac{(10+10+5+4+4+4)}{6}, \frac{(1+1+6+6+1+1)}{6}, \frac{(2+2+2+3+8+8)}{6}\right)\\ &= (6\frac{1}{6}, 2\frac{2}{3}, 4\frac{1}{6}).\end{aligned}$$

Chapter 3

Minimum Cost Spanning Tree Problems and Games

In the mcst problem, a group of agents all need to be connected to a common source where every possible connection is associated with some known nonnegative cost. The mcst problem can be divided into two subproblems. The first problem is an optimization problem: how to construct a network of minimum cost that connects all agents to the source? The second problem is an allocation problem: how to fairly allocate the cost of this constructed network among the agents?

In section 3.1, the problem of constructing such a network of minimum cost is considered. The shape of an optimal network is explored and some algorithms solving this part of the problem are given. In section 3.2 and 3.3, the second part of the mcst problem is considered. In section 3.2 the concepts of mcst games and irreducible mcst games are introduced and in section 3.3 some allocation rules that rely on the algorithms explored in section 3.1 are given.

3.1 Optimization: Constructing a network of minimum cost

Let $N = \{1, \ldots, n\}$ be a set of agents, let 0 represent the source and let $N_0 = N \cup \{0\}$. In particular we write $S_0 = S \cup \{0\}$ for every finite set S. Let $w : E_{N_0} \to \mathbb{R}_+$ be the *weight function*, where $w(e)$ is the cost of constructing edge $e \in E_{N_0}$.

The goal is to connect all agents to the source, either directly or via other agents, at minimum cost. To do this, we can look at the complete graph (N_0, E_{N_0}) with weightfunction w on the edges $e \in E_{N_0}$. When we look at the example where houses need to be connected to a water supply through pipelines, we can represent the pipelines by the set of edges E_{N_0}, the houses by the set of vertices N and the source by the vertex 0. Every pipeline has an associated cost to it, which represents the amount of money that is needed to construct that particular pipeline. The cost is represented by the weight function w, that assigns a cost to every edge.

Let E be a set of edges. We define the cost of constructing all edges of E by $w(E) = \sum_{e \in E} w(e)$. When we speak of the cost of a graph $G = (V, E)$ we actually mean the cost of its edge set, i.e. $w(E)$.

Looking at the graph (N_0, E_{N_0}), we want to use a set of edges $T \subset E_{N_0}$ such that the edges in T connect all agents to the source, i.e., (N_0, T) is a connected graph and the cost of T is minimal. Note that (N_0, T) has to be a spanning tree. Obviously, it contains all vertices in N_0 and if T would contain a cycle, we could just remove an edge in this cycle and still have a graph that connects all agents to the source. Note that because the weights are nonnegative, removing an edge results in a total cost equal to at most $w(T)$. Therefore, T does not contain a cycle. In fact, (N_0, T) is a *minimum cost spanning tree (mcst)*, because it is a spanning tree such that $w(T)$ is minimal. We represent an mcst problem by the triple $(N, 0, w)$. The cost of an mcst can now be defined as

$$m(N, 0, w) = \min \Big\{ \sum_{e \in T} w(e) \mid (N_0, T) \text{ is a spanning tree in } (N_0, E_{N_0}) \Big\}.$$

The network of minimum cost that connects all agents to the source in an mcst problem $(N, 0, w)$ is an mcst in the graph (N_0, E_{N_0}).

Example 3.1.1. *Suppose there are three houses which need to be connected to a common water supply through pipelines. The houses can connect to the water supply directly or via other houses. Figure 3.1.1a shows the cost of constructing these connections between the houses and the water supply. Figure 3.1.1b shows an mcst in the mcst problem given in figure 3.1.1a. The three houses and the water supply are the vertices of the graph, denoted by 1, 2, 3 and 0 respectively. The mcst in this mcst problem has total cost 13.*

Note that if there are multiple mcst's in an mcst problem $(N, 0, w)$, then they all still have the same cost.

3.1.1 Algorithms

Multiple algorithms can be used to construct an mcst in an mcst problem. The algorithms explained in this section are Prim's algorithm, Kruskal's algorithm and the vertex oriented construct procedure.

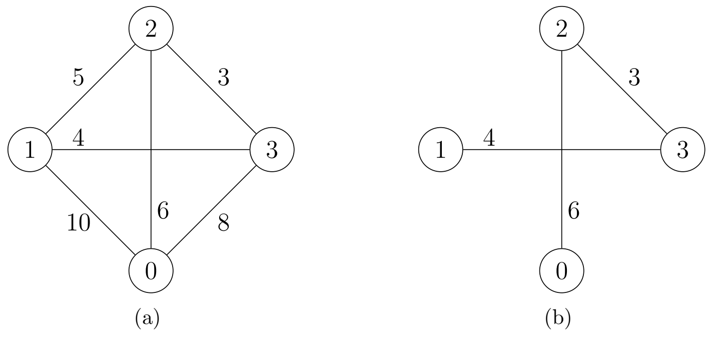


Figure 3.1.1: (a) An mcst problem. (b) An mcst in the graph presented in figure 3.1.1a

In Prim's algorithm, in the first step a cheapest edge is constructed between the source and an agent. In every next step of the algorithm, a cheapest edge is constructed between the source or an agent already connected to the source in a previous step of the algorithm, to an agent not yet connected to the source, making sure that no cycles are created. Prim's algorithm is shown in Algorithm 1.

Algorithm 1 Prim's algorithm

```
  Input: $|N|$ vertices, source 0, weight function $w : E_{N_0} \to \mathbb{R}_+$.
  Output: Minimum cost spanning tree $(N_0, T)$ that connects all vertices in $N$ to 0.

1: Initialize $S = \{0\}, T = \emptyset$.
2: Find the cheapest edge $e \in E_{N_0}$ between a vertex $j$ in $S$ and a vertex $i$ in $N_0 \setminus S$
   such that the graph $(N_0, T \cup \{e\})$ does not contain a cycle.
3: Add vertex $i$ to the set $S$ and edge $e$ to the set $T$.
4: If all vertices are connected to the source, the algorithm terminates. If not, go
   back to step 2.
```

Kruskal's algorithm first orders all edges by increasing cost in a list. In every step of the algorithm, a cheapest edge is selected and is only added to the spanning tree if adding this edge means not creating a cycle with the added edges in the previous steps of the algorithm. Otherwise, this edge is removed from the ordered list of edges and a next cheapest edge is selected. The algorithm is shown in Algorithm 2.

With the vertex oriented construct procedure agents are ordered by a certain permutation. In each step of the algorithm, the first agent in the order who has not yet constructed an edge may construct an edge with minimal cost, connecting

Algorithm 2 Kruskal's algorithm

```
Input: $|N|$ vertices, source 0, weight function $w : E_{N_0} \to \mathbb{R}_+$.
Output: Minimum cost spanning tree $(N_0, T)$ that connects all vertices in $N$ to 0.

1: Initialize $T = \emptyset$.
2: Let $L :=< e_1, \dots, e_m >$ be a list of edges ordered by increasing cost.
3: Pick the first edge $e \in L$ (this is the edge with the lowest cost).
4: If the graph $(N_0, T \cup \{e\})$ has a cycle, remove $e$ from $L$ and go to step 3.
5: If the graph $(N_0, T \cup \{e\})$ has no cycle, remove $e$ from $L$ and add $e$ to $T$.
6: If all vertices are connected to the source, the algorithm terminates. If not, go
   back to step 3.
```

the component that the agent belongs to with a component that the agent does not belong to.

Algorithm 3 Vertex oriented construct procedure

```
Input: $|N|$ vertices, source 0, weight function $w : E_{N_0} \to \mathbb{R}_+$.
Output: Minimum cost spanning tree $(N_0, T_\sigma)$ that connects all vertices in $N$
to 0.

1: Pick $\sigma \in \Pi(N)$.
2: Initialize $T_\sigma^0 = \emptyset$.
3: From $k = 1$ to $k = |N|$, find an edge $e \in E_{N_0}$ of minimal cost that connects
   an agent in the component of $(N_0, T_\sigma^{k-1})$ which contains $\sigma(k)$, to an agent in a
   component of $(N_0, T_\sigma^{k-1})$ which does not contain agent $\sigma(k)$ and add edge $e$ to
   $T_\sigma^{k-1}$.
4: Set $T_\sigma = T_\sigma^{|N|}$ and return $(N_0, T_\sigma)$.
```

Example 3.1.2. *Let $(N, 0, w)$ be the mcst problem given in figure 3.1.1a.*

Prim's algorithm starts at the source and finds the cheapest edge to whom the source can connect, so the algorithm constructs the edge $\{0, 2\}$. Then the algorithm constructs edge $\{2, 3\}$ because this is the cheapest of all edges that connects an agent outside the set $\{0, 2\}$ to an agent in the set $\{0, 2\}$. Finally, because the edge $\{1, 3\}$ is the cheapest edge that connects agent 3 to an agent in the set $\{0, 1, 2\}$ without creating a cycle, this edge is constructed by the algorithm. We now have an mcst with its edge set equal to $\{\{0, 2\}, \{2, 3\}, \{1, 3\}\}$.

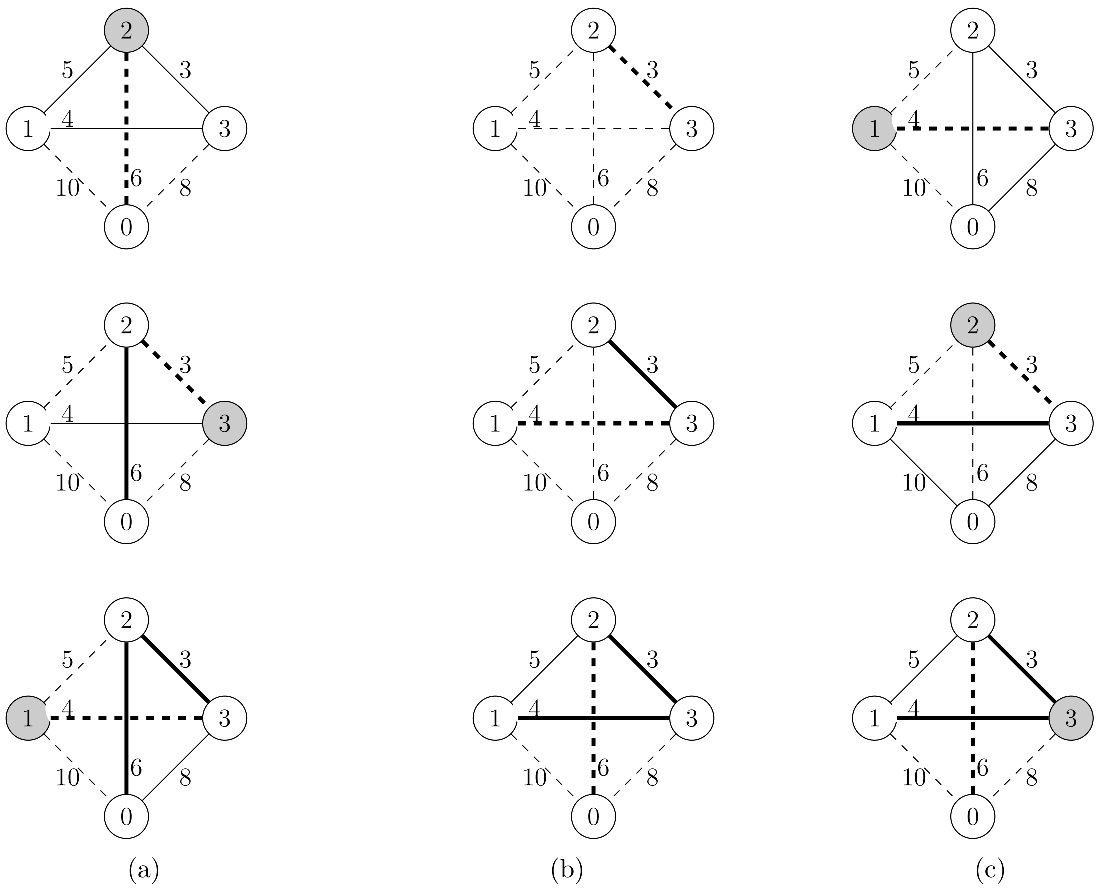


Figure 3.1.2: (a) Prim's algorithm. (b) Kruskal's algorithm. (c) Vertex oriented construct procedure for $\sigma = (123)$.

Kruskal's algorithm first constructs the edge $\{2,3\}$ *since it is the cheapest of all edges. Because* $\{1,3\}$ *is the second cheapest edge, the algorithm constructs the edge* $\{1,3\}$*. Because adding the third cheapest edge creates a cycle with the previously constructed edges, this edge cannot be added to* T*. So edge* $\{0,2\}$ *is constructed. We now have an mcst with its edge set equal to* $\{\{2,3\},\{1,3\},\{0,2\}\}$.

Let $\sigma = (123)$*. So* $\sigma(1) = 1, \sigma(2) = 2$ *and* $\sigma(3) = 3$*. The vertex oriented construct procedure first constructs the edge* $\{1,3\}$ *since it is the cheapest edge that connects agent 1 with another agent or the source. Moving on to agent 2, the algorithm constructs edge* $\{2,3\}$ *since this is the cheapest edge that connects agent 2 to another agent or the source. Moving on to agent 3, agent 3 is part of the component consisting of agents 1, 2 and 3. The cheapest edge that connects this component to another component in the graph (in this case the source) is edge* $\{0,2\}$*, so the algorithm* *constructs edge* $\{0,2\}$ *and we get the mcst with the set of edges* $\{\{1,3\},\{2,3\},\{0,2\}\}$.

In figure 3.1.2, this example is showed step for step. Here, a grey vertex represents the agent who is going to construct the edge, the dashed edges are the edges that are allowed to construct and the thick edges represent the edges the agents chose to construct.

Note that in figure 3.1.2c, in the last step agent 3 constructs an edge not incident to agent 3. So in the vertex oriented construct procedure, it is possible for an agent to construct an edge not incident to that agent, while with Prim's- or Kruskal's algorithm this is not possible.

Theorem 3.1.3. *Let* $(N,0,w)$ *be an mcst problem. Let* (N_0,E_{N_0}) *be the associated graph. Then Prim's algorithm, Kruskal's algorithm and the vertex oriented construct procedure all construct an mcst in* $(N,0,w)$.

Proof. For the proof concerning Prim's algorithm and Kruskal's algorithm we refer to (Prim, 1957) and (Kruskal, 1956) respectively.

In the vertex oriented construct procedure, for each $k\in\{1,\dots,n\}$, an edge is added to the graph (N_0,T_σ^{k-1}) such that this edge does not create a cycle with the previous added edges. Therefore, T_σ has n edges and (N_0,T_σ) does not contain a cycle. Since a spanning tree of (N_0,E_{N_0}) has n edges, connects all vertices in N_0 and does not contain any cycles, (N_0,T_σ) is a spanning tree of (N_0,E_{N_0}).

Now suppose (N_0,T) is a spanning tree in (N_0,E_{N_0}) other than (N_0,T_σ). Then there exists an edge $e\in T_\sigma$ such that $e\notin T$. Suppose this edge is created in the k-th step of the algorithm and call this edge e_k. Suppose (N_0,T) is an mcst such that the smallest value of k for which e_k is not an edge in T is as large as possible. Call this value $f(T)$. Also assume that (N_0,T_σ) is not an mcst in $(N,0,w)$. We thus have $e_k\in T_\sigma$ and $e_k\notin T$. Then $e_1,\dots,e_{k-1}\in T\cap T_\sigma$ by definition of T, but $e_k\notin T$. Since (N_0,T) is a spanning tree in (N_0,E_{N_0}), adding e_k to T means creating a unique cycle C. There exists an edge e' in $C\setminus\{e_k\}$ such that e' connects a component of (N_0,T_σ^{k-1}) which contains agent $\sigma(k)$ with a component of (N_0,T_σ^{k-1}) which does not contain agent $\sigma(k)$ and because (N_0,T) is an mcst in $(N,0,w)$ we have $w(e_k)\geq w(e')$ (because otherwise we could delete the edge e' with $w(e')>w(e_k)$ and this would result in a spanning tree with lesser cost than T which is a contradiction because (N_0,T) is an mcst in $(N,0,w)$). On the other hand, since it is allowed for $\sigma(k)$ to construct e_k in the k-th step of the algorithm, we have $w(e_k)\leq w(e')$. Hence, $w(e')=w(e_k)$. So $T'=(T\cup\{e_k\})\setminus\{e'\}$ and thus (N_0,T') is also a spanning tree of (N_0,E_{N_0}) not equal to (N_0,T). Because $w(T)=w(T')$, (N_0,T') is also an mcst in $(N,0,w)$. But $f(T')>f(T)$, which is a contradiction because $f(T)$ was chosen as large as possible. Therefore, (N_0,T_σ) is an mcst in $(N,0,w)$.

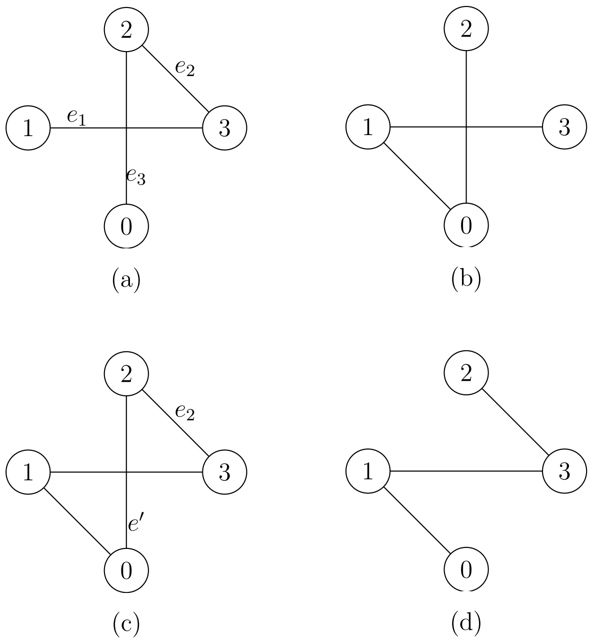


Figure 3.1.3: Illustration of the arguments used in the proof of Theorem 3.1.3. (a) Depicts the spanning tree (N_0, T_σ) where $\sigma = (123)$ constructed by the vertex oriented construct procedure. Here edge e_1 is the first edge, e_2 the second edge and e_3 the third edge constructed by the vertex oriented construct procedure. (b) Depicts the mcst (N_0, T). Note that (N_0, T) is an mcst such that the smallest value of k for which e_k is not an edge in T is as large as possible. Also note that in this case $k = 2$, since e_2 is the first edge constructed by the vertex oriented construct procedure such that $e_2 \in T_\sigma$ and $e_2 \notin T$. (c) Depicts the graph $(N_0, T \cup \{e_2\})$. We can see that by adding edge e_2 to T a cycle C is created. Since $\sigma = (123)$ we have $\sigma(2) = 2$. Therefore e' is the edge in C that connects agent 2 with a component of $(N_0, T^1_\sigma) = (N_0, \{\{1, 3\}\})$ that does not contain agent 2, which in this case is the vertex 0. (d) Depicts the spanning tree (N_0, T') with $T' = (T \cup \{e_2\}) \setminus \{e'\}$. Since $w(e_2) = w(e')$ we have $w(T) = w(T')$, therefore (N_0, T') is also an mcst. However, the first edge in which T_σ and T' differ is now e_3 instead of e_2 which is a contradiction.

$\square$

3.2 Minimum cost spanning tree games

In this section we introduce the concept of mcst games, where a distinction is made between pessimistic mcst games and optimistic mcst games. In addition a description of irreducible mcst games is provided.

3.2.1 Pessimistic minimum cost spanning tree games

Let $(N, 0, w)$ be an mcst problem. A *pessimistic mcst game* (N, c) assigns to every coalition S the cost of an mcst in the graph (S_0, E_{S_0}), i.e.

$$c(S) = \min\Big\{ \sum_{e \in T} w(e) \Big| T \subseteq E_{S_0} \text{ and } (S_0, T) \text{ is a tree} \Big\}.$$

So $c(S)$ is the minimal cost coalition S has to pay such that all agents in S are connected to the source, either directly or via each other, leaving the agents in $N \setminus S$ out of consideration and therefore also all connections to agents in $N \setminus S$. Because it is assumed that agents in $N \setminus S$ are not connected to the source, while these agents do want to be connected to the source, this is a rather pessimistic approach (hence the name).

When speaking of pessimistic mcst games we just refer to it as mcst games, since this is the standard way in the literature.

Example 3.2.1. *Reconsider the mcst problem $(N, 0, w)$ presented in figure 3.1.1. We have the following mcst game (N, c):*

S	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	N
$c(S)$	0	10	6	8	11	12	9	13

Table 3.2.1: Mcst game of the mcst problem $(N, 0, w)$ 3.1.1

If agent 1 wants to connect to the source individually, the cost of connecting agent 1 to the source is equal to the cost of edge $\{0, 1\}$ which is 10. For agent 2 this is equal to the cost of constructing edge $\{0, 2\}$ which is equal to 6 and for agent 3 this is equal to the cost of constructing edge $\{0, 3\}$ which is 8. If coalition $S = \{1, 2\}$ wants to connect to the source, leaving agent 3 out of consideration, they can only construct the edges $\{0, 1\}, \{0, 2\}, \{1, 2\}$ and the optimal way to do so is by constructing the edges $\{0, 2\}$ and $\{1, 2\}$ which have a joint cost of 11. Coalition $S = \{1, 3\}$ can only construct the edges $\{0, 1\}, \{0, 3\}$ and $\{1, 3\}$. The optimal network for agent 1 and 3 *would be constructing the edges* $\{0,3\}$ *and* $\{1,3\}$ *with a joint cost of* 12. *For coalition* $S = \{2,3\}$, *the optimal way to connect to the source would be constructing the edges* $\{0,2\}$ *and* $\{2,3\}$ *with a total cost of* 9. *Looking at the whole coalition* N *we have the mcst given in figure 3.1.1b which has a total cost of* 13.

Note that this game is the same game as in table 2.2.1. It is easily seen that mcst games are subadditive. However, mcst games are not always monotone nor are they always submodular as can be seen by the counterexample given in Example 3.2.2.

Example 3.2.2. *Consider the mcst problem given in figure 3.2.1 and the associated mcst game presented in table 3.2.2. This game is not monotone since* $\{3\} \subset \{2,3\}$ *but* $c(\{3\}) = 10 > 9 = c(\{2,3\}$. *This game is also not submodular since* $12 + 9 = c(\{1,3\}) + c(\{2,3\}) < c(\{1,3\} \cup \{2,3\}) + c(\{1,3\} \cap \{2,3\}) = c(N) + c(\{3\}) = 13 + 10$.

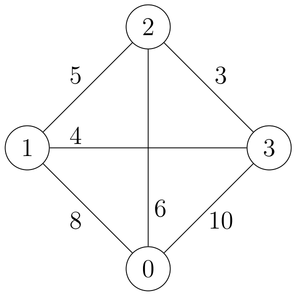


Figure 3.2.1: An mcst problem

S	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	N
$c(S)$	0	8	6	10	11	12	9	13

Table 3.2.2: The mcst game associated with the mcst problem in figure 3.2.1

3.2.2 Irreducible minimum cost spanning tree games

Let (N_0, T) be an mcst in the mcst problem $(N, 0, w)$. Let $e = \{i, j\}$. The path in T from i to j is denoted by $P_e(T)$. So these are the edges e in T such that these edges connect i to j. Define the weight function $w_I^T : E \to \mathbb{R}_+$ as the function that assigns to every edge $e = \{i, j\}$ the largest weight of all edges in $P_e(T)$, i.e.

$$w_I^T(e) = \max\{w(e') | e' \in P_e(T), (N_0, T) \text{ is an mcst in } (N, 0, w)\}$$

for all $e \in E_{N_0}$. Aarts and Driessen (1993) proved that this weight function does not depend upon the choice of the mcst (N_0, T). This result is stated in Theorem 3.2.4. The proof of Theorem 3.2.4 makes use of Lemma 3.2.3.

Lemma 3.2.3. *Let $(N, 0, w)$ be an mcst problem and let (N_0, T) be an mcst in $(N, 0, w)$. Then*

(i) $w_I^T(e) = w(e)$ for all $e \in T$.

(ii) $w_I^T(e) \leq w(e)$ for all $e \in E_{N_0}$.

(iii) (N_0, T) is an mcst in $(N, 0, w_I^T)$.

Proof.

(i) Let $e \in T$. Then $\{e\} = P_e(T)$. Hence, $w_I^T(e) = w(e)$ for all $e \in T$.

(ii) By Lemma 3.2.3(i) this holds for all $e \in T$. Suppose that for some $e = \{i, j\} \in E_{N_0} \setminus T$ we have $w_I^T(e) > w(e)$. Let $w_I^T(e) = w(e')$ for some $e' \in P_e(T)$. Since (N_0, T) is an mcst, (N_0, T) is a spanning tree. Therefore, adding e to T means creating a unique cycle C. Since $e' \in P_e(T)$, e' lies on the path from i to j, hence $e' \in C$. Therefore $T' = (T \cup \{e\}) \setminus \{e'\}$ is also a spanning tree in (N_0, E_{N_0}) and since $w(e') = w_I^T(e) > w(e)$ we have $w(T') < w(T)$, contradicting the fact that (N_0, T) is minimal.

(iii) Since (N_0, T) is an mcst in $(N, 0, w)$, (N_0, T) is a spanning tree in (N_0, E_{N_0}). Therefore, adding any edge $e \in E_{N_0} \setminus T$ to T would create a cycle $C = P_e(T) \cup \{e\}$. For all $e' \in P_e(T)$ we have $w_I^T(e') \leq w_I^T(e)$ by definition of the weight function w_I^T. Now let $T' = (T \cup \{e\}) \setminus \{e'\}$. Then $w_I^T(T') = w_I^T(T) + w_I^T(e) - w_I^T(e') \geq w_I^T(T)$. Since this holds for any $e \in E_{N_0} \setminus T$ and any corresponding $e' \in P_e(T)$, (N_0, T) is an mcst in $(N, 0, w_I^T)$.

□

Theorem 3.2.4. *Let $(N, 0, w)$ be an mcst problem and let (N_0, T) be an mcst in $(N, 0, w)$. Let (N_0, T') be another mcst in $(N, 0, w)$, so $T \neq T'$. Then (N_0, T') is also an mcst in the mcst problem $(N, 0, w_I^T)$ and $w_I^T(e) = w_I^{T'}(e)$ for all $e \in E_{N_0}$.*

Proof.
Because of Lemma 3.2.3 (i) and (ii) and because both (N_0, T) and (N_0, T') are an mcst in $(N, 0, w)$ we have

$$\sum_{e \in T'} w_I^T(e) \leq \sum_{e \in T'} w(e)$$

$$\begin{aligned} &= \sum_{e\in T} w(e) \\ &= \sum_{e\in T} w_I^T(e). \end{aligned}$$

Since (N_0, T) is an mcst in $(N, 0, w_I^T)$ by Lemma 3.2.3(iii) and (N_0, T') is a spanning tree in $(N, 0, w_I^T)$ of lower cost, (N_0, T') is also an mcst in $(N, 0, w_I^T)$. Therefore, we have

$$\sum_{e\in T'} w_I^T(e) = \sum_{e\in T} w_I^T(e). \tag{3.1}$$

Left to prove is $w_I^T(e) = w_I^{T'}(e)$ for all $e \in E_{N_0}$.

Suppose there exists an $e \in T'$ such that $w_I^T(e) \neq w(e)$. Then by Lemma 3.2.3(ii) we have that $w_I^T(e) < w(e)$. Then

$$\begin{aligned} \sum_{e\in T} w(e) &= \sum_{e\in T} w_I^T(e) \\ &= \sum_{e\in T'} w_I^T(e) \\ &< \sum_{e\in T'} w(e) \end{aligned}$$

contradicting the fact that (N_0, T') is an mcst in $(N, 0, w)$. Here, the first equality follows from Lemma 3.2.3(i), the second equality follows from (3.1) and the inequality follows from Lemma 3.2.3(ii). So

$$w_I^T(e) = w(e) \text{ for all } e \in T'.$$

Therefore, we have

$$\begin{aligned} w_I^{T'}(e) &= \max\{w(e') \mid e' \in P_e(T')\} \\ &= \max\{w_I^T(e') \mid e' \in P_e(T')\} \\ &= (w_I^T)_I^{T'}(e). \end{aligned}$$

Since (N_0, T') is an mcst in $(N, 0, w_I^T)$, by Lemma 3.2.3(ii) we have $(w_I^T)_I^{T'} \leq w_I^T(e)$ for all $e \in E_{N_0}$. Hence

$$w_I^{T'}(e) = (w_I^T)_I^{T'} \leq w_I^T(e) \text{ for all } e \in E_{N_0}.$$

Note that if we change the roles of the mcst's (N_0, T) and (N_0, T') we get

$$w_I^T(e) \leq w_I^{T'}(e) \text{ for all } e \in E_{N_0}.$$

Hence $w_I^T = w_I^{T'}$. □

Indeed, the weight function w_I^T is independent of the choice of the mcst (N_0, T) in $(N, 0, w)$. Therefore, we just write w_I instead of w_I^T. We can now define the *irreducible form* of the mcst problem $(N, 0, w)$ as the mcst problem $(N, 0, w_I)$ which was first stated by Bird (1976). The weight function w_I is called the *irreducible weight function.*

Example 3.2.1 (continued)**.** *The irreducible form $(N, 0, w_I)$ of the mcst problem $(N, 0, w)$ is given in figure 3.2.2a. The corresponding mcst is given in 3.2.2b.*

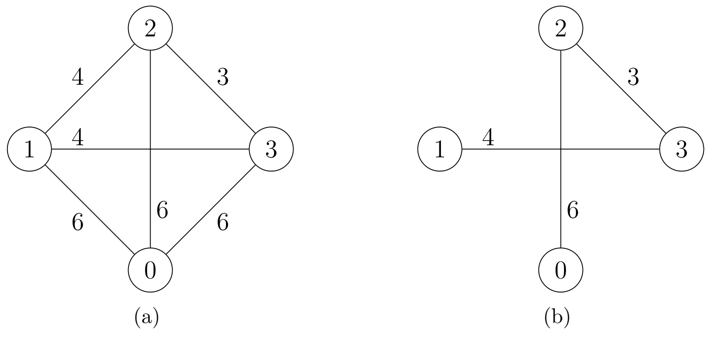


Figure 3.2.2: (a) The irreducible form of the mcst problem $(N, 0, w)$. (b) The corresponding mcst.

The maximum edge on the path in the mcst from agent 1 to agent 2 is 4 *so* $w_I(\{1, 2\}) = 4$*. The maximum edge on the path from the source to agent 1 and agent 3 in the mcst are both* 6 *so* $w_I(\{0, 1\}) = w_I(\{0, 3\}) = 6$*. The weights on the edges in the mcst remain the same.*

The *irreducible mcst game* associated with the mcst problem $(N, 0, w)$ is now defined as the game (N, c_I) that assigns to every coalition S the cost of an mcst in the mcst problem $(S, 0, w_I)$, i.e.,

$$c_I(S) = \min\{\sum_{e \in T} w_I(e) \mid T \subseteq E_{S_0} \text{ and } (S_0, T) \text{ is a tree}\}.$$

In other words, it is the mcst game of the mcst problem $(N, 0, w_I)$. $c_I(S)$ is also called the *irreducible cost of* S.

Example 3.2.1 (continued)**.** *The irreducible mcst game (N, c_I) corresponding to the mcst problem $(N, 0, w)$ is given in table 3.2.3. Note that (N, c_I) is the same as the mcst game corresponding to the mcst problem $(N, 0, w_I)$. If we compare the irreducible mcst game with the associated mcst game, we see that for $S \in \{\{2\}, \{2, 3\}, N\}$ we* *have $c(S) = c_I(S)$. This is because those coalitions only make use of the edges in the mcst.*

S	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	N
$c_I(S)$	0	6	6	6	10	10	9	13

Table 3.2.3: The mcst game corresponding to the mcst problem in figure 3.2.2

As seen in Example 3.2.1, for coalitions using only edges from the mcst (N_0, T) in $(N, 0, w)$ (or $(N, 0, w_I)$), the cost and the irreducible cost of S are the same, i.e., $c(S) = c_I(S)$. This is due to Lemma 3.2.3(i).

Because all mcst games are subadditive, the irreducible mcst game is subadditive. Let S and T be two coalitions in N such that $S \subseteq T \subseteq N$. For every $i, j \in S$, there is no path connecting i and j via agents in $T \setminus S$ that is cheaper than $w_I(\{i,j\})$ per definition of w_I. Therefore, irreducible mcst games are monotone. Moreover, Granot and Huberman (1977) showed that the irreducible mcst game is submodular.

Theorem 3.2.5. *Let $(N, 0, w)$ be an mcst problem and let (N, c_I) be the corresponding irreducible mcst game. Then (N, c_I) is submodular.*

Proof. A proof can be found in Granot and Huberman (1977). $\square$

The core of the irreducible mcst game (N, c_I) is called the *irreducible core* of (N, c) and is denoted by IC(c). Because the irreducible mcst game is submodular, the irreducible core of (N, c) is equal to the convex hull of the marginal vectors of the mcst game (N, c_I), i.e.

$$\mathrm{IC}(c) = \mathrm{conv}\{m^\sigma(c_I) \mid \sigma \in \Pi(N)\}.$$

Furthermore, it turns out that the irreducible core of (N, c) is a subset of the core of (N, c), i.e. $IC(c) \subseteq \mathrm{Core}(c)$. This result is stated in Theorem 3.2.6.

Theorem 3.2.6. *Let $(N, 0, w)$ be an mcst problem. Let (N, c) be the corresponding mcst game. Then we have*

$$IC(c) \subseteq \mathit{Core}(c).$$

Proof. Let (N, c_I) be the corresponding irreducible mcst game. By Lemma 3.2.3(ii) we have $c_I(S) \leq c(S)$ for all $S \subseteq N$ and by Lemma 3.2.3(iii) we have $c_I(N) = c(N)$. Therefore, for every $x \in IC(c)$ we have

$$\sum_{i \in S} x_i \leq c_I(S) \leq c(S) \text{ and } \sum_{i \in N} x_i = c_I(N) = c(N).$$

Hence $IC(c) \subseteq \mathrm{Core}(c)$. $\square$

3.2.3 Optimistic minimum cost spanning tree games

Recall that in a pessimistic mcst game, the cost of connecting coalition S to the source is equal to the cost of the mcst in the graph (S_0, E_{S_0}), and thus leaving the connections to agents in $N \setminus S$ out of consideration. In optimistic mcst games however, it is assumed that agents in $N \setminus S$ are already connected to the source and hence, for agents in S it would suffice to connect to the source through agents in $N \setminus S$. Agents can now not only connect to the source through agents in S but also through agents in $N \setminus S$. This means that for all edges between the source and an agent in S, not only that edge can be constructed but also an edge between that agent and $N \setminus S$. Because it is assumed that the agents in $N \setminus S$ are already connected to the source, this is a rather optimistic approach (hence the name).

We can now define the *optimistic cost* of coalition S, denoted by $c_O(S)$, as the minimum cost of connecting all agents in S to the source, where agents in S can not only connect to the source directly, but can also connect to the source via agents in $N \setminus S$. The notion of optimistic mcst games was first introduced by Bergantiños and Vidal-Puga (2007b).

Let $(N, 0, w)$ be an mcst problem. Define the *optimistic weight function for* S, denoted by w_S, as

$$w_S(e) = \begin{cases} w(e), & \text{if } e \in E_S \\ \min_{j \in (N \setminus S)_0} w(\{i, j\}), & \text{if } e = \{i, 0\} \text{ for all } i \in S. \end{cases} \tag{3.2}$$

The *optimistic mcst game* is now given by the game (N, c_O) that assigns to every coalition S the cost of an mcst in the mcst problem $(S, 0, w_S)$, i.e.,

$$c_O(S) = \min\{\sum_{e \in T} w_S(e) \mid T \subseteq E_{S_0} \text{ and } (S_0, T) \text{ is a tree}\}.$$

Example 3.2.1 (continued). *Consider coalition $S = \{1, 2\}$. The mcst problem $(S, 0, w_S)$ associated with the mcst problem $(N, 0, w)$ and the corresponding mcst are given in figure 3.2.3. Note that the vertex 0 in figure 3.2.3 not only represents the source, but represents $N \setminus S$ as well. The edges $\{0, 1\}$ and $\{0, 2\}$ are therefore the edges with minimal cost in the mcst problem $(N, 0, w)$ from agent 1 respectively agent 2 to $(N \setminus S)_0$. As can be seen in 3.2.3b, the total cost of connecting the agents in S to the source is 7.*

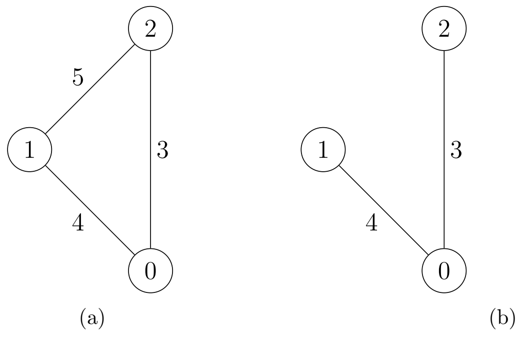


Figure 3.2.3: (a) The mcst problem $(S, 0, w_S)$ – where $S = \{1, 2\}$ – associated with the mcst problem $(N, 0, w)$. (b) The corresponding mcst.

The optimistic mcst game corresponding to the mcst problem $(N, 0, w)$ *is given in table 3.2.4.*

S	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	N
$c_O(S)$	0	4	3	3	7	7	7	13

Table 3.2.4: The optimistic mcst game corresponding to the mcst problem $(N, 0, w)$.

Let $(N, 0, w)$ be an mcst problem. Two mcst games (N, c) and (N, c') are called *dual* if for all $S \in 2^N$ we have

$$c(S) + c'(N \setminus S) = c(N).$$

Bergantiños and Vidal-Puga (2007b) showed that the irreducible mcst game and the optimistic mcst game associated with the mcst problem $(N, 0, w)$ are dual games. This result is stated in Theorem 3.2.7.

Theorem 3.2.7. *Let* $(N, 0, w)$ *be an mcst problem. Let* (N, c_I) *be the irreducible mcst game corresponding to* $(N, 0, w)$ *and let* (N, c_O) *be the optimistic mcst game corresponding to* $(N, 0, w)$*. Then*

$$c_I(S) + c_O(N \setminus S) = c(N) \text{ for all } S \in 2^N$$

Proof. A proof can be found in Bergantiños and Vidal-Puga (2007b). □

Note that because of Theorem 3.2.7 we have $c_I(N) = c_O(N) = c(N)$. Furtermore, because optimistic and irreducible mcst games corresponding to the same mcst problem $(N, 0, w)$ are dual games, we have that optimistic mcst games are supermodular.

Corollary 3.2.8. *Let $(N, 0, w)$ be an mcst problem and let (N, c_O) be the corresponding optimistic mcst game. Then (N, c_O) is supermodular.*

Proof. Let (N, c) and (N, c_I) be the corresponding mcst game and irreducible mcst game respectively. Let $S, T \subseteq N$ be two coalitions. Then

$$\begin{aligned} c_O(S) + c_O(T) &= c(N) - c_I(N \setminus S) + c(N) - c_I(N \setminus T) \\ &= 2 \cdot c(N) - (c_I(N \setminus S) + c_I(N \setminus T)) \\ &\leq 2 \cdot c(N) - (c_I((N \setminus S) \cap (N \setminus T)) + c_I((N \setminus S) \cup (N \setminus T))) \\ &= 2 \cdot N - (c_I(N \setminus (S \cup T)) + c_I(N \setminus (S \cap T))) \\ &= c_O(S \cap T) + c_O(S \cup T) \end{aligned}$$

where the first and last equality come from the fact that (N, c_O) and (N, c_I) are dual games and the inequality comes from the fact that (N, c_I) is submodular. □

Example 3.2.1 (continued)**.** *The pessimistic, optimistic and irreducible mcst games associated with the mcst problem $(N, 0, w)$ are given in Table 3.2.5. When adding $c_I(S)$ to $c_O(N \setminus S)$ it is clear that for every coalition S, this sum equals $c(N)$.*

S	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	N
$c(S)$	0	8	6	10	11	12	9	13
$c_I(S)$	0	6	6	6	10	10	9	13
$c_O(S)$	0	4	3	3	7	7	7	13

Table 3.2.5: The pessimistic-, irreducible- and optimistic mcst game associated with the mcst problem $(N, 0, w)$.

Furthermore, the following theorem states that the marginal vectors corresponding to the orders $\sigma \in \Pi(N)$ of the irreducible mcst game (N, c_I) are equal to the marginal vectors corresponding to the inverse of these orders of the optimistic mcst game (N, c_O).

Theorem 3.2.2. *Let $(N, 0, w)$ be an mcst problem. Let (N, c_I) be the corresponding irreducible mcst game and let (N, c_O) be the corresponding optimistic mcst game. Then*

$$m^{\sigma}(c_I) = m^{\sigma^{-1}}(c_O).$$

for all $\sigma \in \Pi(N)$.

Proof. Pick $k \in N$ arbitrarily. Let $\sigma \in \Pi(N)$ be an arbitrary order. We have

$$\begin{aligned} m^{\sigma}_{\sigma(k)}(c_I) &= c_I(\{\sigma(1),\ldots,\sigma(k)\}) - c_I(\{\sigma(1),\ldots,\sigma(k-1)\}) \\ &= m(N,0,w) - c_O(N \setminus \{\sigma(1),\ldots,\sigma(k)\}) \\ &\quad - (m(N,0,w) - c_O(N \setminus \{\sigma(1),\ldots,\sigma(k-1)\}) \\ &= c_O(\{\sigma(k),\ldots,\sigma(|N|)\}) - c_O(\{\sigma(k+1),\ldots,\sigma(|N|)\}) \\ &= m^{\sigma^{-1}}_{\sigma^{-1}(k)}(c_O) \end{aligned}$$

where the second equality follows from the fact that (N, c_I) and (N, c_O) are dual games. □

Actually, Theorem 3.2.2 holds for all dual games. Because the marginal vectors of the irreducible mcst game correspond to the inverse of those marginal vectors of the optimistic mcst game associated with the same mcst problem, the Shapley values of the irreducible cost and the optimistic cost are equal.

Corollary 3.2.3. *Let $(N, 0, w)$ be an mcst problem. Let (N, c_I) be the corresponding irreducible mcst game and let (N, c_O) be the corresponding optimistic mcst game. Then the Shapley values of c_I and c_O are equal, i.e.*

$$\Phi(c_I) = \Phi(c_O).$$

3.3 Fair cost allocation

To answer the question how to fairly divide the total cost among the agents, one can look at the algorithms used for constructing an mcst. For each edge constructed by the algorithm, the cost of that constructed edge can be allocated among the agents by following so called *construct and charge rules.* Examples of construct and charge rules are the *Bird rule* (Bird, 1976), the *equal remaining obligations rule (ERO)* and the *vertex oriented construct and charge procedure (voccp)* (Çiftçi and Tijs, 2009). The Bird rule relies on Prim's algorithm, while *ERO* can rely on Kruskal's algorithm (Feltkam, Tijs and Muto, 1994) or on the vertex oriented construct procedure. We call the vertex oriented construct procedure together with the *ERO* allocation the vertex oriented construct and charge procedure.

For each mcst problem $(N, 0, w)$, an *allocation rule* is defined as a function $f(N, 0, w) : N \to \mathbb{R}_+$ such that $\sum_{i \in N} f_i(N, 0, w) = c(N)$, where (N, c) is the mct game corresponding to the mcst problem $(N, 0, w)$ and $f_i(N, 0, w)$ stands for the cost allocated to agent i.

3.3.1 Bird Rule

Bird (1976) and Granot and Huberman (1981) showed that the core of every mcst game is non-empty. As described in the previous section, one way to fairly allocate the joint costs is by means of a core allocation, which we know every mcst game has.

The Bird rule computes a core element of an mcst game as follows: for every edge constructed by Prim's algorithm, allocate the weight of that edge to the agent who joins the minimum spanning tree that was formed before constructing that edge. The Bird rule is an allocation rule defined as $B_i(N, 0, w) = w(\{i, i^0\})$ where i^0 is the first vertex in the path in the constructed mcst of $(N, 0, w)$ from i to the source. The Bird rule is given in Algorithm 4.

Algorithm 4 Bird rule

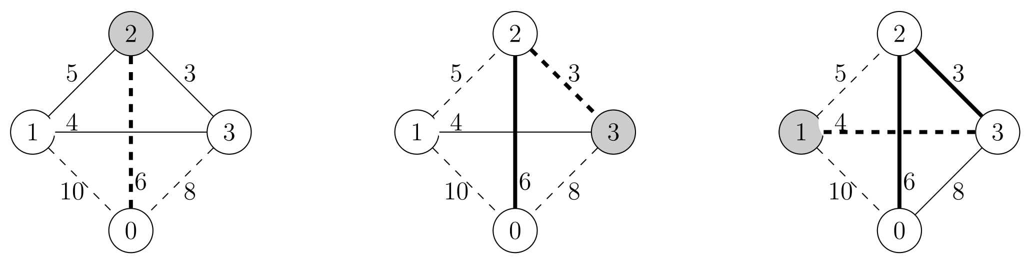
```
**Input:** $|N|$ vertices, source 0, weight function $w : E_{N_0} \to \mathbb{R}_+$.
**Output:** Minimum cost spanning tree $(N_0, T)$ that connects all vertices to 0 and
a cost allocation vector $B(N, 0, w)$.

1: Initialize $S = \{0\}, T = \emptyset$.
2: Find the cheapest edge $e$ between a vertex $j$ in $S$ and a vertex $i$ in $N_0 \setminus S$ such
   that the graph $(N_0, T \cup \{e\})$ does not contain a cycle.
3: Add vertex $i$ to the set $S$ and edge $e$ to the set $T$. Assign the cost $w(e)$ to agent
   $i$, i.e. $B_i(N, 0, w) = w(e)$.
4: If all vertices are connected to the source, the algorithm terminates. If not, go
   back to step 2.
```

Note that the only difference between Prim's algorithm (Algorithm 1) and the Bird rule (Algorithm 4) is the extra allocation rule in step 3 of the Bird rule.

Example 3.3.1. *Consider the mcst problem presented in figure 3.1.1a. In figure 3.3.1, Prim's algorithm with the corresponding Bird allocation is given step for step. Here a grey vertex represents the agent who is going to construct the edge, the dashed edges are the edges that are allowed to construct and the thick edges represent the edges the agents chose to construct.*

The first edge that is constructed by Prim's algorithm is $\{0, 2\}$. *Because agent 2 gets connected to the source, the cost of constructing edge* $\{0, 2\}$ *is assigned to agent 2. Next, agent 3 gets connected to the source by constructing the edge* $\{2, 3\}$ *so agent 2 gets assigned the cost of creating edge* $\{2, 3\}$. *Finally, agent 1 connects to the source by constructing edge* $\{1, 3\}$ *and therefore gets assigned the cost of constructing edge* $\{1, 3\}$. *This results in the allocation vector* $B(N, 0, w) = (4, 6, 3)$.



$B_2(N, 0, w) = 6$ $B_3(N, 0, w) = 3$ $B_1(N, 0, w) = 4$

Figure 3.3.1: The Bird rule step for step.

Note that if there are multiple mcst's in $(N, 0, w)$, the Bird allocation does not need to be the same.

3.3.2 Equal remaining obligations rule

The ERO rule can rely on Kruskal's algorithm or on the vertex oriented construct procedure. In this section, we explore the ERO rule relying on Kruskal's algorithm.

In this construct and charge procedure, we speak of connection vectors b^k. Here k stands for the k-th step of Kruskal's algorithm. In every step of the algorithm, for every agent i, b_i^k keeps track of how many agents are connected to that agent (including herself) and if an agent is connected to the source. If agent i is connected to the source we have $b_i^k = 0$. If not, b_i^k equals one over the number of agents that agent is connected to including herself.

Note that if an edge e is considered in the k-th step of Kruskal's algorithm such that it creates a cycle with the previously added edges, we have $b^k = b^{k-1}$.

The ERO rule – relying on Kruskal's algorithm – now computes an allocation vector as follows: we start with Kruskal's algorithm, but besides initializing $T = \emptyset$ we also initialize $b^0 = (1, 1, 1)$ and $ERO^0(N, 0, w) = (0, 0, 0)$ since no agent is connected by an edge to either another agent or the source. Then, for every edge e considered by Kruskal's algorithm, the cost of that edge is allocated to the agents, proportionally to the difference in the connection vectors resulting from the introduction of the edge in the $(k-1)$-th step and the edge e in the k-th step of Kruskal's algorithm. This allocation vector is denoted by $ERO^k(N, 0, w)$ and is then subsequently added to the allocation vector in the previous step of the algorithm, i.e., to $ERO^{k-1}(N, 0, w)$.

When all vertices are connected to the source, the output of the ERO rule is not only a minimum cost spanning tree but also the allocation vector corresponding to the allocation vector in the last step of the algorithm. The ERO rule is given in Algorithm 5.

Algorithm 5 Equal remaining obligations rule

```
Input: $|N|$ vertices, source 0, weight function $w : E_{N_0} \to \mathbb{R}_+$.
Output: Minimum cost spanning tree $(N_0, T)$ that connects all vertices in $N$ to 0 and a cost allocation vector $ERO(N, 0, w)$.

1: Initialize $T = \emptyset$, $b^0 = (1, 1, 1), ERO^0(N, 0, w) = (0, 0, 0)$.
2: Let $L :=< e_1, \dots, e_m >$ be a list of edges ordered by increasing cost.
3: Pick the first edge $e \in L$ (this is the edge with the lowest cost).
4: If the graph $(N_0, T \cup \{e\})$ has a cycle, remove $e$ from $L$, set $b^k = b^{k-1}$,
   $ERO^k(N, 0, w) = ERO^{k-1}(N, 0, w)$ and go to step 3.
5: If the graph $(N_0, T \cup \{e\})$ has no cycle, remove $e$ from $L$, add $e$ to $T$ and set
   $ERO^k(N, 0, w) = ERO^{k-1}(N, 0, w) + (b^k - b^{k-1}) \cdot w(e)$.
6: If all vertices are connected to the source, set $ERO(N, 0, w) = ERO^k(N, 0, w)$.
   If not, go back to step 3.
```

Note that the change in the connection vectors in the $(k-1)$-th and the k-th step of the algorithm $(b^{k-1} - b^k)$ represents the fraction of the cost of the edge constructed in step k of the algorithm each player has to pay.

Example 3.3.2. *Consider the mcst problem $(N, 0, w)$ represented in figure 3.1.1a. As seen in Example 3.1.3, Kruskal's algorithm first constructs edge $\{2, 3\}$, then constructs edge $\{1, 3\}$, then considers edge $\{1, 2\}$ but removes this edge from the list since it creates a cycle with the edges $\{1, 3\}$ and $\{2, 3\}$ and then constructs the edge $\{0, 2\}$. Since these edges together form a spanning tree in (N_0, E_{N_0}), Kruskal's algorithm terminates.*

In table 3.3.1, the edge sets T^k considered and possibly formed by Kruskal's algorithm and the corresponding connection vectors are given. In figure 3.3.2 Kruskal's algorithm is depicted step for step and the amount each agent has to pay for constructing the edge in that particular step is given. Here the dashed edges are the edges that are allowed to construct and the thick edges represent the edges the agents chose to construct.

The ERO rule starts with $b^0 = (1, 1, 1)$ and $ERO^0(N, 0, w) = (0, 0, 0)$. Since in the first step, edge $\{2, 3\}$ is constructed we have $b^1 = (1, \frac{1}{2}, \frac{1}{2})$. By the introduction *of edge $\{2,3\}$, agent 1 is still not connected to the other agents or the source and therefore b_1^1 remains 1. Since agent 2 and agent 3 are connected to each other by the edge $\{2,3\}$, and both agents are not yet connected to the source, we have $b_2^1 = b_3^1 = \frac{1}{2}$. Because $(b^0 - b^1) = (0, \frac{1}{2}, \frac{1}{2})$, agent 2 and agent 3 both pay half of the cost of the edge $\{2,3\}$ and agent 1 does not have to pay a fraction of the cost of $\{2,3\}$ at all. Also we have*

$$\begin{aligned} ERO^1(N,0,w) &= ERO^0(N,0,w) + (b^0 - b^1) \cdot w(\{2,3\}) \\ &= (0,0,0) + (0, \tfrac{1}{2}, \tfrac{1}{2}) \cdot 3 \\ &= (0, 1\tfrac{1}{2}, 1\tfrac{1}{2}). \end{aligned}$$

In the second step, edge $\{1,3\}$ is constructed. Since now agent 1, 2 and 3 are connected to each other but are not connected to the source yet, we have $b^2 = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. Because $(b^1 - b^2) = (\frac{2}{3}, \frac{1}{6}, \frac{1}{6})$, agent 1 pays $\frac{2}{3}$ of the cost of edge $\{1,3\}$ and agent 2 and 3 both pay $\frac{1}{6}$ of the cost of edge $\{1,3\}$. Also we have

$$\begin{aligned} ERO^2(N,0,w) &= ERO^1(N,0,w) + (b^1 - b^2) \cdot w(\{1,3\}) \\ &= (0, 1\tfrac{1}{2}, 1\tfrac{1}{2}) + (\tfrac{2}{3}, \tfrac{1}{6}, \tfrac{1}{6}) \cdot 4 \\ &= (2\tfrac{2}{3}, 2\tfrac{1}{6}, 2\tfrac{1}{6}). \end{aligned}$$

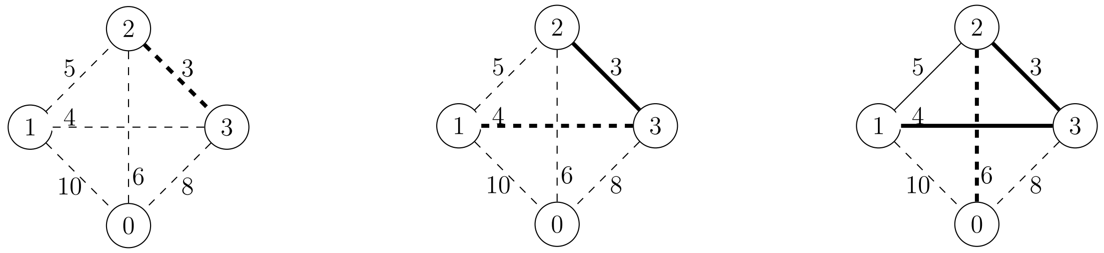
In the third step edge $\{1,2\}$ is considered, but since this edge creates a cycle with the edges $\{1,3\}$ and $\{2,3\}$ it is deleted from the list and we have $b^3 = b^2$ and $ERO^3(N,0,w) = ERO^2(N,0,w)$.

Finally edge $\{0,2\}$ is constructed. We have $b^4 = (0,0,0)$ since now all agents are connected to the source. Because $(b^3 - b^4) = b^2 = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, all agents pay $\frac{1}{3}$ of the cost of edge $\{0,2\}$. Also we have

$$\begin{aligned} ERO(N,0,w) &= ERO^4(N,0,w) \\ &= ERO^3(N,0,w) + (b^3 - b^4) \cdot w(\{0,2\}) \\ &= (2\tfrac{2}{3}, 2\tfrac{1}{6}, 2\tfrac{1}{6}) + (\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3}) \cdot 6 \\ &= (4\tfrac{2}{3}, 4\tfrac{1}{6}, 4\tfrac{1}{6}). \end{aligned}$$

$T^{\sigma,k}$	$b^{\sigma,k}$
$\emptyset$	$(1,1,1)^T$
$\{\{2,3\}\}$	$(1,\frac{1}{2},\frac{1}{2})^T$
$\{\{2,3\},\{1,3\}\}$	$(\frac{1}{3},\frac{1}{3},\frac{1}{3})^T$
$\{\{2,3\},\{1,3\},\{1,2\}\}$	$(\frac{1}{3},\frac{1}{3},\frac{1}{3})^T$
$\{\{2,3\},\{1,3\},\{1,2\},\{0,2\}\}$	$(0,0,0)^T$

Table 3.3.1: Edge sets formed by Kruskal's algorithm and the corresponding connection vectors of the mcst problem $(N, 0, w)$.



$2 \rightarrow (1-\frac{1}{2}) \cdot 3 = 1\frac{1}{2}$
$3 \rightarrow (1-\frac{1}{2}) \cdot 3 = 1\frac{1}{2}$

$1 \rightarrow (1-\frac{1}{3}) \cdot 4 = 2\frac{2}{3}$
$2 \rightarrow (\frac{1}{2}-\frac{1}{3}) \cdot 4 = \frac{2}{3}$
$3 \rightarrow (\frac{1}{2}-\frac{1}{3}) \cdot 4 = \frac{2}{3}$

$1 \rightarrow (\frac{1}{3}-0) \cdot 6 = 2$
$2 \rightarrow (\frac{1}{3}-0) \cdot 6 = 2$
$3 \rightarrow (\frac{1}{3}-0) \cdot 6 = 2$

Figure 3.3.2: Kruskal's algorithm step for step with the corresponding cost each agent has to pay for the constructed edge according to the *ERO* rule.

3.3.3 Vertex oriented construct and charge procedure

The voccp relies on the vertex oriented construct procedure and computes an allocation as follows: let $\sigma \in \Pi(N)$ be the permutation in which the agents are ordered. For every edge e constructed in the k-th step of the vertex oriented construt procedure, allocate the weight of that edge to agent $\sigma(k)$ – the agent constructing edge e in the k-th step of the algorithm. The voccp is formally presented in algorithm 5.

Notice that the only difference between the vertex oriented construct procedure (Algorithm 3) and the voccp (Algorithm 6) is the extra allocation rule in step 3 of the voccp.

Example 3.3.3. *In Example 3.1.2, we saw that when we pick the order $\sigma = (123)$ and consider the mcst problem presented in figure 3.1.1a, we get the mcst with the set*

Algorithm 6 Vertex oriented construct and charge procedure

Input: $|N|$ vertices, source 0, weight function $w : E_{N_0} \to \mathbb{R}_+$.
Output: Minimum cost spanning tree (N_0, T_σ) that connects all vertices in N to 0 and a cost allocation vector $v^\sigma(N, 0, w)$.

```
1: Pick $\sigma \in \Pi(N)$.
2: Initialize $T_\sigma^0 = \emptyset$.
3: From $k = 1$ to $k = |N|$, find an edge $e \in E_{N_0}$ of minimal cost that connects
   an agent in the component of $(N_0, T_\sigma^{k-1})$ which contains $\sigma(k)$, to an agent in a
   component of $(N_0, T_\sigma^{k-1})$ which does not contain agent $\sigma(k)$, set $T_\sigma^k = T_\sigma^{k-1} \cup \{e\}$
   and assign cost $w(e)$ to agent $\sigma(k)$, i.e., $v_{\sigma(k)}^\sigma(N, 0, w) = w(e)$.
4: Set $T_\sigma = T_\sigma^{|N|}$ and return $(N_0, T_\sigma)$ and $v^\sigma(N, 0, w)$.
```

of edges $\{\{1,3\},\{2,3\},\{0,2\}\}$ *where first edge* $\{1,3\}$ *is constructed by the vertex oriented construct procedure, then edge* $\{2,3\}$ *and finally edge* $\{0,2\}$. *The costs of these edges are respectively* 4, 3 *and* 6. *We therefore have* $v_1^\sigma(N,0,w) = v_{\sigma(1)}^\sigma(N,0,w) = 4$, $v_2^\sigma = v_{\sigma(2)}^\sigma(N,0,w) = 3$ *and* $v_3^\sigma(N,0,w) = v_{\sigma(3)}^\sigma = 6$. *This gives us the allocation* $v^\sigma(N,0,w) = (4,3,6)$. *See also figure 3.3.3.*

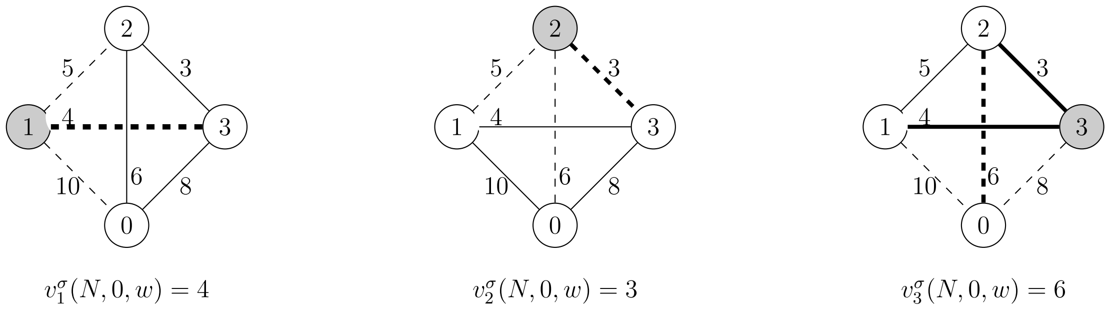


Figure 3.3.3: The voccp step for step with the corresponding cost allocation.

Notice that agent 3 constructs an edge in the third step of figure 3.3.3 that she is not incident to. This is not the case when the Bird allocation or the ERO allocation based on Kruskal's algorithm is used.

Because the allocation provided by the voccp depends on a particular ordering, this allocation is not really fair due to the fact that the first agent in the order has the right to construct an edge first, then the second and so on. To achieve fairness, the average of these allocations over all orderings of agents can be taken. It turns out this average is equal to $ERO(N, 0, w)$. A proof is given by Çiftçi (2009).

Theorem 3.3.4. *For $w \in \mathcal{W}^N$, we have*

$$ERO(N, 0, w) = \sum_{\sigma \in \Pi(N)} \frac{v^\sigma(N, 0, w)}{n!}.$$

Example 3.3.5. *The possible orders of agents of the mcst problem $(N, 0, w)$ presented in figure 3.1.1a and the corresponding edgeset and allocation vectors are given in table 3.3.2.*

σ	E	$v^\sigma(N, 0, w)$
(123)	$\{\{1,3\},\{2,3\},\{0,2\}\}$	$(4,3,6)$
(132)	$\{\{1,3\},\{2,3\},\{0,2\}\}$	$(4,6,3)$
(213)	$\{\{2,3\},\{1,3\},\{0,2\}\}$	$(4,3,6)$
(231)	$\{\{2,3\},\{0,2\},\{1,3\}\}$	$(6,3,4)$
(312)	$\{\{2,3\},\{1,3\},\{0,2\}\}$	$(4,6,3)$
(321)	$\{\{2,3\},\{2,0\},\{1,3\}\}$	$(6,4,3)$

Table 3.3.2: The possible orders of agents of the mcst problem $(N, 0, w)$ and the corresponding edgesets and allocation vectors.

Taking the average of the cost allocations over all possible orders σ gives

$$\begin{aligned} ERO(N, 0, w) = ((4, 3, 6) + (4, 6, 3) + (4, 3, 6) + (6, 3, 4) + (4, 6, 3) \\ + (6, 4, 3))/6 = (4\tfrac{2}{3}, 4\tfrac{1}{6}, 4\tfrac{1}{6}). \end{aligned}$$

Çiftçi (2009) proved that for every order $\sigma \in \Pi(N)$, the voccp allocation is exactly the marginal vector of the optimistic mcst game. In order to prove this, Lemma 3.3.6 is used.

Lemma 3.3.6. *Let $(N, 0, w)$ be an mcst problem. Let $\sigma \in \Pi(N)$, $k \in \{1, \dots, n\}$ and $\sigma(k) = i$. Denote the component of (N_0, T_σ^{k-1}) which contains agent i by C_i. Then the following statements are true*

(i) The source is not contained in $V(C_i)$.

(ii) Every agent in $V(C_i)$ other than agent i has already constructed an edge in a previous step of the algorithm.

Proof.

(i) Suppose the source is contained in $V(C_i)$, i.e., $0 \in V(C_i)$. Let $|V(C_i)| = r$ and $|E(C_i)| = q$. Because C_i is a tree, we have $r = q + 1$. Therefore, all agents in $V(C_i)$ must have constructed an edge. But i did not construct an edge yet, which is a contradiction.

(ii) Suppose there exists an agent $j \in V(C_i)$ that did not construct an edge yet. C_i is a tree so $r = q + 1$ where again $|V(C_i)| = r$ and $|E(C_i)| = q$. Therefore all agents except for one agent must have constructed an edge. This must be agent i per definition. So we have a contradiction.

□

We can now proof the following theorem

Theorem 3.3.7. *Let $(N, 0, w)$ be an mcst problem. Then for every $\sigma \in \Pi(N)$ we have*

$$v^\sigma(N, 0, w) = m^\sigma(c_O)$$

Proof. Let $\sigma \in \Pi(N)$. Let $S_k = \{\sigma(1), \ldots, \sigma(k)\}$ for every $k \in \{1, \ldots n\}$. Pick $k \in \{1, \ldots, n\}$. By Lemma 3.3.6, every component of (N_0, T_σ^k) contains exactly one vertex in $N \setminus S_k$. So every edge $e \in T_\sigma^k$ has at least one endpoint that belongs to S_k. Let $\{u, v\}$ be an edge in T_σ^k with $u \in S_k$ and $v \in (N \setminus S_k)_0$. Because $v \notin S_k$ we know that $\{u, v\}$ is the cheapest edge connecting $u \in S_k$ with a vertex v in $(N \setminus S)_0$. In other words, $w(\{u, v\}) = \min_{j \in (N \setminus S_k)_0} w(\{u, j\}) = w_{S_k}(\{u, 0\})$. This means that the cost of the edge $\{u, v\}$ is equal to the cost of the edge $\{u, 0\}$ in the mcst problem $(S_k, 0, w_{S_k})$. We can now construct an mcst in $(S_k, 0, w_{S_k})$ as follows: For every edge $\{u, v\}$ in T_σ^k, if both u and v are in S_k, then construct $\{u, v\}$ in $(S_k, 0, w_{S_k})$, if not, so if $u \in S_k$ and $v \notin S_k$ construct $\{u, 0\}$ in $(S_k, 0, w_{S_k})$. Let T be the set of edges that we just constructed. Because (N_0, T_σ^k) does not contain a cycle and every component of (N_0, T_σ^k) contains exactly one vertex not in S_k, T does not contain a cycle. Now, because T has k edges, T is a spanning tree in $(S_k, 0, w_{S_k})$. Because T_σ^k is constructed using the voccp and T uses all edges of T_σ^k, T is an mcst in $(S_k, 0, w_{S_k})$. We therefore have $c_O(S_k) = w(T) = w(T_\sigma^k)$ and

$$\begin{aligned} m^\sigma(c_O)_{\sigma(k)} &= c_O(S_k) - c_O(S_{k-1}) \\ &= w(T_\sigma^k) - w(T_\sigma^{k-1}) \\ &= v^\sigma_{\sigma(k)}(N, 0, w) \end{aligned}$$

where the last equation comes from the fact that $v^{\sigma}_{\sigma(k)}(N,0,w) = w(e)$ for e such that $T^k_\sigma = T^{k-1}_\sigma \cup \{e\}$.

Because this holds for every $k \in \{1,\dots,n\}$ we have $m^\sigma(c_O) = v^\sigma(N,0,w)$ (Çiftçi, 2009). □

From Theorem 3.3.7 it follows that ERO equals the Shapley value of the optimistic mcst game and the irreducible mcst game.

In the following we show that every voccp allocation is an element of the core.

Theorem 3.3.8. *Let $(N,0,w)$ be an mcst problem and let (N,c) be the corresponding mcst game. Every voccp allocation is an element of Core(c).*

Proof. For every order $\sigma \in \Pi(N)$ we have

$$v^\sigma(N,0,w) = m^\sigma(c_O) = m^{\sigma^{-1}}(c_I)$$

where the first equality follows from Theorem 3.3.7 and the second equality follows from Theorem 3.2.2. Therefore, for every order $\sigma \in \Pi(N)$ we have that the voccp allocation equals the marginal vector with respect to σ^{-1} of the irreducible mcst game. Because the irreducible core is the convex hull of it's marginal vectors and the irreducible core is a subset of the core by Theorem 3.2.6, every voccp allocation is an element of Core(c). □

With Theorem 3.3.8, together with the fact that every mcst can be constructed by the vertex oriented construct procedure, we can prove the following theorem.

Theorem 3.3.9. *Every mcst game has a non-empty core.*

Proof. Every mcst can be constructed by the vertex oriented construct procedure and every corresponding voccp allocation is in the core by Theorem 3.3.8. So every mcst game has a core element, therefore the core of an mcst game is non-empty. □

In this way, a complete proof that for every order of agents σ, the voccp allocation is an element of the core, is given.

Example 3.3.10. *The core corresponding to the mcst game (N,c) presented in table 3.2.1, together with the Bird- the voccp- and the ERO allocations is depicted in figure 3.3.4. We have $v^{(123)}(N,0,w) = v^{(213)}(N,0,w) = (4,3,6)$, $B(N,0,w) = v^{(132)}(N,0,w) = v^{(312)}(N,0,w) = (4,6,3)$, $v^{(231)}(N,0,w) = v^{(321)}(N,0,w) = (6,3,4)$ and $ERO(N,0,w) = (4\frac{2}{3}, 4\frac{1}{6}, 4\frac{1}{6})$. In figure 3.3.4, it can be seen that every voccp allocation is an element of the core of (N,c).*

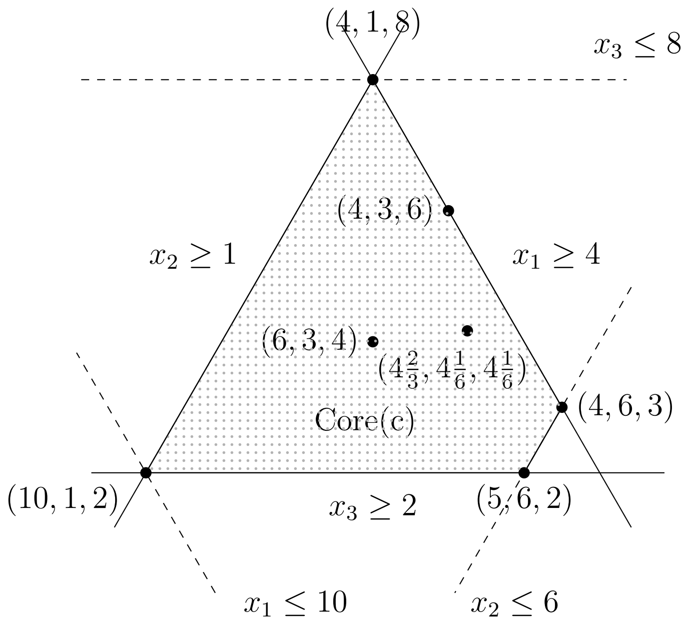


Figure 3.3.4: The core of the mcst game (N, c).

Chapter 4

Minimum Cost Spanning Tree Problems and Games with Time

This chapter explores the mcst problem in which edges can be constructed at two points in time and in which agents not only have to incur the cost of the edges but may also have to incur an additional cost. This problem is called the *Minimum Cost Spanning Tree Problem with Time (mcstt problem)* and can also be divided into two subproblems. The first problem is to construct a network of minimum cost that connects all agents to the source, either through a direct or indirect connection, in which the agents can connect to the source at two points in time. Once such an optimal network is found, the second problem is to divide the cost of this network among the agents in a fair way.

In section 4.1 the shape of an optimal network in the mcstt problem is explored. Section 4.2 provides an algorithm for the mcstt problem in case we know which agents connect to the source at which point in time. In section 4.3 we define the mcstt game and explore its properties. Section 4.4 tries to find a fair cost allocation method in order to answer the question how to divide the cost of the optimal network among the agents in a fair way. Finding a cost allocation method for which the allocation is an element of the core is harder than initially thought and it is not even clear if every mcstt game has a nonempty core. However, in section 4.4 we show that for a 3-agent mcstt game, the core is always nonempty.

4.1 Optimization: Constructing a network of minimum cost

Let $N = \{1, \ldots, n\}$ be a set of agents, let 0 represent the source and let $N_0 = N \cup \{0\}$, just as in section 3.1. There are two points in time at which the edges of the graph (N_0, E_{N_0}) can be constructed; at $t = 1$ and at $t = 2$. The costs of the edges at $t = 1$ are higher than the costs of the edges at $t = 2$. So, let $w_1 : E_{N_0} \to \mathbb{R}_+$ be the weight function that assigns a cost to every edge in the graph (N_0, E_{N_0}) if it is constructed at $t = 1$ and let $w_2 : E_{N_0} \to \mathbb{R}_+$ be the weight function that assigns a cost to every edge in (N_0, E_{N_0}) if it is constructed at $t = 2$. So $w_1(e)$ and $w_2(e)$ are the costs of constructing edge $e \in E_{N_0}$ at $t = 1$ and $t = 2$ respectively and we have $w_1(e) \geq w_2(e)$ for all edges $e \in E_{N_0}$. Furthermore, let $\alpha \in \mathbb{R}_+^N$ be a vector such that α_i represents the additional cost of agent i, i.e. α_i is the additional cost agent i makes to connect to the source at $t = 2$.

An mcstt problem is represented by the 5-tuple $(N, 0, w_1, w_2, \alpha)$ and can be depicted by two graphs: (N_0, E_{N_0}) with the weight function w_1 and (N_0, E_{N_0}) with the weight function w_2 and the additional cost vector α, see Example 4.1.1.

Example 4.1.1. *Suppose there are three houses in a village which need to be connected to a common water supply through pipelines. The houses can connect to the water supply, directly or through other houses, at $t = 1$ or at $t = 2$. If some villagers decide to construct pipelines in order to connect their house to the water supply at $t = 1$, the cost of the pipelines increases. This can be a result of hiring workers also in the night to construct the pipelines in a faster time span. However, if they decide to connect to the water supply at $t = 2$, they incur an additional cost, which can represent for example the extra cost they make to obtain water in a different way than through their private wells. Consider figure 4.1.1. The figure on the left shows the cost of constructing these connections between the houses and the water supply at $t = 1$. The figure on the right shows the cost of constructing these connections between the houses and the water supply at $t = 2$. Note that the cost of the edges at $t = 1$ are higher than the costs of the edges at $t = 2$. At $t = 2$ every node has an associated additional cost to it. Agent 1 has an additional cost of* 3*, agent 2 has an additional cost of* 9 *and agent 3 has an additional cost of* 2*. This means that if for example only agent* 1 *is connected to the source at $t = 2$ using edge $\{1, 2\}$, then the cost of this edge is $9 + 3 = 12$ instead of* 9*, while if only agent* 2 *is connected to the source at $t = 2$ using edge $\{1, 2\}$, then the cost of this edge is $9 + 9 = 18$ instead of* 9*.*

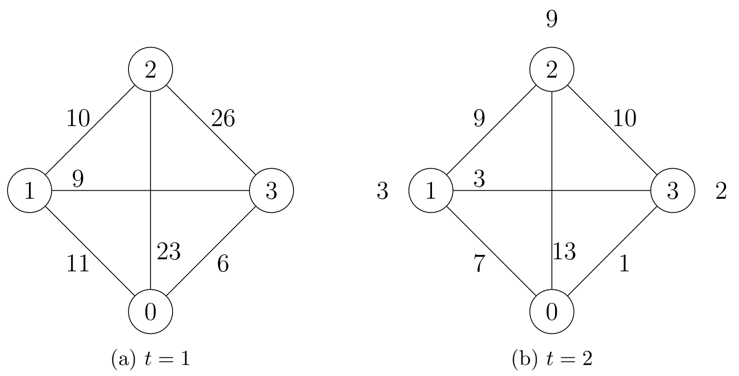


Figure 4.1.1: An mcstt problem.

Note that we can also represent the mcstt problem by two mcst problems: the mcst problem $(N, 0, w_1)$ at $t = 1$ and the mcst problem $(N, 0, w_2)$ together with the additional costs α at $t = 2$.

In order to connect all agents to the source, either at $t = 1$ or at $t = 2$, at minimum cost, we may have to construct all edges at $t = 1$, some edges at $t = 1$ and some edges at $t = 2$ or all edges at $t = 2$. Let us denote the set of edges constructed at $t = 1$ and $t = 2$ by T_1 respectively T_2 and let us denote the union of T_1 and T_2 by the edgeset T. So $T_1 \subseteq E_{N_0}$ for which the cost of an edge $e \in T_1$ equals $w_1(e)$ and $T_2 \subseteq E_{N_0}$ for which the cost of an edge $e \in T_2$ equals $w_2(e)$. First, note that since there is no use in constructing an edge at $t = 2$ that is already constructed at $t = 1$ we have $T_1 \cap T_2 = \emptyset$. Secondly, note that for edgesets T_1 and T_2 such that all agents are connected to the source at minimum cost, T_1 and T_2 do not have to be unique, since there can be multiple ways to connect the agents to the source at minimum cost. The cost of T, denoted by $w(T)$ is given by

$$w(T) = w(T_1 \cup T_2) = \sum_{e \in T_1} w_1(e) + \sum_{e \in T_2} w_2(e).$$

We can denote a network of minimum cost in the mcstt problem $(N, 0, w_1, w_2, \alpha)$ by a graph (N_0, T) in which $T = T_1 \cup T_2$ such that all agents are connected to the source at minimum cost by either edges in T_1 or edges in T_2. Note that, in this case, it is important to know at which time the edges in T are constructed. Therefore, in addition to (N_0, T), T_1 and T_2 have to be given. For simplicity, in the remainder of this thesis, we often only give (N_0, T) assuming T_1 and T_2 are known.

Because all agents need to be connected to the source, the network of minimum cost (N_0, T) has to be a connected graph. Now, $N(T_1)$ is the set of agents that are incident to the edgeset T_1. Other than the mcst problem, the mcstt problem does not only have costs on the edges but also deals with additional costs on the agents that connect to the source at $t = 2$. Therefore, a network of minimum cost (N_0, T) in the mcstt problem $(N, 0, w_1, w_2, \alpha)$ has a cost equal to the cost of the edge set T plus the additional costs of the agents that connect to the source at $t = 2$. The cost of this optimal network is denoted by $m(N, 0, w_1, w_2, \alpha)$. In general, we have

$$m(N, 0, w_1, w_2, \alpha) = \min\Big\{ \sum_{e \in T_1} w_1(e) + \sum_{e \in T_2} w_2(e) + \sum_{i \in N \setminus N(T_1)} \alpha_i \mid T_1, T_2 \subset E_{N_0},$$
$$T = T_1 \cup T_2, T_1 \cap T_2 = \emptyset \text{ and } (N_0, T) \text{ is connected} \Big\}.$$

Lemma 4.1.2 states that there always exists an optimal network (N_0, T) of the mcstt problem $(N, 0, w_1, w_2, \alpha)$ such that $(N(T_1)_0, T_1)$ is connected, meaning the edge set T_1 connects all agents $N(T_1)$ to the source.

Lemma 4.1.2. *Let $(N, 0, w_1, w_2, \alpha)$ be an mcstt problem. There exists a network of minimum cost (N_0, T) in $(N, 0, w_1, w_2, \alpha)$ where $T = T_1 \cup T_2$ and $T_1 \cap T_2 = \emptyset$ such that $(N(T_1)_0, T_1)$ is connected.*

Proof. Let (N_0, T) be an arbititrary network of minimum cost in $(N, 0, w_1, w_2, \alpha)$ and suppose $(N(T_1)_0, T_1)$ is not connected. Then there exists an edge $e = \{i, j\}$ in T_1 that is not an element of a path containing the source. So at least two agents, in this case agent i and j, are not connected to the source at $t = 1$. This means that those agents are connected to the source at $t = 2$ and that there are additional costs α_i and α_j. But then, because $w_2(e) \leq w_1(e)$ for all $e \in E_{N_0}$, it is cheaper or at least as cheap to construct edge e at $t = 2$. When $w_2(e) < w_1(e)$ this is a contradiction to the fact that $m(N, 0, w_1, w_2, \alpha)$ is minimal. So all $e \in T_1$ such that $w_2(e) < w_1(e)$ are an element of a path containing the source. When $w_1(e) = w_2(e)$, it does not matter if edge e is constructed at $t = 1$ or $t = 2$, so for all edges e in T_1 such that $w_1(e) = w_2(e)$ we can also construct edge e at $t = 2$, i.e., we delete all such edges e from T_1 and add all those edges to T_2. We denote the new edge set at $t = 1$ and $t = 2$ by T_1' and T_2' respectively. Then $(N_0, T_1' \cup T_2')$ is still a network of minimum cost and now $(N(T_1')_0, T_1')$ is connected. □

Furthermore, there always exists a network of minimum cost (N_0, T) such that $(N(T_1)_0, T_1)$ is a spanning tree in $(N(T_1)_0, E_{N(T_1)_0})$. This result is stated in Theorem 4.1.3.

Theorem 4.1.3. *Let $(N, 0, w_1, w_2, \alpha)$ be an mcstt problem. There exists a network of minimum cost (N_0, T) in $(N, 0, w_1, w_2, \alpha)$ where $T = T_1 \cup T_2$, $T_1 \cap T_2 = \emptyset$ and $(N(T_1)_0, T_1)$ is a spanning tree in $(N(T_1)_0, E_{N(T_1)_0})$.*

Proof. We need to proof that there exists a network of minimum cost (N_0, T) in $(N, 0, w_1, w_2, \alpha)$ such that $(N(T_1)_0, T_1)$ is connected and T_1 contains no cycles. With Lemma 4.1.2 we know that there exists a network of minimum cost such that $(N(T_1)_0, T_1)$ is connected. So suppose we have an network of minimum cost where $(N(T_1)_0, T_1)$ is connected but T_1 contains a cycle C. Because the weight function w_1 is nonnegative, we can distinguish two cases: $w_1(e) > 0$ for at least one edge $e \in C$ and $w_1(e) = 0$ for all edges $e \in C$.

Suppose that $w_1(e) > 0$ for at least one edge $e \in C$. Removing this edge, results in a lesser cost of $m(N, 0, w_1, w_2, \alpha)$, which is a contradiction. So T_1 does not contain any cycle C if $w_1(e) > 0$ for at least one edge $e \in C$.

Suppose now that $w_1(e) = 0$ for all edges $e \in C$. If we remove an arbitrary edge $e \in C$ then $(N(T_1 - e)_0, T_1 - e)$ still is a connected graph, contains no cycle and $(N_0, (T_1 - e) \cup T_2)$ still is a network of minimum cost in $(N, 0, w_1, w_2, \alpha)$. □

Because of Theorem 4.1.3 we can put extra constraints on T_1 and T_2 such that

$$m(N, 0, w_1, w_2, \alpha) = \min\Big\{ \sum_{e \in T_1} w_1(e) + \sum_{e \in T_2} w_2(e) + \sum_{i \in N \setminus N(T_1)} \alpha_i \mid T_1, T_2 \subset E_{N_0}, \\ T = T_1 \cup T_2, T_1 \cap T_2 = \emptyset, (N(T_1)_0, T_1) \text{ is a tree and } (N_0, T) \text{ is connected} \Big\}.$$

A network of minimum cost (N_0, T) where $T = T_1 \cup T_2, T_1, T_2 \subset E_{N_0}, T_1 \cap T_2 = \emptyset$ and $(N(T_1)_0, T_1)$ and (N_0, T) are trees is called a *minimum cost spanning tree with time (mcstt).* Note that there always exists an optimal network (N_0, T) in $(N, 0, w_1, w_2, \alpha)$ such that (N_0, T) is a spanning tree in (N_0, E_{N_0}) for the same reason as in the proof of Theorem 4.1.3. The cost of an mcstt in $(N, 0, w_1, w_2, \alpha)$ therefore equals

$$m(N, 0, w_1, w_2, \alpha) = \min\Big\{ \sum_{e \in T_1} w_1(e) + \sum_{e \in T_2} w_2(e) + \sum_{i \in N \setminus N(T_1)} \alpha_i \mid T_1, T_2 \subset E_{N_0}, \\ T = T_1 \cup T_2, T_1 \cap T_2 = \emptyset, (N(T_1)_0, T_1) \text{ and } (N_0, T) \text{ are trees} \Big\}.$$

Another way to write $m(N, 0, w_1, w_2, \alpha)$ is by using the definitions of the pessimistic and optimistic mcst games.

Theorem 4.1.4. *Let $(N, 0, w_1, w_2, \alpha)$ be an mcstt problem. Let (N, c) be the mcst game associated with the mcst problem $(N, 0, w_1)$ and let (N, c_O) be the optimistic mcst game associated with the mcst problem $(N, 0, w_2)$. Then*

$$m(N, 0, w_1, w_2, \alpha) = \min_{N(T_1) \subseteq N} \{c(N(T_1)) + c_O(N \setminus N(T_1)) + \sum_{i \in N \setminus N(T_1)} \alpha_i\}.$$

Proof. We have

$$\begin{aligned}
m(N,0,w_1,w_2,\alpha) = \min\Big\{ &\sum_{e\in T_1} w_1(e) + \sum_{e\in T_2} w_2(e) + \sum_{i\in N\setminus N(T_1)} \alpha_i \mid T_1,T_2 \subset E_{N_0}, \\
&T = T_1\cup T_2, T_1\cap T_2=\emptyset, (N(T_1)_0,T_1) \text{ and } (N_0,T) \text{ are trees}\Big\} \\
= \min\Big\{ &\sum_{e\in T_1} w_1(e) \mid T_1 \subseteq E_{(N(T_1)_0} \text{ and } ((N(T_1)_0,T_1) \text{ is a tree}\Big\} \\
+ \min\Big\{ &\sum_{e\in T_2} w_2(e) \mid T_1\cap T_2=\emptyset, (N_0, T_1\cup T_2) \text{ is a spanning} \\
&\text{tree in } (N_0,E_{N_0})\Big\} + \sum_{i\in N\setminus N(T_1)} \alpha_i \\
= \min_{N(T_1)\subseteq N} &\{c(N(T_1)) + c_O(N\setminus N(T_1)) + \sum_{i\in N\setminus N(T_1)} \alpha_i\}
\end{aligned}$$

□

This theorem shows that if we know which edges have to be constructed at which point in time, we also know which agents connect to the source at which point in time.

4.2 Algorithm

If we know which edges have to be constructed at which point in time, in the previous section we saw that we then know which agents need to connect to the source at which point in time. Conversely, if we know which agents connect to the source at which point in time, using any algorithm given in section 2.3, we can find which edges to construct at which point in time. So once we know which agents connect to the source at $t = 1$ and which agents connect to the source at $t = 2$, it is not that hard to provide an algorithm for the mcstt problem.

Let S_1 be the set of agents that connect to the source at $t = 1$ – and therefore the agents in $N \setminus S_1$ connect to the source at $t = 2$. Suppose we know S_1. To obtain an mcstt in the mcstt problem $(N, 0, w_1, w_2, \alpha)$ we can use a combination of any of the algorithms given in section 2.3 as follows: in the first step, use an algorithm given in section 2.3 on the mcst problem $(S_1, 0, w_1)$. The output of this algorithm will then be an mcst $((S_1)_0, T_1)$ in $(S_1, 0, w_1)$. In the second step, again use any of the algorithms given in section 2.3 on the agents in $N \setminus S_1$. Add the edges constructed by the algorithm in the second step to the edgeset T_2 to ultimately obtain an mcstt $(N_0, T_1 \cup T_2) = (N_0, T)$ in $(N, 0, w_1, w_2, \alpha)$. Algorithm 7 is one example of such a construct procedure, where in the first step Prim's algorithm is applied to the mcst problem $(S_1, 0, w_1)$ and in the second step the vertex oriented construct procedure is applied to the mcst problem $(N \setminus S_1, 0, w_2)$ with initialization $T_\sigma^0 = T_1$, where T_1 is the edgeset constructed by Prim's algorithm in the first step.

Theorem 4.2.1. *Let $(N, 0, w_1, w_2, \alpha)$ be an mcstt problem. Suppose S_1 is known. The graph (N_0, T_σ) constructed by Algorithm 7 is an mcstt in $(N, 0, w_1, w_2, \alpha)$.*

Proof. In step 1, Prim's algorithm finds a spanning tree of the graph $((S_1)_0, E_{(S_1)_0})$. In the vertex oriented construct procedure in step 2, for each $k \in \{1, \ldots, |N \setminus S_1|\}$, an edge is added to the graph (N_0, T_σ^{k-1}) such that this edge does not create a cycle with the previous added edges. Therefore, T_σ has n edges and (N_0, T_σ) does not contain a cycle. So (N_0, T_σ) is a spanning tree in $(N, 0, w_1, w_2, \alpha)$. The only thing left to proof is proving that (N_0, T_σ) has minimum cost.

The spanning tree constructed by Prim's algorithm in step 1 has cost $c(S_1)$ per definition of the mcst game and because of Theorem 3.1.3. The total cost allocated to the agents in step 2 of the algorithm is equal to

$$\sum_{k=1}^{|N \setminus S_1|} \Big(c_O(\{\sigma(1), \ldots, \sigma(k)\}) - c_O(\{\sigma(1), \ldots, \sigma(k-1)\}) \Big) + \sum_{i \in N \setminus S_1} \alpha_i$$

by Theorem 3.3.7. By writing the first sum out, the above equation equals

$$c_O(\{\sigma(1), \ldots, \sigma(|N \setminus S_1|)\} - c_O(\emptyset) + \sum_{i \in N \setminus S_1} \alpha_i$$

which in turn is equal to

$$c_O(N \setminus S_1) + \sum_{i \in N \setminus S_1} \alpha_i$$

Therefore, the cost of (N_0, T_σ) is equal to

$$c(S_1) + c_O(N \setminus S_1) + \sum_{i \in N \setminus S_1} \alpha_i$$

Because $S_1 = N(T_1)$, by theorem 4.1.4 (N_0, T_σ) is an mcstt in $(N, 0, w_1, w_2, \alpha)$. □

Algorithm 7 Combination of Prim's algorithm and the vertex oriented construct procedure

```
Input: $|N|$ vertices, $S_1 \subseteq N$, source 0, weight functions $w_1, w_2 : E_{N_0} \to \mathbb{R}_+$,
additional cost function $\alpha : N \to \mathbb{R}_+$.
Output: Minimum cost spanning tree with time $(N_0, T_\sigma)$ that connects all
vertices in $N$ to 0, $T_1$ (the edges constructed at $t = 1$) and $T_2$ (the edges
constructed at $t = 2$).

STEP 1:
1: Initialize $S = \{0\}, T_1 = \emptyset$.
2: Find the cheapest edge $e$ between a vertex $j$ in $S$ and a vertex $i$ in $(S_1 \cup \{0\}) \setminus S$
   such that the graph $(S_1 \cup \{0\}, T_1 \cup \{e\})$ does not contain a cycle.
3: Add vertex $i$ to the set $S$ and edge $e$ to the set $T_1$.
4: If $S \neq S_1 \cup \{0\}$, go back to step 2.
5: Else, all vertices in $S_1$ are connected to the source and the algorithm terminates.
   Return $T_1$.

STEP 2:
1: Pick $\sigma \in \Pi(N \setminus S_1)$.
2: Initialize $T_\sigma^0 = T_1$ and $T_2 = \emptyset$.
3: From $k = 1$ to $k = |N \setminus S_1|$, find an edge $e$ of minimal cost that connects an agent
   in the component of $(N_0, T_\sigma^{k-1})$ which contains $\sigma(k)$, to an agent in a component
   of $(N_0, T_\sigma^{k-1})$ which does not contain agent $\sigma(k)$, set $T_\sigma^k = T_\sigma^{k-1} + e$ and add $e$
   to $T_2$.
4: Set $T_\sigma = T_1 \cup T_2$ and return $(N_0, T_\sigma)$ and $T_2$.
```

4.3 Minimum cost spanning tree games with time

A *minimum cost spanning tree game with time (mcstt game)* is a game (N, c_t) that assigns to every coalition $S \in 2^N$ the cost of an mcstt of the mcstt problem $(S, 0, w_1, w_2, \alpha)$, i.e.,

$$c_t(S) = m(S, 0, w_1, w_2, \alpha)$$

So $c_t(S)$ is the minimal cost coalition S incurs such that all agents in S are connected to the source, either directly or via each other, either at $t = 1$ or at $t = 2$, leaving the agents in $N \setminus S$ out of consideration and therefore also all connections to agents in $N \setminus S$.

Example 4.3.1. *Reconsider the mcstt problem $(N, 0, w_1, w_2, \alpha)$ depicted in figure 4.1.1. We have the following mcstt game (N, c_t):*

S	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	N
$c(S)$	0	10	22	3	21	9	22	24

Table 4.3.1: The mcstt game corresponding to the mcstt problem $(N, 0, w_1, w_2, \alpha)$.

In contrast to the mcst problem, where for every coalition there is only one possible point in time to connect to the source, in the mcstt problem for every coalition there are several possibilities: every agent in the coalition connects to the source at $t = 1$, every agent in the coalition connects to the source at $t = 2$ or some agents in the coalition connect to the source at $t = 1$ while the other agents in the coalition connect to the source at $t = 2$. To see what one agent would do individually we have to look at 2 possibilities: either the agent constructs an edge directly to the source at $t = 1$ or the agent constructs an edge directly to the source at $t = 2$. To see which connections a coalition of size 2 constructs we would have to look at 2^2 possibilities. In general, for a coalition $S \in 2^N$ of size $|S|$ we have to look at $2^{|S|}$ possibilities.

To define the minimum cost of connecting a coalition S to the source, we can also look at the costs of all the possibilities in which the coalition S can connect to the source, and choose the one with minimum cost.

Let S_1 be the set of agents in the coalition S that connect to the source at $t = 1$ – and therefore the agents in $S \setminus S_1$ connect to the source at $t = 2$. Note that there are exactly $2^{|S|}$ possibilities in which agents can connect to the source at $t = 1$. The cost of connecting a coalition S to the source, given that the agents in S_1 connect to the source at $t = 1$ is denoted by $c(S \mid S_1)$ and is given by

$$c(S \mid S_1) = \min \Big\{ \sum_{e \in T_1} w_1(e) + \sum_{e \in T_2} w_2(e) + \sum_{i \in S \setminus S_1} \alpha_i \mid ((S_1)_0, T_1) \text{ is a tree }, T_1 \cap T_2 = \emptyset$$
$$(N_0, T_1 \cup T_2) \text{ is a tree spanning tree in } (N_0, E_{N_0}) \Big\}$$

Theorem 4.3.2. *Let $(N, 0, w_1, w_2, \alpha)$ be an mcstt problem. Let (N, c) be the mcst game associated with the mcst problem $(N, 0, w_1)$ and let (N, c_O) be the optimistic mcst game associated with the mcst problem $(N, 0, w_2)$ such that c_O assigns to every coalition $S \in 2^N$ the cost of an mcst in the mcst problem $(S, 0, w_S)$. Then*

$$c(S \mid S_1) = c(S_1) + c_O(S \setminus S_1) + \sum_{i \in S \setminus S_1} \alpha_i$$

Proof. We have

$$\begin{aligned}
c(S \mid S_1) &= \min\Big\{\sum_{e\in T_1} w_1(e) + \sum_{e\in T_2} w_2(e) + \sum_{i\in S\setminus S_1} \alpha_i \mid ((S_1)_0, T_1) \text{ is a tree }, T_1 \cap T_2 = \emptyset \\
&\quad (N_0, T_1 \cup T_2) \text{ is a tree spanning tree in } (N_0, E_{N_0})\Big\} \\
&= \min\Big\{\sum_{e\in T_1} w(e) \mid T_1 \subset (E_{S_1})_0 \text{ and } ((S_1)_0, T_1) \text{ is a tree}\Big\} + \min\Big\{\sum_{e\in T_2} w_2(e) \\
&\quad \mid T_1 \cap T_2 = \emptyset, (N_0, T_1 \cup T_2) \text{ is a spanning tree in } (N_0, E_{N_0})\Big\} + \sum_{i\in S\setminus S_1} \alpha_i \\
&= c(S_1) + c_O(S \setminus S_1) + \sum_{i\in S\setminus S_1} \alpha_i
\end{aligned}$$

□

Example 4.3.3. *Consider the mcstt problem $(N, 0, w_1, w_2, \alpha)$ given in figure 4.1.1. In table 4.3.2 all possibilities for the agents in coalition S to connect to the source at $t = 1$ are given with their corresponding cost. Note that only S_1 is given, because if we know which agents connect to the source at $t = 1$ we also know which agents connect to the source at $t = 2$. In this example, if we look at the coalition N, agent 1 and 2 construct an edge at $t = 1$ and agent 3 constructs an edge at $t = 2$ because this results in the lowest total cost of* 24.

The mcstt game can now also be given by the pair (N, c_t) where $N = \{1, \ldots n\}$ is a set of agents and $c_t : 2^N \to \mathbb{R}_+$ a cost function that assigns to every coalition S the lowest possible value of $c(S \mid S^1)$, i.e.

$$c_t(S) = \min_{S_1 \subseteq S}\{c(S \mid S_1)\}$$

Example 4.3.3 (continued)**.** *The mcstt game associated with the mcstt problem $(N, 0, w_1, w_2, \alpha)$ can be derived from table 4.3.2 where for every coalition S we choose the lowest value of $c(S \mid S_1)$. The corresponding mcstt game is given in table 4.3.3. Note that this table is the same as table 4.3.1*

S	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	N
$c_t(S)$	0	10	22	3	21	9	22	24

Table 4.3.3: The mcstt game associated with the mcstt problem $(N, 0, w_1, w_2, \alpha)$.

S	Possibilities
$\{1\}$	$\begin{array}{\|l\|l\|l\|}\hline S_1 & \emptyset & \{1\} \\ \hline c(\{1\} \mid S_1) & 10 & 11 \\ \hline\end{array}$
$\{2\}$	$\begin{array}{\|l\|l\|l\|}\hline S_1 & \emptyset & \{2\} \\ \hline c(\{2\} \mid S_1) & 22 & 23 \\ \hline\end{array}$
$\{3\}$	$\begin{array}{\|l\|l\|l\|}\hline S_1 & \emptyset & \{3\} \\ \hline c(\{3\} \mid S_1) & 3 & 6 \\ \hline\end{array}$
$\{1,2\}$	$\begin{array}{\|l\|l\|l\|l\|l\|}\hline S_1 & \emptyset & \{1\} & \{2\} & \{1,2\} \\ \hline c(\{1,2\} \mid S_1) & 28 & 29 & 33 & 21 \\ \hline\end{array}$
$\{1,3\}$	$\begin{array}{\|l\|l\|l\|l\|l\|}\hline S_1 & \emptyset & \{1\} & \{3\} & \{1,3\} \\ \hline c(\{1,3\} \mid S_1) & 9 & 14 & 16 & 15 \\ \hline\end{array}$
$\{2,3\}$	$\begin{array}{\|l\|l\|l\|l\|l\|}\hline S_1 & \emptyset & \{2\} & \{3\} & \{2,3\} \\ \hline c(\{2,3\} \mid S_1) & 22 & 26 & 27 & 29 \\ \hline\end{array}$
$\{1,2,3\}$	$\begin{array}{\|l\|l\|l\|l\|l\|l\|l\|l\|l\|}\hline S_1 & \emptyset & \{1\} & \{2\} & \{3\} & \{1,2\} & \{1,3\} & \{2,3\} & \{1,2,3\} \\ \hline c(N \mid S_1) & 27 & 32 & 32 & 30 & 24 & 33 & 35 & 25 \\ \hline\end{array}$

Table 4.3.2: The different possibilities for agents in all coalitions to connect to the source at $t = 1$ or $t = 2$ with their corresponding costs.

We thus have

$$c_t(S) = m(S, 0, w_1, w_2, \alpha) = \min_{S_1 \subseteq S}\{c(S \mid S_1)\} = \min_{S_1 \subseteq S}\left\{c(S_1) + c_O(S \setminus S_1) + \sum_{i \in S \setminus S_1} \alpha_i\right\}.$$

By writing the mcstt game in this form, it is easy to see that mcstt games are subadditive. This result is stated in Theorem 4.3.4

Theorem 4.3.4. *Let $(N, 0, w_1, w_2, \alpha)$ be an mcstt problem. Then, the corresponding mcstt game (N, c_t) is subadditive.*

Proof. Suppose we have two disjoint subsets S and T of N, i.e. $S, T \subseteq N$ and

$S \cap T = \emptyset$. Then we have

$$\begin{aligned}
c_t(S) + c_t(T) &= \min_{S_1 \subseteq S} c(S \mid S_1) + \min_{T_1 \subseteq T} c(T \mid T_1) \\
&= c(S \mid S_1^*) + c(T \mid T_1^*) \\
&= c(T_1^*) + c(S_1^*) + c_0(S \setminus S_1^*) + c_0(T \setminus T_1^*) + \sum_{i \in S \setminus S_1^*} \alpha_i + \sum_{i \in T \setminus T_1^*} \alpha(i) \\
&\geq c(S_1^* \cup T_1^*) + c_0((S \setminus S_1^*) \cup (T \setminus T_1^*)) + \sum_{i \in (S \setminus S_1^*) \cup (T \setminus T_1^*)} \alpha_i \\
&= c(S_1^* \cup T_1^*) + c_0((S \cup T) \setminus (S_1^* \cup T_1^*)) + \sum_{i \in (S \cup T) \setminus (S_1^* \cup T_1^*)} \alpha_i \\
&\geq \min_{R \subseteq S \cup T} c(S \cup T \mid R) \\
&= c_t(S \cup T).
\end{aligned}$$

where S_1^* and T_1^* are subsets of agents in the coalitions S respectively T that connect to the source at $t = 1$ such that $c(S \mid S_1)$ respectively $c(T \mid T_1)$ are minimal. Here, the first inequality comes from the fact that the games (N, c) and (N, c_0) are subadditive and the fourth equality comes from the fact that S and T are disjunct. □

For an mcstt problem $(N, 0, w_1, w_2, \alpha)$, when the additional costs of the agents are all higher than the reduced amount of the cost of the edges, all agents connect to the source at $t = 1$. This means we can consider only the mcst problem $(N, 0, w_1)$ and the mcst in $(N, 0, w_1)$ is equal to the mcstt in $(N, 0, w_1, w_2, \alpha)$. To see this, consider an mcst in $(N, 0, w_1)$. At least one agent is directly connected to the source. If the additional cost of the agents directly connected to the source is higher then the maximum reduced amount of the edges that those agents are incident to, they will construct an edge at $t = 1$, since there does not exists an agent depending on one of those agents with an additional cost less than the reduced amount of the edges. The same holds now for all agents that are incident to one of those agents that are directly connected to the source and so on. If the additional costs of the agents are all lower than the reduced amount of the cost of the edges, all agents connect to the source at $t = 2$. This means we can consider only the mcst problem $(N, 0, w_2)$ together with the additional costs α and the mcst in this problem is equal to the mcstt in $(N, 0, w_1, w_2, \alpha)$.

Because of the fact that for high enough additional costs, we can return to the initial mcst problem, mcstt games do not have to be monotone, nor do they have to be submodular, since mcst games are not always monotone nor are they always submodular. Even if the additional costs of all agents are not all higher or all lower than the reduced amount of the cost of the edges incident to the corresponding agents, an mcstt game can still be not monotone and not submodular. For example, the mcstt game given in table 4.3.1 is an mcstt game that is not monotone nor is it submodular.

4.4 In search of a fair cost allocation method

Because any combination of Prim's algorithm, Kruskal's algorithm and the vertex oriented construct procedure can be used to obtain an mcstt in the mcstt problem $(N, 0, w_1, w_2, \alpha)$, an obvious way to obtain a cost allocation vector would be using the corresponding combination of allocation rules. For example, in step one of Algorithm 7 we could use the associated Bird rule and in step 2 of Algorithm 7 we could use the associated voccp, see Algorithm 8.

However this method does not necessarily output a cost allocation that is an element of the core. To see this, any allocation rule explained in section 3.3 can be used for agents connecting to the source at $t = 1$, but the cost of a coalition $S \subseteq S_1$ in the mcst problem $(N, 0, w_1)$ does not need to be the same as the cost of that coalition, $c_t(S)$, in the mcstt problem $(N, 0, w_1, w_2, \alpha)$, see Example 4.4.1.

Example 4.4.1. *Consider the mcstt problem $(N, 0, w_1, w_2, \alpha)$ given in figure 4.1.1 and the corresponding mcstt game (N, c_t) given in table 4.3.1. The optimal way for N to connect to the source is for agents 1 and 2 to connect to the source at $t = 1$ and for agent 3 to connect to the source at $t = 2$, i.e., $S_1 = \{1, 2\}$. The first edge that is constructed by Algorithm 8 is $\{0, 1\}$. Because agent 1 gets connected to the source, the cost of constructing edge $\{0, 1\}$ at $t = 1$ is assigned to agent 1. Next, agent 2 gets connected to the source by constructing the edge $\{1, 2\}$ so agent 2 gets assigned the cost of building edge $\{1, 2\}$ at $t = 1$. Finally, agent 3 connects to the source at $t = 2$ by constructing edge $\{0, 3\}$ and therefore gets allocated the cost of constructing edge $\{0, 3\}$ at $t = 2$ plus the additional cost α_3. This results in the allocation vector $x = (11, 10, 3)$. As can be seen in table 4.3.1 we have $c_t(\{1\}) = 10 < 11 = x_1$. Hence the allocation vector x constructed by Algorithm 8 is not an element of the core.*

Instead of using the Bird rule in step 1 of Algorithm 8 and the voccp in step 2 of Algorithm 8, we could use any combination construct and charge procedures. However, for the mcstt problem depicted in figure 4.1.1 none of these combinations give an allocation vector that is an element of the core of the corresponding mcstt game.

Algorithm 8 Combination of the Bird rule and the voccp

```
**Input:** $|N|$ vertices, $S_1 \subseteq N$, source 0, weight functions $w_1, w_2 : E_{N_0} \rightarrow \mathbb{R}_+$,
additional cost function $\alpha : N \rightarrow \mathbb{R}_+$.
**Output:** Minimum cost spanning tree with time $(N_0, T_\sigma)$ that connects all
vertices in $N$ to 0,the associated edgesets $T_1$ and $T_2$ and a cost allocation vector $x^\sigma$.

STEP 1:
1: Initialize $S = \{0\}, T_1 = \emptyset$.
2: Find the cheapest edge $e$ between a vertex $j$ in $S$ and a vertex $i$ in $(S_1 \cup \{0\}) \setminus S$
   such that the graph $(S_1 \cup \{0\}, T_1 \cup \{e\})$ does not contain a cycle.
3: Add vertex $i$ to the set $S$ and edge $e$ to the set $T_1$. Assign the cost $w_1(e)$ to agent
   $i$, i.e. $x_i^\sigma = w_1(e)$.
4: If $S \neq S_1 \cup \{0\}$, go back to step 2.
5: Else, all vertices in $S_1$ are connected to the source and the algorithm terminates.
   Return $T_1$ and $x_i^\sigma$ for all $i \in S_1$.

STEP 2:
1: Pick $\sigma \in \Pi(N \setminus S_1)$.
2: Initialize $T_\sigma^0 = T_1$ and $T_2 = \emptyset$.
3: From $k = 1$ to $k = |N \setminus S_1|$, find an edge $e$ of minimal cost that connects an agent
   in the component of $(N_0, T_\sigma^{k-1})$ which contains $\sigma(k)$, to an agent in a component
   of $(N_0, T_\sigma^{k-1})$ which does not contain agent $\sigma(k)$, set $T_\sigma^k = T_\sigma^{k-1} + e$ and add $e$ to
   $T_2$. Assign cost $w_2(e) + \alpha_{\sigma(k)}$ to agent $\sigma(k)$, i.e. $x_i^\sigma = w_2(e) + \alpha_i$ where $i = \sigma(k)$.
4: Set $T_\sigma = T_1 \cup T_2$ and return $(N_0, T_\sigma)$ and $x^\sigma$.
```

Example 4.4.1 (continued)**.** *In table 4.4.1 all combinations of algorithms given in section 2.3 are given with their corresponding allocation. Because the ERO rule can rely on both Kruskal's algorithm as on the vertex oriented construct procedure, only the voccp is shown in table 4.4.1. If the Bird rule is used in step 1 and step 2, we get the allocation vector* $(11, 10, 3)$*. If the Bird rule is used in step 1 and the voccp in step 2 we also get the allocation vector* $(11, 10, 3)$*. If the voccp is used in step 1 and the Bird rule is used in step 2 we get either the allocation vector* $(10, 11, 3)$ *if* $\sigma = (1, 2)$ *or the allocation vector* $(11, 10, 3)$ *if* $\sigma = (2, 1)$*. Taking the average gives the allocation vector* $(10\frac{1}{2}, 10\frac{1}{2}, 3)$*. If the voccp is used in both steps, we also get the allocation vector* $(10\frac{1}{2}, 10\frac{1}{2}, 3)$*.*

A natural way to deal with this issue is to restrict the amount one of the allocation rules allocates in the first step in such a way that the allocation of a coalition $S \subseteq S_1$

Step 1	Step 2	x
Bird rule	Bird rule	$(11, 10, 3)$
Bird rule	Voccp	$(11, 10, 3)$
Voccp	Bird rule	$(10\frac{1}{2}, 10\frac{1}{2}, 3)$
Voccp	Voccp	$(10\frac{1}{2}, 10\frac{1}{2}, 3)$

Table 4.4.1: The allocation vectors corresponding to every combination of allocation rules used for agents who connect to the source at $t = 1$ and at $t = 2$.

is at most $c_t(S)$. Unfortunately, this still does not have to be an element of the core since a coalition that consists of some agents that connect to the source at $t = 1$ and some agents that connect to the source at $t = 2$ can use different edges such that the cost of this coalition is smaller than the sum of the costs allocated to those agents.

Example 4.4.1 (continued)**.** *As* $c_t(\{1\}) = 10$ *we know* x_1 *can be at most* 10*. As* $c_t(\{1, 2\}) = 21$ *we know* x_2 *can be at most* 11*. Because* $c_t(\{2\}) > 11$*, we could therefore consider the allocation vector* $x = (10, 11, 3)$*. However,* $x_1 + x_3 = 13 > 9 = c_t(\{1, 3\})$*. Therefore this allocation vector is not an element of the core of the corresponding mcstt game.*

As Example 4.4.1 shows, finding a fair allocation method for the mcstt problem, making use of either the Bird rule, the voccp, the ERO rule or a combination of those algorithms, does not work for the mcstt problem. We have tried a lot of different mcstt problems, also for mcstt problems with $|N| > 3$ and in none of the corresponding mcstt games we have found an empty core. This could indicate that all mcstt games have a non-empty core. However, we did not yet find an allocation method that always constructs an allocation vector that is an element of the core. Fortunately, for an mcstt problem in which $|N| = 3$, we know that the core of the corresponding mcstt game is nonempty.

Theorem 4.4.2. *Let* $(N, 0, w_1, w_2, \alpha)$ *be an mcstt problem and let* (N, c_t) *be the corresponding mcstt game. If* $|N| = 3$*, then the mcstt game has a nonempty core.*

Proof. The proof can be found in the Appendix 1. □

Example 4.4.2 (continued)**.** *The core of the mcstt game is defined by the following* *equations:*

$$\begin{aligned} x_1 + x_2 + x_3 &= 24, \\ x_1 &\leq 10, \\ x_2 &\leq 22, \\ x_3 &\leq 3, \\ x_1 + x_2 \leq 21 &\implies x_3 \geq 3, \\ x_1 + x_3 \leq 9 &\implies x_2 \geq 15, \\ x_2 + x_3 \leq 22 &\implies x_1 \geq 2. \end{aligned}$$

Because $x_3 \leq 3$ *and* $x_3 \geq 3$ *we have* $x_3 = 3$*. Therefore, we have* $Core(c) = conv\{(6, 15, 3), (2, 19, 3)\}$.

Chapter 5

Minimum Cost Spanning Tree Problems with Time with Constant Reduced Edges

The difficulty in finding an algorithm for the mcstt problem lies in knowing which agents connect to the source at $t = 1$. Also, finding a core allocation method for the mcstt problem appears not to be straight forward. In order to try to find an answer to these problems, this chapter explores a special class of mcstt problems in which the weight of the edges at $t = 1$ are all reduced at $t = 2$ by the same amount. This subproblem is called the *Minimum Cost Spanning Tree Problem with Time with Constant Reduced Edges (mcsttr problem)*.

This chapter is divided into 4 sections. In section 1 the mcsttr problem is introduced and some properties of the associated mcsttr games are given. Section 2 tries to find an answer to the question how to find which agents connect to the source at which point in time. Once we have answered this question, it is easy to find an mcstt in an mcsttr problem. Section 3 explores the shape of an mcstt in mcsttr problems. In section 4, it is shown that the cost allocation method described in the previous chapter also does not work for mcsttr problems. Furthermore, for a 3-agent mcsttr problem, a cost allocation method is given such that the allocation provided by this method is an element of the core of the corresponding mcsttr game.

5.1 Minimum cost spanning tree problems with time with constant reduced edges

The mcsttr problem is given by the 6-tuple $(N, 0, w_1, w_2, \alpha, r)$ in which $(N, 0, w_1, w_2, \alpha)$ is the mcstt problem and r is such that for every edge $e \in E_{N_0}$ we have

$$w_2(e) = w_1(e) - r$$

with r a constant real number such that $r \leq w_1(e)$ for every $e \in E_{N_0}$. This problem can also be depicted by the two graphs (N_0, E_{N_0}) with the weight function w_1 and (N_0, E_{N_0}) with the weight function w_2 and the additional cost vector α, see Example 5.1.1.

Example 5.1.1. *In figure 5.1.1 an mcsttr problem $(N, 0, w_1, w_2, \alpha, r)$ is given. The costs of the edges at $t = 1$ are reduced with an amount equal to $r = 10$ at $t = 2$, i.e., $w_2(e) = w_1(e) - r$.*

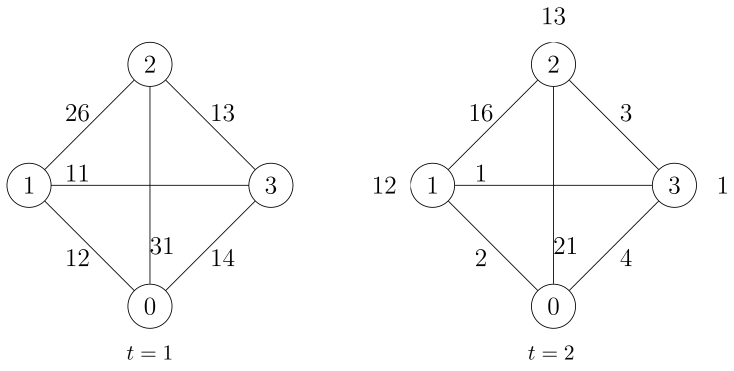


Figure 5.1.1: An mcsttr problem

We call the associated mcstt games of an mcsttr problem *mcsttr games*. Because mcsttr problems are a special case of mcstt problems, mcsttr games are subadditive. However mcsttr games are not monotone, nor are they submodular for the same reason as mcstt games are not monotone nor submodular as discussed in Chapter 4. The question arises if this also holds for mcsttr problems in which not all additional costs are higher or lower than the reduced amount of the edges. The mcsttr problem given in Example 5.1.1 is an mcsttr problem for which not all additional costs are higher or lower than the reduced amount of the edges. Still, the corresponding game of this mcsttr problem is not monotone and not submodular.

Example 5.1.1 (continued)**.** *The corresponding mcsttr game* (N, c_t) *is given in table 5.1.1. This game is not monotone since* $c_t(\{2\}) = 31 > 21 = c_t(\{2,3\})$*. This game is also not submodular because* $c_t(\{1,2\}) + c_t(\{2,3\}) = 59 < 61 = c_t(N) + c_t(\{2\})$*.*

S	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	N
$c_t(S)$	0	12	31	5	38	14	21	30

Table 5.1.1: The mcsttr game corresponding to the mcsttr problem $(N, 0, w_1, w_2, \alpha, r)$

The mcsttr problem (actually this holds for all mcstt problems) can be split into three different problems. First, we want to know which agents connect to the source at $t = 1$. Secondly, we want to find an mcstt of the mcsttr problem, knowing which agents connect to the source at $t = 1$. In section 4.2 we already saw an algorithm that finds such an mcstt. Finally we want to divide the cost of this mcstt fairly among the agents. In the following subsections, we will deal with these problems separately.

5.2 In search of which agents connect to the source at which point in time.

One possibility to find which agents connect to the source at which point in time may be ordering the agents by increasing additional cost α. The question then is; can we by means of an ordering on the agents say something about which agents connect to the source at $t = 1$? For example, if we know one agent connects to the source at $t = 1$, do the agents next in the order also connect to the source at $t = 1$? Unfortunately, this is not the case as can be seen in Example 5.2.1

Example 5.2.1. *Consider the mcsttr problem* $(N, 0, w_1, w_2, \alpha, r)$ *given in figure 5.1.1. The optimal way for the agents in* N *to connect to the source is for agent* 1 *to connect to the source at* $t = 1$ *and for agents 2 and 3 to connect to the source at* $t = 2$*. However, if we order the agents on the value of* α *we get the ordering* $\sigma = (312)$*. Despite the fact that agent* 1 *connects to the source at* $t = 1$*, agent 2 – which has a higher additional cost than agent 1 and is therefore next in the order – connects to the source at* $t = 2$*.*

It is remarkable that even for agents with an additional cost higher than r this does not need to be the case. In Example 5.2.1 agent 2 has a cost higher than r and still connects to the source at $t = 2$.

Recall from section 4.3 that there are $2^{|S|}$ possibilities for agents in S to connect to the source. If we look at the grand coalition, N, we can reduce the number of possibilities in which agents can connect to the source by looking at the additional costs of the agents. Look for example at the mcsttr problem given in figure 5.1.1. We have 8 possibilities in which the agents can connect to the source. Because α_1 and α_2 are higher than $r = 10$ – the amount in which the edges are reduced – we can delete the possibilities where $S_1 = \{1,3\}$ and $S_1 = \{2,3\}$. Agent 1 would not be the only one to connect to the source at $t = 2$ since then she would rather connect to the source at $t = 1$. The same holds for agent 2. We can also delete the possibility $S_1 = \{3\}$, since if agents 1 and 2 connect to the source at $t = 2$, agent 3 will also connect to the source at $t = 2$ because $\alpha_3 < r$. The number of possibilities that we can delete for N agents with additional costs α is given in Theorem 5.2.2.

Theorem 5.2.2. *Let $(N, 0, w_1, w_2, \alpha, r)$ be an mcsttr problem. Let $\sigma \in \Pi(N)$ be such that σ orders the agents on decreasing additional cost, i.e., $\sigma(i) \geq \sigma(j)$ if and only if $\alpha_i \leq \alpha_j$. Let k be the number of agents i with $\alpha_i \geq r$ and let S_1 be the set of agents that connect to the source at $t = 1$. Then we can delete all possibilities in which*

$$N \setminus S_1 \subseteq \{\sigma(1), \ldots, \sigma(k)\}, N \setminus S_1 \neq \emptyset \text{ and } S_1 \subseteq \{\sigma(k+1), \ldots, \sigma(n)\}, S_1 \neq \emptyset.$$

Furthermore, the number of possibilities in which n agents can connect to the source either at $t = 1$ or $t = 2$ is equal to

$$2^n - 2^k - 2^{n-k} + 2,$$

Proof. Note that we took $\alpha_i \geq r$ instead of $\alpha_i > r$ because when $\alpha_i = r$, agent i is indifferent between connecting to the source at $t = 1$ and at $t = 2$. Because σ orders the agents on decreasing additional cost and there are k agents i with $\alpha_i \geq r$, all agents in $\{\sigma(1), \ldots, \sigma(k)\}$ have an additional cost higher than or equal to r. If $N \setminus S_1 \subseteq \{\sigma(1), \ldots, \sigma(k)\}$ and $N \setminus S_1 \neq \emptyset$, all agents in $N \setminus S_1$ would rather connect to the source at $t = 1$. If $S_1 \subseteq \{\sigma(k+1), \ldots, \sigma(n)\}$, $S_1 \neq \emptyset$, all agents in S_1 have an additional cost lower than r. Therefore all agents in S_1 would rather connect to the source at $t = 2$. Hence, we can delete all possibilities in which $N \setminus S_1 \subseteq \{\sigma(1), \ldots, \sigma(k)\}, N \setminus S_1 \neq \emptyset$ and $S_1 \subseteq \{\sigma(k+1), \ldots, \sigma(n)\}, S_1 \neq \emptyset$.

There are $2^k - 1$ subsets of $\{\sigma(1), \ldots, \sigma(k)\}$ without the emptyset and there are $2^{n-k} - 1$ subsets of $\{\sigma(k+1), \ldots, \sigma(n)\}$ without the emptyset. There are 2^n possibilities for agents in N to connect to the source. Therefore the reduced number of possibilities in which n agents can connect to the source either at $t = 1$ or $t = 2$ is equal to

$$2^n - (2^k - 1 + 2^{n-k} - 1) = 2^n - 2^k - 2^{n-k} + 2.$$

□

Corollary 5.2.3. *Let $(N, 0, w_1, w_2, \alpha, r)$ be an mcsttr problem. Let k be the number of agents i with $\alpha_i \geq r$. If $k = 0$ or $k = n$, there is only one possibility for the agents in N to connect to the source. If $k = 1$ or $k = n - 1$, we can delete half of the possibilities in which agents can connect to the source. Furthermore, the number of possibilities we can delete decreases when k goes from $k = 0$ to $k = \lfloor \frac{k}{2} \rfloor$ and it increases again when k goes from $\lfloor \frac{k}{2} \rfloor$ to $k = n$.*

5.3 Knowing more about the shape of an mcstt in the mcsttr problem

One way to know more about the shape of an mcstt in the mcsttr problem $(N, 0, w_1, w_2, \alpha, r)$ can be by comparing this mcstt with the mcst in $(N, 0, w_1)$. If these trees were to be the same, this would make things a lot easier because we then would know which edges are constructed by the agents. The only thing left to find out would then be at which point in time it is best to construct these edges. Unfortunately this is not always the case. Consider for example an mcstt in $(N, 0, w_1, w_2, \alpha, r)$ in which one agent connects to the source at $t = 2$, say agent 1, and the other agents connect to the source at $t = 1$. Every agent i other than agent 1 that would have connected to the source through a direct edge incident to agent 1 in case of the mcst problem $(N, 0, w_1)$, now has to connect to the source through some other edge. Hence, the mcstt in $(N, 0, w_1, w_2, \alpha, r)$ and the mcst in $(N, 0, w_1)$ do not have to be the same, see also Example 5.3.1.

Example 5.3.1. *Consider the mcsttr problem $(N, 0, w_1, w_2, \alpha, r)$ given in figure 5.3.1 in which $r = 5$. The optimal way for the 3 agents to connect to the source is for agents 2 and 3 to connect to the source at $t = 1$ by constructing the edges $\{0, 2\}$ and $\{0, 3\}$ and for agent 1 to connect to the source at $t = 2$ by constructing edge $\{0, 1\}$. The mcstt in $(N, 0, w_1, w_2, \alpha, r)$ therefore equals the graph (N_0, T) with $T = T_1 \cup T_2 = \{\{0, 1\}, \{0, 2\}, \{0, 3\}\}$. The mcst in $(N, 0, w_1)$ is equal to the graph (N_0, T^*) with $T^* = \{\{0, 1\}, \{1, 2\}, \{1, 3\}\}$. The edgesets T and T^* therefore differ in all edges except for one. Note that agents 2 and 3 would construct an edge directly through agent 1 if agent 1 had constructed an edge at $t = 1$. Because it is best for all agents* *if agent 1 constructs an edge at* $t = 2$*, all edges directly incident to agent* 1 *cannot be part of the mcstt in* $(N, 0, w_1, w_2, \alpha, r)$*.*

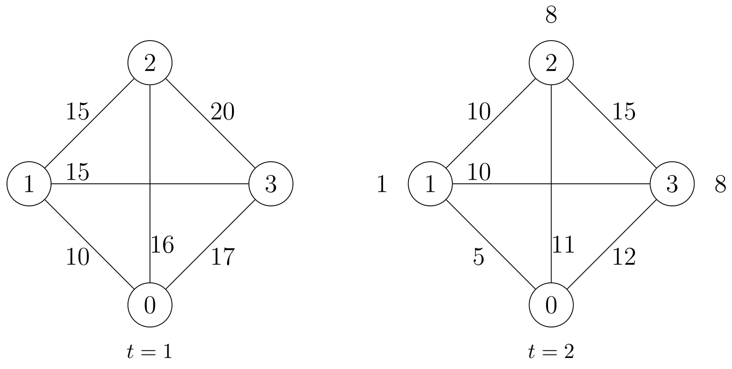


Figure 5.3.1: An mcsttr problem.

So the edgesets of the mcst in the mcst problem $(N, 0, w_1)$ and the mcstt in the mcsttr problem $(N, 0, w_1, w_2, \alpha, r)$ do not have to be the same. However, the edges that are constructed at $t = 2$ are always contained in at least one edgeset of an mcst in $(N, 0, w_1)$, i.e., there exists an mcst (N_0, T^*) in $(N, 0, w_1)$ such that $T_2 \subseteq T^*$ where $(N_0, T_1 \cup T_2)$ is an mcstt in $(N, 0, w_1, w_2, \alpha, r)$ as is shown in Theorem 5.3.2.

Theorem 5.3.2. *Let* $(N, 0, w_1, w_2, \alpha, r)$ *be an mcsttr problem and let* $(N, 0, w_1)$ *be the corresponding mcst problem at* $t = 1$*. Let* (N_0, T) *be an mcstt in* $(N, 0, w_1, w_2, \alpha, r)$ *where* $T = T_1 \cup T_2$ *such that* $T_1, T_2 \subseteq E_{N_0}, T_1 \cap T_2 = \emptyset$*. Suppose* $(N, 0, w_1)$ *has* m *mcst's, i.e.* (N_0, T_i^*) *is an mcst in* $(N, 0, w_1)$ *for* $i \in \{1, \ldots, m\}$*. Then* $T_2 \subseteq \bigcup_i T_i^*$*.*

Proof. Let us construct the mcstt (N_0, T) using Algorithm 7. Let S_1 be the set of agents that connect to the source at $t = 1$. Step 1 of the algorithm returns a tree $((S_1)_0, T_1)$. In step 2, for every agent in $N \setminus S_1$ an edge e of minimal cost is added to the edgeset T_2 using the vertex oriented construct procedure such that eventually $(N_0, T_1 \cup T_2)$ is an mcstt in $(N, 0, w_1, w_2, \alpha, r)$. Now suppose in the $k - th$ step in step 2 of the algorithm, an edge $e_k \in T_2$ is constructed such that $e_k \notin \bigcup_i T_i^*$. Edge e_k connects a component of (N_0, T_σ^{k-1}) which contains agent $\sigma(k)$ to a component of (N_0, T_σ^{k-1}) which does not contain agent $\sigma(k)$. Adding edge e_k to a random edgeset T_i^* means creating a unique cycle C. Therefore, there exists an edge $e_i^* \in C \setminus \{e_k\}$ connecting the same components as edge e_k. Because the vertex oriented construct procedure chooses to construct e_k in the $k - th$ step of the algorithm, instead of e_i^*, we have $w(e_i^*) \geq w(e_k)$. But then we have $w_1(T_i^* \cup \{e_k\} \setminus \{e_i^*\}) \leq w_1(T_i^*)$. If

$w_1(T_i^* \cup \{e_k\} \setminus \{e_i^*\}) < w_1(T_i^*)$ this is a contradiction since (N_0, T_i^*) is an mcst in $(N, 0, w_1)$. If $w_1(T_i^* \cup \{e_k\} \setminus \{e_i^*\}) = w_1(T_i^*)$ this is also a contradiction since there are only m mcst's in $(N, 0, w_1)$ and none of them contain the edge e_k. □

So there exists an mcst (N_0, T^*) in $(N, 0, w_1)$ such that T and T^* have at least one edge in common. Notice that Theorem 5.3.2 only holds for mcstt problems with constant reduced edges. It does not hold for general mcstt problems since in general mcstt problems, an mcst in $(N, 0, w_1)$ does not need to be the same as an mcst in $(N, 0, w_2)$. In fact, for $|N| > 3$ there exist mcstt problems $(N, 0, w_1, w_2, \alpha)$ in which the mcst in $(N, 0, w_1)$ does not even has one edge in common with the mcstt in $(N, 0, w_1, w_2, \alpha)$, see Example 5.3.3.

Example 5.3.3. *Consider the mcstt problem $(N, 0, w_1, w_2, \alpha)$ with $|N| = 4$ depicted in figure 5.3.2. In table 5.3.1 all possible options for the agents to connect to the source are given with their corresponding costs and edgesets T_1 and T_2, where T_1 consists of the edges constructed at $t = 1$ and T_2 consists of the edges constructed at $t = 2$. We can see that the optimal way for the four agents to connect to the source is for agent 3 to connect to the source at $t = 1$ and for agents 1, 2 and 4 to connect to the source at $t = 2$, in which the four agents have to pay a total cost of* 71 *. Let us denote the mcst in $(N, 0, w_1)$ by (N_0, T^*) and the mcstt in $(N, 0, w_1, w_2, \alpha)$ by (N, T) where $T = T_1 \cup T_2$. Then $T^* = \{\{0,1\},\{1,2\},\{2,3\},\{0,4\}\}$ and $T = \{\{0,2\},\{0,3\},\{1,3\},\{2,4\}\}$. We can see that the sets of edges T^* and T do not have any edge in common.*

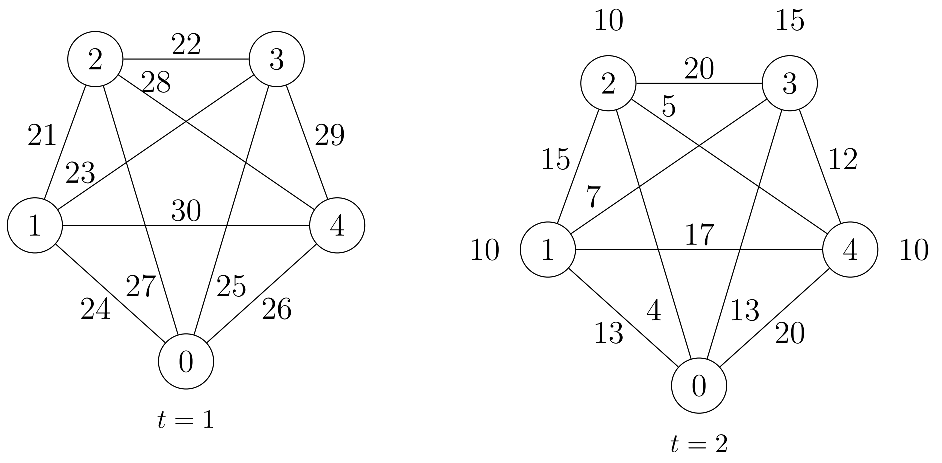


Figure 5.3.2: An mcstt problem.

S_1	$c(N \mid S_1)$	T_1	T_2
$\emptyset$	73	$\emptyset$	$\{\{0,2\},\{1,3\},\{2,4\},\{3,4\}\}$
$\{1\}$	75	$\{\{0,1\}\}$	$\{\{0,2\},\{1,3\},\{2,4\}\}$
$\{2\}$	86	$\{\{0,2\}\}$	$\{\{1,3\},\{2,4\},\{3,4\}\}$
$\{3\}$	71	$\{\{0,3\}\}$	$\{\{0,2\},\{1,3\},\{2,4\}\}$
$\{4\}$	84	$\{\{0,4\}\}$	$\{\{0,2\},\{1,3\},\{3,4\}\}$
$\{1,2\}$	82	$\{\{0,1\},\{1,2\}\}$	$\{\{1,3\},\{2,4\}\}$
$\{1,3\}$	76	$\{\{0,1\},\{1,3\}\}$	$\{\{0,2\},\{2,4\}\}$
$\{1,4\}$	86	$\{\{0,1\},\{0,4\}\}$	$\{\{0,2\},\{1,3\}\}$
$\{2,3\}$	79	$\{\{0,3\},\{2,3\}\}$	$\{\{1,3\},\{2,4\}\}$
$\{2,4\}$	97	$\{0,2\},\{0,4\}\}$	$\{\{1,3\},\{3,4\}\}$
$\{3,4\}$	82	$\{\{0,3\},\{0,4\}\}$	$\{\{0,2\},\{1,3\}\}$
$\{1,2,3\}$	82	$\{\{0,1\},\{1,2\},\{1,3\}\}$	$\{\{2,4\}\}$
$\{1,2,4\}$	93	$\{\{0,1\},\{1,2\},\{0,4\}\}$	$\{\{1,3\}\}$
$\{1,3,4\}$	87	$\{\{0,1\},\{1,3\},\{0,4\}\}$	$\{\{0,2\}\}$
$\{2,3,4\}$	90	$\{\{0,3\},\{0,4\},\{2,3\}\}$	$\{\{1,3\}\}$
N	93	$\{\{0,1\},\{1,2\},\{2,3\},\{0,4\}\}$	$\emptyset$

Table 5.3.1: All possible options for the four agents to connect to the source with their corresponding cost and edgesets T_1 and T_2.

5.4 In search of a fair cost allocation method for the problem with 3 agents

The question arises if for an mcsttr problem $(N, 0, w_1, w_2, \alpha, r)$ Algorithm 8, as presented in the previous chapter, does provide a cost allocation that is an element of the core. Unfortunately, also for this special class of mcstt problems this is not the case. Even if we set restrictions on the amount one of the allocation rules allocates in the first step in such a way that the allocation of coalition $S \subseteq S_1$ is at most $c_t(S)$ it still does not have to be a core element, as can be seen in Example 5.4.1.

Example 5.4.1. *Consider the mcsttr problem $(N, 0, w_1, w_2, \alpha, r)$ given in figure 5.4.1 and the corresponding mcsttr game (N, c_t) given in table 5.4.1. The core of the mcsttr game has to meet the following equations: $x_3 \leq 2$ and since $x_1 + x_2 \leq 21$ we have $x_3 = 2$. Also $x_1 \leq 7$, $x_2 \leq 100, x_1 \geq -79$ and $x_2 \geq 16$. Therefore $Core(c_t) = conv\{(5, 16, 2), (-79, 100, 2)\}$. Setting restrictions on the amount the Bird rule allocates in the first step of Algorithm 8 in such a way that the allocation of coalition $S \subseteq \{1, 2\}$ is at most $c_t(S)$ gives us the core allocation $(7, 14, 2)$. But*

$(7, 14, 2) \notin Core(c_t)$.

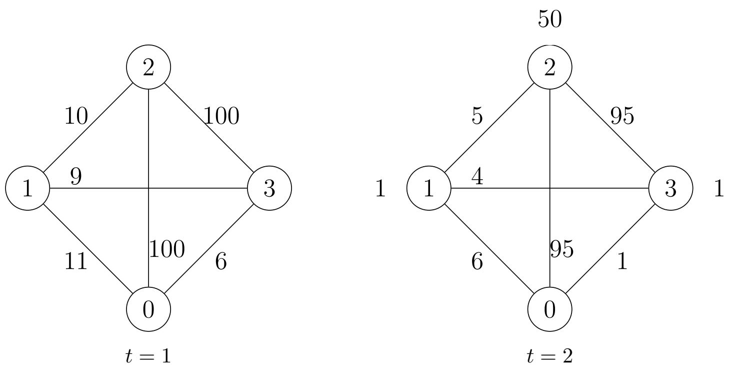


Figure 5.4.1: An mcsttr problem.

S	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	N
$c_t(S)$	0	7	100	2	21	7	102	23

Table 5.4.1: The mcsttr game associated with the mcsttr problem $(N, 0, w_1, w_2, \alpha, r)$.

To construct a cost allocation method for a 3-agent mcsttr problem, we distinguish 4 cases:

(1) Every agent connects to the source at $t = 2$.

(2) One agent connects to the source at $t = 1$.

(3) Two agents connect to the source at $t = 1$

(4) All agents connect to the source at $t = 1$.

In the first and second case we can just use Algorithm 8 to compute an element of the core of the associated mcsttr game. In the third and fourth case, we need a new allocation method that provides an allocation vector which is an element of the core, and this allocation method differs for both cases. The allocation method for the third case is given in Algorithm 9 and the allocation method for the fourth case is given in Algorithm 10. Notice that in all three algorithms we assume we know which agents connect to the source at $t = 1$.

Algorithm 9 Cost allocation method for a 3-agent mcsttr problem in which 2 agents connect to the source at $t = 1$

```
Input: $N = \{1,2,3\}$ vertices, $S_1 \subseteq N, S_1 = |2|$, source 0, weight functions
$w_1, w_2 : E_{N_0} \to \mathbb{R}_+$, additional cost vector $\alpha \in \mathbb{R}^N_+$.
Output: Minimum cost spanning tree with time $(N_0, T)$ that connects all
vertices in $N$ to 0, the associated edgesets $T_1$ and $T_2$ and a cost allocation vector $x$.

1: Initialize $T_1 = \emptyset$, $T_2 = \emptyset$.
2: Find the cheapest edge $e_1$ incident to agent $i \in N \setminus S_1$. Add edge $e_1$ to $T_1$ and
   assign cost $w_2(e_1) + \alpha_i$ to agent $i$, i.e. $x_i = w_2(e_1) + \alpha_i$.
3: Find the cheapest edge $e_2$ between a vertex $j \in S_1$ and the source, i.e., $e_2 =
   \{0,j\}$. Add edge $e_2$ to $T_1$ and assign cost $x_j = \min\{w_1(\{0,j\}), w_2(\{0,j\}) +
   \alpha_j, w_1(\{0,i\}) + w_1(\{i,j\}) - x_i, w_2(\{0,i\}) + w_2(\{i,j\}) + \alpha_i + \alpha_j - x_i\}$ to agent $j$.
4: Find the cheapest edge $e_3$ between vertex $k \in S_1 \setminus \{j\}$ and $\{0,j\}$ such that
   $(N_0, T_1 \cup T_2 \cup \{e_3\})$ does not contain a cycle. Add $e_3$ to $T_1$ and assign cost
   $x_k = w_1(e_2) + w_1(e_3) - x_j$ to agent $k$.
5: Set $T = T_1 \cup T_2$.
```

Algorithm 9 constructs an mcstt (N_0, T) and a cost allocation vector x as follows: first, the one agent that connects to the source at $t = 2$ (agent i) constructs the cheapest edge (edge e_1) she is incident to and pays the cost of this edge plus α_i, i.e., $x_i = w_2(e_1) + \alpha_i$. Then, a cheapest edge (edge e_2) between an agent (agent j) that connects to the source at $t = 1$ and the source is constructed. Because it is possible that if only agent i and j were to cooperate, agent i would construct an edge at $t = 1$ instead of at $t = 2$, we cannot just allocate the cost of edge e_2 to agent j. Instead, we allocate the cost $c_t(\{i,j\}) - x_i$ to agent j. There are four possibilities in which we can construct two edges that connect agent i and j to the source, taken into account the fact that an agent only constructs an edge at $t = 1$ if this means that agent is ultimately connected to the source at $t = 1$ and taken into account the fact that $\alpha_i < r$ since agent i connects to the source at $t = 2$. Next, the remaining agent (agent k) constructs the cheapest edge (edge e_3) between herself and either agent j or the source. Now agent k pays the remaining cost, i.e., $x_k = m(N, 0, w_1, w_2, \alpha) - x_1 - x_2$.

In Algorithm 10, Prim's algorithm is used to construct an mcstt in the mcsttr problem $(N, 0, w_1, w_2, \alpha, r)$. A cost allocation vector x is constructed by the algorithm as follows: the agent (agent i) that constructs edge e_1 in the first step of the algorithm, pays the maximum amount she would pay if she were to connect to the source by herself, i.e., $x_i = c_t(\{1\})$. In the second step of the algorithm, the agent

Algorithm 10 Cost allocation method for a 3-agent mcsttr problem in which all agents connect to the source at $t = 1$

```
Input: $N = \{1,2,3\}$ vertices, source 0, weight functions $w_1, w_2 : E_{N_0} \to \mathbb{R}_+$,
additional cost vector $\alpha \in \mathbb{R}^N_+$.
Output: Minimum cost spanning tree with time $(N_0, T)$ that connects all
vertices in $N$ to 0, the associated edgesets $T_1$ and $T_2$ and a cost allocation vector $x$.

1: Initialize $T = \emptyset$.
2: Find the cheapest edge $e_1$ between an agent $i \in N$ and the source, i.e., $e_1 = \{0, i\}$.
   Add edge $e_1$ to $T$ and assign cost $x_i = \min\{w_1(e_1), w_2(e_1) + \alpha_i\}$ to agent $i$.
3: Find the cheapest edge $e_2$ between $\{0, i\}$ and an agent $j \in N \setminus \{0, i\}$. Add edge
   $e_2$ to $T$ and assign cost $x_j = \min\{w_1(e_1) + w_1(e_2), w_1(e_1) + w_2(e_2) + \alpha_j, w_2(e_1) + w_2(e_2) + \alpha_i + \alpha_j, w_2(e_1) + \alpha_i + w_1(\{0, j\})\} - x_i$ to agent $j$.
4: Find the cheapest edge $e_3$ between an agent $k$ and $\{0, i, j\}$ such that $(N_0, T \cup \{e_3\})$
   does not contain a cycle. Assign cost $x_k = w_1(e_1) + w_1(e_2) + w_1(e_3) - x_i - x_j$ to
   agent $k$
```

(agent j) that constructs edge e_2, pays the maximum amount she would pay if she were to cooperate with agent i, i.e., $x_j = c_t(\{1, 2\}) - x_i$. The third agent (agent k) pays the remaining cost, i.e. $x_k = m(N, 0, w_1, w_2, \alpha) - x_i - x_j$.

Theorem 5.4.2. *Let $(N, 0, w_1, w_2, \alpha, r)$ be an mcsttr problem with $|N| = 3$. Then the following statements are true*

(i) If every agent connects to the source at $t = 2$ (1), then the vector x constructed by Algorithm 8 is an element of the core.

(ii) If one agent connects to the source at $t = 1$ (2), then the vector x constructed by Algorithm 8 is an element of the core.

(iii) If two agents connect to the source at $t = 1$ (3), then the vector x constructed by Algorithm 9 is an element of the core.

(iv) If all agents connect to the source at $t = 1$ (4), then the vector x constructed by Algorithm 10 is an element of the core.

Proof. The proof can be found in Appendix 2. □

Example 5.4.3. *Consider the mcsttr problem $(N, 0, w_1, w_2, \alpha, r)$ presented in figure 5.4.2 and the corresponding mcsttr game (N, c_t) given in table 5.4.2. The optimal way* *for the three agents to connect to the source is for all agents to connect to the source at* $t = 2$*. Therefore, to find an mcstt in* $(N, 0, w_1, w_2, \alpha, r)$ *and an allocation vector that is an element of the core of* (N, c_t)*, by Theorem 5.4.2 we can use Algorithm 8. Since* $S_1 = \emptyset$*, nothing happens in step 1 of the algorithm. Suppose the algorithm picks* $\sigma = (123)$*. So* $\sigma(1) = 1$*,* $\sigma(2) = 2$ *and* $\sigma(3) = 3$*. Algorithm 8 first constructs the edge* $\{0, 1\}$ *since it is the cheapest edge that connects agent* 1 *with another agent or the source and allocates the cost* $w_2(\{0, 1\}) + \alpha_1 = 11$ *to agent 1. Moving on to agent 2, the algorithm constructs the edge* $\{1, 2\}$ *since this is the cheapest edge that connects agent* 2 *with another agent or the source and allocates the cost* $w_2(\{1, 2\}) + \alpha_2 = 31$ *to agent 2. Finally, the algorithm constructs the edge* $\{2, 3\}$ *because this is the cheapest edge that connects agent 3 to another agent or the source and allocates the cost* $w_2(\{2, 3\}) + \alpha_3 = 33$ *to agent 3. So algorithm 8 constructs the allocation vector* $x = (11, 31, 33)$ *and one can easily check that* x *is an element of the core of* (N, c_t)*.*

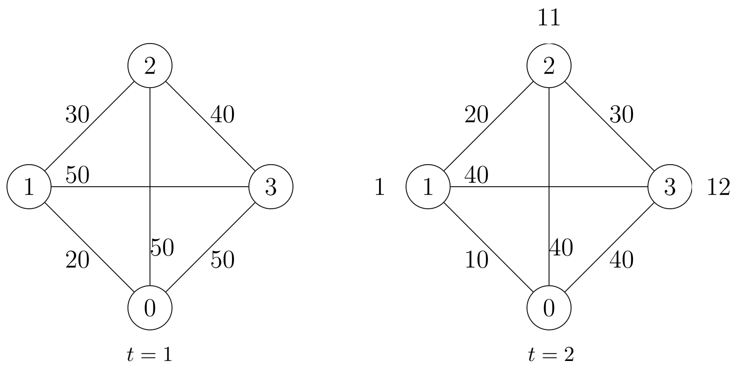


Figure 5.4.2: An mcsttr problem.

S	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	N
$c_t(S)$	0	11	50	50	42	61	90	84

Table 5.4.2: The mcsttr game associated with the mcsttr problem $(N, 0, w_1, w_2, \alpha, r)$.

Example 5.4.4. *Consider the mcsttr problem* $(N, 0, w_1, w_2, \alpha, r)$ *presented in figure 5.4.3 and the corresponding mcsttr game* (N, c_t) *given in table 5.4.3. The optimal way for the three agents to connect to the source is for agent 1 to connect to the source at* $t = 1$ *and for agents 2 and 3 to connect to the source at* $t = 2$*. Therefore, to find an mcstt in* $(N, 0, w_1, w_2, \alpha, r)$ *and an allocation vector that is an element* *of the core of* (N, c_t)*, by Theorem 5.4.2 we can use Algorithm 8. Since* $S_1 = \{1\}$*, Algorithm 8 constructs the edge* $\{0, 1\}$ *in the first step of the algorithm since it is the cheapest edge that connects an agent in* S_1 *to the source. Algorithm 8 allocates the cost* $w_1(\{0, 1\}) = 20$ *to agent 1. Let* $\sigma = (23)$*. So* $\sigma(1) = 2$ *and* $\sigma(2) = 3$*. Algorithm 8 first constructs the edge* $\{2, 3\}$ *since it is the cheapest edge that connects agent 2 with another agent or the source and allocates the cost* $w_2(\{1, 2\}) + \alpha_2 = 31$ *to agent 2. Moving on to agent 3, agent 3 is part of the component consisting of the agents 2 and 3. The cheapest edge that connects this component to another component in the graph (in this case the component consisting of the agents 1 and 2) is edge* $\{0, 3\}$*. Therefore, the algorithm constructs edge* $\{0, 3\}$ *and allocates the cost* $w_2(\{0, 3\}) + \alpha_3 = 31$ *to agent 3. So Algorithm 8 constructs the allocation vector* $x = (20, 31, 31)$ *and one can easily check that* x *is an element of the core of* (N, c_t)*.*

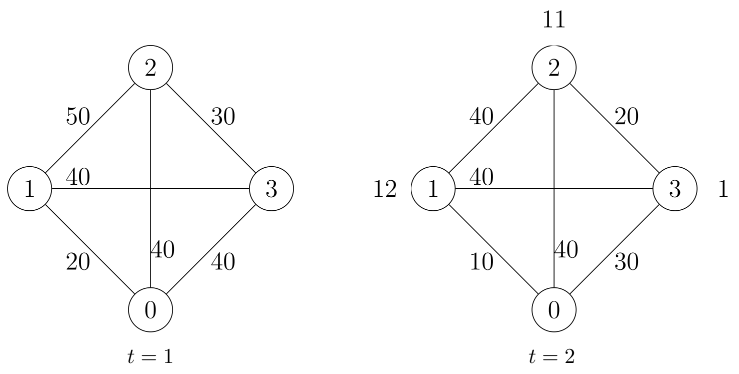


Figure 5.4.3: An mcsttr problem.

S	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	N
$c_t(S)$	0	20	50	31	70	51	72	82

Table 5.4.3: The mcsttr game associated with the mcsttr problem $(N, 0, w_1, w_2, \alpha, r)$.

Example 5.4.5. *Consider the mcsttr problem* $(N, 0, w_1, w_2, \alpha, r)$ *given in figure 5.4.1 and the corresponding mcsttr game* (N, c_t) *given in table 5.4.1. The optimal way for the three agents to connect to the source is for agent 1 and agent 2 to connect to the source at* $t = 1$ *and for agent 3 to connect to the source at* $t = 2$*. Therefore, to find an mcstt in* $(N, 0, w_1, w_2, \alpha, r)$ *and an allocation vector that is an element of the core of* (N, c_t)*, by Theorem 5.4.2 we can use Algorithm 9. Since* $N \setminus S_1 = \{3\}$*, Algorithm 9* *constructs the edge* $\{0,3\}$ *since it is the cheapest edge incident to agent 3 and allocates the cost* $x_3 = w_2(\{0,3\}) + \alpha_3 = 2$ *to agent 3. Since* $S_1 = \{1,2\}$*, Algorithm 9 then constructs the edge* $\{0,1\}$ *because it is the cheapest edge between a vertex in* $\{1,2\}$ *and the source. The algorithm assigns the cost* $x_1 = \min\{w_1(\{0,1\}), w_2(\{0,1\}) + \alpha_1, w_1(\{0,3\}) + w_1(\{1,3\}) - w_2(\{0,3\}) - \alpha_3, w_2(\{1,3\}) + \alpha_1) = \min\{11,7,8,5\} = 5$ *to agent 1. Finally, Algorithm 9 constructs edge* $\{1,2\}$ *since it is the cheapest edge between agent 2 and agent 1 or the source and assigns the cost* $x_2 = w_1(\{0,1\}) + w_1(\{1,2\}) - x_2 = 16$ *to agent 2. So algorithm 8 constructs the allocation vector* $x = (5,16,2)$ *and as can be seen in Example 5.4.1, this is an element of the core of* (N, c_t)*.*

Example 5.4.6. *Consider the mcsttr problem* $(N, 0, w_1, w_2, \alpha, r)$ *given in figure 5.4.4 and the corresponding mcsttr problem* (N, c_t) *given in table 5.4.4. The optimal way for the three agents to connect to the source is for all agents to connect to the source at* $t = 1$*. Therefore, to find an mcstt in* $(N, 0, w_1, w_2, \alpha, r)$ *and an allocation vector that is an element of the core of* (N, c_t)*, by Theorem 5.4.2 we can use Algorithm 10. The algorithm first constructs edge* $\{0,1\}$ *since it is the cheapest edge between an agent in N and the source and assigns cost* $x_1 = \min\{w_1(\{0,1\}), w_2(\{0,1\}) + \alpha_1\} = 9$ *to agent 1. Next, Algorithm 10 constructs edge* $\{1,2\}$ *because it is the cheapest edge between the source or agent 1 and an agent in* $\{2,3\}$*. Algorithm 10 assigns cost* $x_2 = \min\{w_1(\{0,1\}) + w_1(\{1,2\}), w_1(\{0,1\}) + w_2(\{1,2\}) + \alpha_2, w_2(\{0,1\}) + w_2(\{1,2\}) + \alpha_1 + \alpha_2, w_2(\{0,1\}) + \alpha_1 + w_1(\{0,2\})\} - x_1 = \min\{20,19,18,109\} - 9 = 9$ *to agent 2. Finally, the algorithm constructs edge* $\{2,3\}$ *since it is the cheapest edge between agent 3 and another agent or the source. Algorithm 10 assigns cost* $x_3 = w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{2,3\}) - x_1 - x_2 = 30 - 18 = 12$ *to agent 3. So Algorithm 10 constructs the allocation vector* $x = (9,9,12)$ *and one can easily check that x is an element of the core of* (N, c_t)*.*

S	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	N
$c_t(S)$	0	9	99	100	18	109	109	30

Table 5.4.4: The mcsttr game associated with the mcsttr problem $(N, 0, w_1, w_2, \alpha, r)$.

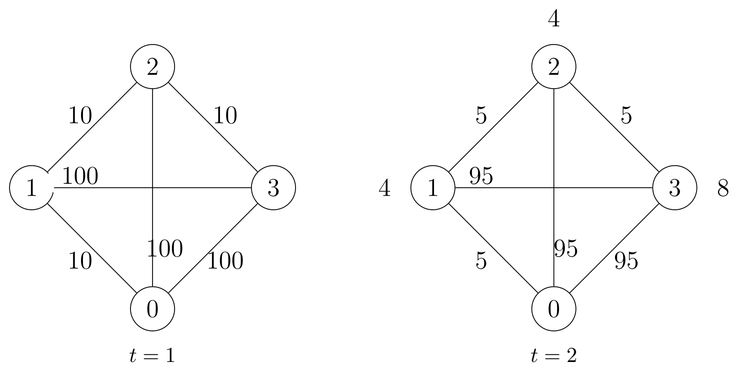


Figure 5.4.4: An mcsttr problem.

Bibliography

Aarts, H. and Driessen, T. (1993). The irreducible core of a minimum cost spanning tree game. *ZOR - Methods and Models of Operations Research*, 38:163–174.

Bergantiños, G. and Lorenzo, L. (2004). A non-cooperative approach to the cost spanning tree problem. *Mathematical Methods of Operational Research*, 59:393–403.

Bergantiños, G. and Vidal-Puga, J. (2007b). The optimistic TU game in minimum cost spanning tree problems. *International Journal of Game Theory*, 36:223–239.

Bird, C. (1976). On cost allocation for a spanning tree: a game theoretic approach. *Networks*, 6:335–350.

Branzei, R., Moretti, S., Norde, H., and Tijs, S. (2004). The p-value for cost sharing in minimum spanning tree situations. *Theory and Decision*, 56(1):47–61.

Çiftçi, B. (2009). *A cooperative approach to sequencing and connection problems*, chapter 3. CentER Dissertation Series. CentER, Center for Economic Research.

Çiftçi, B. and Tijs, S. (2009). A vertex oriented approach to the equal remaining obligations rule for minimum cost spanning tree situations. *TOP*, 17:440–453.

Claus, A. and Kleitman, D. (1973). Cost allocation for a spanning tree. *Networks*, 3:289–304.

Feltkamp, V., Tijs, S., and Muto, S. (1994). On the irreducible core and the equal remaining obligations rule of minimum cost spanning extension problems. *centER Discussion Paper*, 106.

Graham, R. and Hell, P. (1985). On the history of the minimum spanning tree problem. *Annals of the History of Computing*, 7:43–57.

Granot, D. and Huberman, G. (1977). The relationship between convex games and minimal spanning tree games: A case for permutationally convex games. *SIAM. J. on Algebraic and Discrete Methods*, 3:288–292.

Granot, D. and Huberman, G. (1981). Minimum cost spanning tree games. *Mathematical Programming*, 21:1–18.

Ichiishi, T. (1981). Super-modularity: applications to convex games and to the greedy algorithm for LP. *Journal of Economic Theory*, 25:283–286.

Kruskal, J. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7:48–50.

Prim, R. (1957). Shortest connection networks and some generalizations. *Bell Systems Technical Journal*, 36:1389–1401.

Quant, M., Borm, P., Reijnierse, H., and van Velzen, B. (2005). The core cover in relation to the nucleolus and the weber set. *International Journal of Game Theory*, 33:491–503.

Shapley, L., S. (1971). Cores of convex games. *International Journey of Game Theory*, 1:11–26.

von Neumann, J. and Von Morgenstern, O. (1944). *Theory of Games and Economic Behavior*. Princeton University Press.

Appendix

A.1 Proof of Theorem 4.4.2

Theorem 4.4.2. *Let $(N, 0, w_1, w_2, \alpha)$ be an mcstt problem and let (N, c_t) be the corresponding mcstt game. If $|N| = 3$, then the mcstt game has a nonempty core.*

Proof. Because all mcstt games are subadditive we only have to proof that $c_t(\{1,2\})+ c_t(\{1,3\}) + c_t(\{2,3\}) \geq 2c_t(N)$.

Because $c_t(S) = m(S, 0, w_1, w_2, \alpha)$ every $c_t(S)$ with $|S| = 2$ is made up of the cost of two edges that together form a tree (S_0, T_S) in the mcstt problem $(S, 0, w_1, w_2, \alpha)$ plus possibly some additional costs. Note that the cost of a tree (S_0, T_S) in an mcstt problem consists of the cost of the edgeset, $w(T_S)$ and additional costs for the agents that connect to the source at $t = 2$.

We claim that the 6 constructed edges by the three coalitions $\{1, 2\}, \{1, 3\}$ and $\{2, 3\}$ can be regrouped into two sets of edges T and T' such that (N_0, T) and (N_0, T') are spanning trees in $(N, 0, w_1, w_2, \alpha)$ and the sum of the costs of these two constructed spanning trees equals the sum of the costs of the three different coalitions. Here, $T = T_1 \cup T_2$ and $T' = T'_1 \cup T'_2$ where T_1 and T'_1 are the constructed edges at $t = 1$ and T_2 and T'_2 are the constructed edges at $t = 2$. Recall that (N_0, T) is a spanning tree in $(N, 0, w_1, w_2, \alpha)$ if and only if $(N(T_1)_0, T_1)$ and (N_0, T) are trees. The proof then follows since these spanning trees have a cost that is greater or equal to the cost of the mcstt in $(N, 0, w_1, w_2, \alpha)$ which equals $c_t(N)$.

To proof our claim we distinguish 10 cases. For each case, we regroup the edges constructed by the three coalitions $\{1, 2\}, \{1, 3\}$ and $\{2, 3\}$ in two edgesets T and T' such that (N_0, T) and (N_0, T') are spanning trees in $(N, 0, w_1, w_2, \alpha)$. One can easily check that (N_0, T) and (N_0, T') are spanning trees in $(N, 0, w_1, w_2, \alpha)$ and that the sum of these spanning trees is equal to the sum of the costs of the three coalitions of size 2, i.e., $c_t(\{1, 2\}) + c_t(\{1, 3\}) + c_t(\{2, 3\})$.

1. *All coalitions of two agents construct their edges at $t = 1$:*

Let $T_{\{2,3\}} = \{e, e'\}$ where e is incident to agent 2 and e' is incident to agent 3. Take $T_1 = T_{\{1,2\}} \cup \{e'\}, T_2 = \emptyset, T_1' = T_{\{1,3\}} \cup \{e\}$ and $T_2' = \emptyset$.

2. *All coalitions of two agents construct their edges at $t = 2$:*
Let $T_{\{2,3\}} = \{e, e'\}$ where e is incident to agent 2 and e' is incident to agent 3. Take $T_1 = \emptyset, T_2 = T_{\{1,2\}} \cup \{e'\}, T_1' = \emptyset$ and $T_2' = T_{\{1,3\}} \cup \{e\}$.

3. *One coalition of two agents constructs both edges at $t = 1$ while the other two coalitions of two agents construct both edges at $t = 2$:*
Without loss of generality let coalition $\{1, 2\}$ be the coalition that constructs both edges at $t = 1$ while coalitions $\{1, 3\}$ and $\{2, 3\}$ construct both their edges at $t = 2$. We can take $T_1 = T_{\{1,2\}}$. The only edge that coalitions $\{1, 3\}$ and $\{2, 3\}$ might have in common is edge $\{0, 3\}$. Therefore, we can distinguigh 3 cases:

   - Both coalitions $\{1, 3\}$ and $\{2, 3\}$ construct the edge $\{0, 3\}$: Call e_{13} the other edge constructed by coalition $\{1, 3\}$ and e_{23} the other edge constructed by coalition $\{2, 3\}$. Note that edge e_{13} has endpoints 1 and either 0 or 3 and edge e_{23} has endpoints 2 and either 0 or 3. Take $T_2 = \{\{0, 3\}\}$, $T_1' = \emptyset$ and $T_2' = \{e_{13}, e_{23}, \{0, 3\}\}$.
   - One of the coalitions $\{1, 3\}$ and $\{2, 3\}$, say coalition $\{1, 3\}$, constructs the edge $\{0, 3\}$: call e_{13} the other edge constructed by coalition $\{1, 3\}$. Then e_{13} has endpoints 1 and either 0 or 3. Take $T_2 = \{\{0, 3\}\}$, $T_1' = \emptyset$ and $T_2' = e_{13} \cup T_{\{2,3\}}$.
   - None of the coalitions $\{1, 3\}$ and $\{2, 3\}$ construct the edge $\{0, 3\}$: then $T_{\{1,3\}} = \{\{0, 1\}, \{1, 3\}\}$ and $T_{\{2,3\}} = \{\{0, 2\}, \{2, 3\}\}$. Take $T_2 = \{\{1, 3\}\}$, $T_1' = \emptyset$ and $T_2' = \{\{0, 1\}, \{0, 2\}, \{2, 3\}\} = T_{\{2,3\}} \cup \{0, 1\}$.

4. *Two coalitions of two agents construct both edges at $t = 1$ while one coalition of two agents constructs both edges at $t = 2$.*
Without loss of generality let coalition $\{1, 2\}$ and coalition $\{1, 3\}$ be the coalitions of two agents that construct both their edges at $t = 1$ while coalition $\{2, 3\}$ constructs both edges at $t = 2$. We can take $T_1 = T_{\{1,2\}}$, $T_1' = T_{\{1,3\}}$. We can distinguish 3 cases to divide the edges constructed by coalition $\{2, 3\}$ among the two edgesets T_2 and T_2'.

   - Both edges constructed by $\{2, 3\}$ at $t = 2$ are the same as two edges constructed by the coalitions $\{1, 2\}$ and $\{1, 3\}$ at $t = 1$: This means that $T_{\{2,3\}} = \{\{0, 2\}, \{0, 3\}\}$. Now let $T_2 = \{\{0, 3\}\}$ and $T_2' = \{\{0, 2\}\}$.

  - One of the edges, say edge e, constructed by $\{2,3\}$ at $t = 2$ is the same as one edge constructed by either coalition $\{1,2\}$ or coalition $\{1,3\}$: If edge e is the same as an edge constructed by coalition $\{1,2\}$, then take $T_2' = \{e\}$ and $T_2 = T_{\{2,3\}} \setminus \{e\}$. If e is the same as an edge constructed by coalition $\{1,3\}$, then take $T_2 = \{e\}$ and $T_2' = T_{\{2,3\}} \setminus \{e\}$.
  - None of the edges constructed by coalition $\{2,3\}$ at $t = 2$ is the same as an edge constructed by the coalitions $\{1,2\}$ and $\{1,3\}$ at $t = 1$: the possibilities for $T_{\{2,3\}}$ are the sets of edges $\{\{0,2\},\{0,3\}\},\{\{0,2\},\{2,3\}\}$ and $\{\{0,3\},\{2,3\}\}$. In the first case take $T_2 = \{\{0,3\}\}$ and $T_2' = \{\{0,2\}\}$. In the second case take $T_2 = \{\{2,3\}\}$ and $T_2' = \{\{0,2\}\}$ and in the last case take $T_2 = \{\{0,3\}\}$ and $T_2' = \{\{2,3\}\}$.

5. *One coalition of two agents constructs one edge at* $t = 1$ *and one edge at* $t = 2$ *while the other two coalitions of two agents construct both their edges at* $t = 1$. Without loss of generality let coalition $\{1,2\}$ and coalition $\{1,3\}$ be the coalitions of two agents that construct both their edegs at $t = 1$, while coalition $\{2,3\}$ constructs one edge at $t = 1$ and one edge at $t = 2$. We can distinguish 2 cases:

   - In coalition $\{2,3\}$, agent 2 connects to the source at $t = 1$ and agent 3 connects to the source at $t = 2$: Then edge $\{0,2\}$ is constructed at $t = 1$ and edge e is constructed at $t = 2$ with endpoints 3 and either 0 or 2. Take $T_1 = T_{\{1,2\}}$, $T_2 = \{e\}$, $T_1' = T_{\{1,3\}} \cup \{0,2\}$ and $T_2' = \emptyset$.
   - In coalition $\{2,3\}$, agent 3 connects to the source at $t = 1$ and agent 2 connects to the source at $t = 1$. Then edge $\{0,3\}$ is constructed at $t = 1$ and edge e is constructed at $t = 2$ with endpoints 2 and either 0 or 3. Take $T_1 = T_{\{1,2\}} \cup \{0,3\}, T_2 = \emptyset, T_1' = T_{\{1,3\}}$ and $T_2' = \{e\}$.

6. *One coalition of two agents constructs one edge at* $t = 1$ *and one edge at* $t = 2$ *while the other two coalitions of two agents construct both their edges at* $t = 2$. Without loss of generality let coalition $\{1,2\}$ be the coalition that constructs one edge at $t = 1$ and one edge at $t = 2$, while coalitions $\{1,3\}$ and coalition $\{2,3\}$ construct both their edges at $t = 2$. We can distinguish 2 cases:

   - In coalition $\{1,2\}$, agent 1 connects to the source at $t = 1$ and agent 2 connects to the source at $t = 2$. Then edge $\{0,1\}$ is constructed at $t = 1$ and edge e is constructed at $t = 2$ with endpoints 2 and either 0 or 1. Take $T_1 = \{0,1\}, T_2 = T_{\{2,3\}}, T_1' = \emptyset$ and $T_2' = T_{\{1,3\}} \cup e$.

- In coalition $\{1,2\}$, agent 2 connects to the source at $t=1$ and agent 1 connects to the source at $t=2$. Then edge $\{0,2\}$ is constructed at $t=1$ and edge e is constructed at $t=2$ with endpoints 1 and either 0 or 2. Take $T_1=\{\{0,2\}\}, T_2=T_{\{1,3\}}, T_1'=\emptyset$ and $T_2'=T_{\{2,3\}}\cup e$.

7. *One coalition of two agents constructs one edge at $t=1$ and one edge at $t=2$, one coalition of two agents constructs both edges at $t=1$ and one coalition of two agents constructs both edges at $t=2$.*
Without loss of generality let coalition $\{1,2\}$ be the coalition that constructs one edge at $t=1$ and one edge at $t=2$. Let coalition $\{1,3\}$ be the coalition that constructs both edges at $t=1$ and let coalition $\{2,3\}$ be the coalition that constructs both edges at $t=2$. We can distinguish 2 cases:

   - In coalition $\{1,2\}$, agent 1 connects to the source at $t=1$ and agent 2 connects to the source at $t=2$. Then edge $\{0,1\}$ is constructed at $t=1$ and edge e is constructed at $t=2$ with endpoints 2 and either 0 or 1. Take $T_1=T_{\{1,3\}}, T_2=\{e\}, T_1'=\{\{0,1\}\}$ and $T_2'=T_{\{2,3\}}$.
   - In coalition $\{1,2\}$, agent 2 connects to the source at $t=1$ and agent 1connects to the source at $t=2$. Then edge $\{0,2\}$ is constructed at $t=1$ and edge e is constructed at $t=2$ with endpoints 1 and either 0 or 2. Take $T_1=\{0,2\}\cup T_{\{1,3\}}, T_2=\emptyset, T_1'=\emptyset$ and $T_2'=T_{\{2,3\}}\cup e$.

8. *Two coalitions of two agents construct one edge at $t=1$ and one edge at $t=2$ while the other coalition of two agents construct both edges at $t=1$.*
Without loss of generality let coalition $\{1,2\}$ and coalition $\{1,3\}$ be the coalitions that construct one edge at $t=1$ and one edge at $t=2$. Let coalition $\{2,3\}$ be the coalition that constructs both edges at $t=1$. We can distinguish 4 cases:

   - In coalition $\{1,2\}$, agent 1 connects to the source at $t=1$ and agent 2 connects to the source at $t=2$. Then edge $\{0,1\}$ is constructed at $t=1$ and edge e_{12} is constructed at $t=2$ with endpoints 2 and either 0 or 1. In coalition $\{1,3\}$ agent 1 connects to the source at $t=1$ and agent 3 connects to the source at $t=2$. Then edge $\{0,1\}$ is constructed at $t=1$ and edge e_{13} is constructed at $t=2$ with endpoints 3 and either 0 or 1. Take $T_1=T_{\{2,3\}}\cup\{0,1\}$, $T_2=\emptyset$, $T_1'=\{0,1\}$ and $T_2'=\{e_{12},e_{13}\}$.
   - In coalition $\{1,2\}$, agent 1 connects to the source at $t=1$ and agent 2 connects to the source at $t=2$. Then edge $\{0,1\}$ is constructed at $t=1$ and edge e_{12} is constructed at $t=2$ with endpoints 2 and either 0 or 1.

In coalition $\{1,3\}$ agent 3 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0,3\}$ is constructed at $t = 1$ and edge e_{13} is constructed at $t = 2$ with endpoints 1 and either 0 or 3. Take $T_1 = T_{\{2,3\}} \cup \{0,1\}$, $T_2 = \emptyset$, $T_1' = \{0,3\}$, and $T_2' = \{e_{12}, e_{23}\}$.

- In coalition $\{1,2\}$, agent 2 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0,2\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 1 and either 0 or 2. In coalition $\{1,3\}$ agent 1 connects to the source at $t = 1$ and agent 3 connects to the source at $t = 2$. Then edge $\{0,1\}$ is constructed at $t = 1$ and edge e_{13} is constructed at $t = 2$ with endpoints 3 and either 0 or 1. Take $T_1 = T_{\{2,3\}}, T_2 = e_{12}, T_1' = \{\{0,1\},\{0,2\}\}$ and $T_2' = e_{13}$.
- In coalition $\{1,2\}$, agent 2 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0,2\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 1 and either 0 or 2. In coalition $\{1,3\}$ agent 3 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0,3\}$ is constructed at $t = 1$ and edge e_{13} is constructed at $t = 2$ with endpoints 1 and either 0 or 3. Take $T_1 = T_{\{2,3\}}, T_2 = \{e_{12}\}, T_1' = \{\{0,2\},\{0,3\}\}$ and $T_2' = \{e_{13}\}$.

9. *Two coalitions of two agents construct one edge at* $t = 1$ *and one edge at* $t = 2$ *while the other coalition of two agents constructs both edges at* $t = 2$.
Without loss of generality let coalition $\{1,2\}$ and coalition $\{1,3\}$ be the coalitions that construct one edge at $t = 1$ and one edge at $t = 2$. Let coalition $\{2,3\}$ be the coalition that constructs both edges at $t = 2$. We can distinguish 4 cases:

- In coalition $\{1,2\}$, agent 1 connects to the source at $t = 1$ and agent 2 connects to the source at $t = 2$. Then edge $\{0,1\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 2 and either 0 or 1. In coalition $\{1,3\}$ agent 1 connects to the source at $t = 1$ and agent 3 connects to the source at $t = 2$. Then edge $\{0,1\}$ is constructed at $t = 1$ and edge e_{13} is constructed at $t = 2$ with endpoints 3 and either 0 or 1. Take $T_1 = \{0,1\}, T_2 = T_{\{2,3\}}, T_1' = \{0,1\}$ and $T_2' = \{e_{12}, e_{13}\}$.
- In coalition $\{1,2\}$, agent 1 connects to the source at $t = 1$ and agent 2 connects to the source at $t = 2$. Then edge $\{0,1\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 2 and either 0 or 1. In coalition $\{1,3\}$ agent 3 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0,3\}$ is constructed at $t = 1$

and edge e_{13} is constructed at $t = 2$ with endpoints 1 and either 0 or 3. Take $T_1 = \{\{0, 1\}\}$, $T_2 = T_{\{2,3\}}, T_1' = \{\{0, 3\}\}$ and $T_2' = \{e_{12}, e_{13}\}$.

- In coalition $\{1, 2\}$, agent 2 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0, 2\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 1 and either 0 or 2. In coalition $\{1, 3\}$ agent 1 connects to the source at $t = 1$ and agent 3 connects to the source at $t = 2$. Then edge $\{0, 1\}$ is constructed at $t = 1$ and edge e_{13} is constructed at $t = 2$ with endpoints 3 and either 0 or 1. Take $T_1 = \{0, 1\}, T_2 = T_{\{2,3\}}, T_1' = \{0, 2\}$ and $T_2' = \{e_{12}, e_{12}\}$.
- In coalition $\{1, 2\}$, agent 2 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0, 2\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 1 and either 0 or 2. In coalition $\{1, 3\}$ agent 3 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0, 3\}$ is constructed at $t = 1$ and edge e_{13} is constructed at $t = 2$ with endpoints 1 and either 0 or 3. Take $T_1 = \emptyset, T_2 = T_{\{2,3\}} \cup e_{12}, T_1' = \{\{0, 2\}, \{0, 3\}\}$ and $T_2' = \{e_{13}\}$.

10. *All coalitions of two agents construct one edge at* $t = 1$ *and one edge at* $t = 2$. We can distinguish 8 cases:

    - In coalition $\{1, 2\}$, agent 1 connects to the source at $t = 1$ and agent 2 connects to the source at $t = 2$. Then edge $\{0, 1\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 2 and either 0 or 1. In coalition $\{1, 3\}$, agent 1 connects to the source at $t = 1$ and agent 3 connects to the source at $t = 2$. Then edge $\{0, 1\}$ is constructed at $t = 1$ and edge e_{13} is constructed at $t = 2$ with endpoints 3 and either 0 or 1. In coalition $\{2, 3\}$, agent 2 connects to the source at $t = 1$ and agent 3 connects to the source at $t = 2$. Then edge $\{0, 2\}$ is constructed at $t = 1$ and edge e_{23} is constructed at $t = 2$ with endpoints 3 and either 0 or 2. Take $T_1 = \{\{0, 1\}\}$, $T_2 = \{e_{12}, e_{13}\}, T_1' = \{\{0, 1\}, \{0, 2\}\}$ and $T_2' = \{e_{23}\}$.
    - In coalition $\{1, 2\}$, agent 1 connects to the source at $t = 1$ and agent 2 connects to the source at $t = 2$. Then edge $\{0, 1\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 2 and either 0 or 1. In coalition $\{1, 3\}$, agent 1 connects to the source at $t = 1$ and agent 3 connects to the source at $t = 2$. Then edge $\{0, 1\}$ is constructed at $t = 1$ and edge e_{13} is constructed at $t = 2$ with endpoints 3 and either 0 or 1. In coalition $\{2, 3\}$, agent 3 connects to the source at $t = 1$ and agent 2 connects to the source at $t = 2$. Then edge $\{0, 3\}$ is constructed at $t = 1$

and edge e_{23} is constructed at $t = 2$ with endpoints 2 and either 0 or 3. Take $T_1 = \{\{0,1\}\}, T_2 = \{e_{12}, e_{13}\}, T'_1 = \{\{0,1\},\{0,3\}\}$ and $T'_2 = \{e_{23}\}$.

- In coalition $\{1,2\}$, agent 1 connects to the source at $t = 1$ and agent 2 connects to the source at $t = 2$. Then edge $\{0,1\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 2 and either 0 or 1. In coalition $\{1,3\}$, agent 3 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0,3\}$ is constructed at $t = 1$ and edge e_{13} is constructed at $t = 2$ with endpoints 1 and either 0 or 3. In coalition $\{2,3\}$, agent 2 connects to the source at $t = 1$ and agent 3 connects to the source at $t = 2$. Then edge $\{0,2\}$ is constructed at $t = 1$ and edge e_{23} is constructed at $t = 2$ with endpoints 3 and either 0 or 2. Take $T_1 = \{\{0,1\},\{0,2\},\{0,3\}\}$, $T_2 = \emptyset, T'_1 = \emptyset$ and $T'_2 = \{e_{12}, e_{13}, e_{23}\}$.
- In coalition $\{1,2\}$, agent 1 connects to the source at $t = 1$ and agent 2 connects to the source at $t = 2$. Then edge $\{0,1\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 2 and either 0 or 1. In coalition $\{1,3\}$, agent 3 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0,3\}$ is constructed at $t = 1$ and edge e_{13} is constructed at $t = 2$ with endpoints 1 and either 0 or 3. In coalition $\{2,3\}$, agent 3 connects to the source at $t = 1$ and agent 2 connects to the source at $t = 2$. Then edge $\{0,3\}$ is constructed at $t = 1$ and edge e_{23} is constructed at $t = 2$ with endpoints 2 and either 0 or 3. Take $T_1 = \{\{0,1\},\{0,3\}\}, T_2 = \{e_{12}\}, T'_1 = \{\{0,3\}\}$ and $T'_2 = \{e_{13}, e_{23}\}$.
- In coalition $\{1,2\}$, agent 2 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0,2\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 1 and either 0 or 2. In coalition $\{1,3\}$, agent 1 connects to the source at $t = 1$ and agent 3 connects to the source at $t = 2$. Then edge $\{0,1\}$ is constructed at $t = 1$ and edge e_{13} is constructed at $t = 2$ with endpoints 3 and either 0 or 1. In coalition $\{2,3\}$, agent 2 connects to the source at $t = 1$ and agent 3 connects to the source at $t = 2$. Then edge $\{0,2\}$ is constructed at $t = 1$ and edge e_{23} is constructed at $t = 2$ with endpoints 3 and either 0 or 2. Take $T_1 = \{\{0,2\}\}, T_2 = \{e_{12}, e_{23}\}, T'_1 = \{\{0,1\},\{0,2\}\}$ and $T'_2 = \{e_{13}\}$.
- In coalition $\{1,2\}$, agent 2 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0,2\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 1 and either 0 or 2. In coalition $\{1,3\}$, agent 1 connects to the source at $t = 1$ and agent 3 connects to the source at $t = 2$. Then edge $\{0,1\}$ is constructed at $t = 1$

and edge e_{13} is constructed at $t = 2$ with endpoints 3 and either 0 or 1. In coalition $\{2, 3\}$, agent 3 connects to the source at $t = 1$ and agent 2 connects to the source at $t = 2$. Then edge $\{0, 3\}$ is constructed at $t = 1$ and edge e_{23} is constructed at $t = 2$ with endpoints 2 and either 0 or 3. Take $T_1 = \{\{0, 1\}, \{0, 2\}, \{0, 3\}\}, T_2 = \emptyset, T_1' = \emptyset$ and $T_2' = \{e_{12}, e_{13}, e_{23}\}$.

- In coalition $\{1, 2\}$, agent 2 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0, 2\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 1 and either 0 or 2. In coalition $\{1, 3\}$, agent 3 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0, 3\}$ is constructed at $t = 1$ and edge e_{13} is constructed at $t = 2$ with endpoints 1 and either 0 or 3. In coalition $\{2, 3\}$, agent 2 connects to the source at $t = 1$ and agent 3 connects to the source at $t = 2$. Then edge $\{0, 2\}$ is constructed at $t = 1$ and edge e_{23} is constructed at $t = 2$ with endpoints 3 and either 0 or 2. Take $T_1 = \{\{0, 2\}\}, T_2 = \{e_{12}, e_{23}\}, T_1' = \{\{0, 2\}, \{0, 3\}\}$ and $T_2' = \{e_{13}\}$.
- In coalition $\{1, 2\}$, agent 2 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0, 2\}$ is constructed at $t = 1$ and edge e_{12} is constructed at $t = 2$ with endpoints 1 and either 0 or 2. In coalition $\{1, 3\}$, agent 3 connects to the source at $t = 1$ and agent 1 connects to the source at $t = 2$. Then edge $\{0, 3\}$ is constructed at $t = 1$ and edge e_{13} is constructed at $t = 2$ with endpoints 1 and either 0 or 3. In coalition $\{2, 3\}$, agent 3 connects to the source at $t = 1$ and agent 2 connects to the source at $t = 2$. Then edge $\{0, 3\}$ is constructed at $t = 1$ and edge e_{23} is constructed at $t = 2$ with endpoints 2 and either 0 or 3. Take $T_1 = \{\{0, 3\}\}, T_2 = \{e_{12}, e_{23}\}, T_1' = \{\{0, 2\}, \{0, 3\}\}$ and $T_2' = \{e_{13}\}$.

□

A.2 Proof of Theorem 5.4.2

Theorem 5.4.2. *Let $(N, 0, w_1, w_2, \alpha, r)$ be an mcsttr problem with $|N| = 3$. Then the following statements are true*

(i) If every agent connects to the source at $t = 2$ (1), then the vector x constructed by Algorithm 8 is an element of the core.

(ii) If one agent connects to the source at $t = 1$ (2), then the vector x constructed by Algorithm 8 is an element of the core.

(iii) If two agents connect to the source at $t = 1$ (3), then the vector x constructed by Algorithm 9 is an element of the core.

(iv) If all agents connect to the source at $t = 1$ (4), then the vector x constructed by Algorithm 10 is an element of the core.

Proof.

(i) By Theorem 4.2.1, we know the constructed mcstt by Algorithm 8 is minimal and $x_1 + x_2 + x_3 = c_t(N)$. We have $x_i \leq c_t(\{i\})$ for every $i \in \{1, 2, 3\}$ and $x_i + x_j \leq c_t(\{i, j\})$ for every pair of agents in $\{1, 2, 3\}$ since otherwise the constructed mcstt would not be minimal.

(ii) Without loss of generality suppose agent 1 connects to the source at $t = 1$ and agent 2 and 3 connect to the source at $t = 2$. We have $x_1 = w_1(\{0, 1\})$, $x_2 = w_2(e_2) + \alpha_2$ and $x_3 = w_2(e_3) + \alpha_3$ where e_2 is the edge constructed by agent 2 and e_3 is the edge constructed by agent 3. If the agents 2 and 3 both construct an edge either directly to the source or via agent 1, this statement is obviously true because then the additional costs of the agents 2 and 3 are less than r. The only case in which we can not imediately see that this statement is true is if agent 2 connects to the source through agent 3 and $\alpha_2 > r$ or if agent 3 connects to the source through agent 2 and $\alpha_3 > r$. Without loss of generality, suppose agent 2 connects to the source through agent 3 and $\alpha_2 > r$.

- $x_1 \leq c_t(\{1\}$:
  Because agent 1 is the only agent connecting to the source at $t = 1$ we have $x_1 \leq c_t(\{1\})$.

- $x_2 \leq c_t(\{2\})$:
  Because $\alpha_2 > r$ we have $c_t(\{2\}) = w_1(\{0,2\})$. Since $x_1 + x_2 + x_3 = c_t(N) \leq w_1(\{0,1\}) + w_1(\{0,2\}) + w_2(e_3) + \alpha_3 = x_1 + c_t(\{2\}) + x_3$ we have $x_2 \leq c_t(\{2\})$.
- $x_3 \leq c_t(\{3\})$:
  Because $\alpha_3 \leq r$ we have $c_t(\{3\}) = w_2(\{0,3\}) + \alpha_3$. By definition of the voccp we have $e_3 \leq w_2(\{0,3\})$ therefore $x_3 \leq c_t(\{3\})$.
- $x_1 + x_2 \leq c_t(\{1,2\})$:
  When we look at coalition $\{1,2\}$, agent 1 and agent 2 would connect to the source at $t = 1$. We have $x_1 + x_2 + x_3 = c_t(N) \leq c_t(\{1,2\}) + x_3$, therefore $x_1 + x_2 \leq c_t(\{1,2\})$.
- $x_1 + x_3 \leq c_t(\{1,3\})$:
  Because agent 2 connects to the source at $t = 2$ we have $x_1 + x_2 + x_3 = c_t(N) \leq c_t(\{1,3\}) + x_2$, so $x_1 + x_3 \leq c_t(\{1,3\})$.
- $x_2 + x_3 \leq c_t(\{2,3\})$:
  Because $x_1 + x_2 + x_3 = c_t(N) \leq w_1(\{0,1\}) + c_t(\{2,3\}) = x_1 + c_t(\{2,3\})$ we have $x_2 + x_3 \leq c_t(\{2,3\})$.

Hence, the allocation vector returned by Algorithm 8 is an element of the core of the associated mcsttr game.

(iii) Without loss of generality suppose agents 1 and 2 connect to the source at $t = 1$ and agent 3 connects to the source at $t = 2$. Also suppose without loss of generality that agent 1 constructs an edge in the second step and agent 2 constructs an edge in the third step of the algorithm. Then e_1 is the edge constructed by agent 3, e_2 is the edge constructed by agent 1 and e_3 is the edge constructed by agent 2. Algorithm 9 returns an mcstt and we have

$$\begin{aligned}
x_1 + x_2 + x_3 &= w_2(e_1) + \alpha_3 + x_1 + w_1(e_2) + w_1(e_3) - x_1 \\
&= w_2(e_1) + \alpha_3 + w_1(e_2) + w_1(e_3) \\
&= m(N, 0, w_1, w_2, \alpha) \\
&= c_t(N).
\end{aligned}$$

Per definition of the voccp and because $\alpha_3 < r$ we have

$$\begin{aligned}
x_3 &= w_2(e_1) + \alpha_3 \\
&\leq w_2(\{0,3\}) + \alpha_3
\end{aligned}$$

$$= c_t(\{3\}).$$

Furthermore we have

$$\begin{aligned} x_1 &= \min\{w_1(\{0,1\}), w_2(\{0,1\}) + \alpha_1, w_1(\{0,3\}) + w_1(\{1,3\}) - x_3, \\ &\quad w_2(\{0,3\}) + w_2(\{1,3\}) + \alpha_1 + \alpha_3 - x_3\} \\ &= \min\{c_t(\{1\}), w_1(\{0,3\}) + w_1(\{1,3\}) - x_3, \\ &\quad w_2(\{0,3\}) + w_2(\{1,3\}) + \alpha_1 + \alpha_3 - x_3\} \\ &\leq c_t(\{1\}) \end{aligned}$$

and

$$\begin{aligned} x_2 &= w_1(e_2) + w_1(e_3) - x_1 \\ &\leq w_1(\{0,1\}) + w_1(\{0,2\}) - x_1 \\ &= w_1(\{0,2\}) - \min\{0, r - \alpha_1, w_1(\{0,1\}) - w_1(\{0,3\}) - w_1(\{1,3\}) + x_3, \\ &\quad w_1(\{0,1\}) - w_2(\{0,3\}) - w_2(\{1,3\}) - \alpha_1 - \alpha_3 + x_3\} \\ &\leq w_1(\{0,2\}) \\ &= c_t(\{2\}) \end{aligned}$$

where the last inequality comes from the fact that agent 2 constructs an edge at $t = 1$ in the last step of the algorithm and therefore $\alpha_2 \geq r$.

Since $c_t(N) = w_1(e_2) + w_1(e_3) + w_2(e_1) + \alpha_3$ we have $c_t(\{1,2\}) = w_1(e_2) + w_1(e_3)$, since otherwise the constructed mcstt would not be minimal which is a contradiction. Therefore,

$$\begin{aligned} x_1 + x_2 &= c_t(N) - x_3 \\ &= w_1(e_2) + w_1(e_3) \\ &= c_t(\{1,2\}). \end{aligned}$$

There are six possibilities for agent 1 and agent 3 to cooperate, taking into account the fact that $\alpha_3 < r$. We therefore have

$$\begin{aligned} x_1 + x_3 &= \min\{w_1(\{0,1\}) + w_2(e_1) + \alpha_3, w_2(\{0,1\}) + w_2(e_1) + \alpha_1 + \alpha_3, \\ &\quad w_1(\{0,3\}) + w_1(\{1,3\}), w_2(\{0,3\}) + w_2(\{1,3\}) + \alpha_1 + \alpha_3\} \\ &\leq \min\{w_1(\{0,1\}) + w_2(\{1,3\}) + \alpha_3, w_1(\{0,1\}) + w_2(\{0,3\}) + \alpha_3, \\ &\quad w_2(\{0,1\}) + w_2(\{0,3\}) + \alpha_1 + \alpha_3, w_2(\{0,1\}) + w_2(\{1,3\}) + \alpha_1 + \alpha_3, \\ &\quad w_1(\{0,3\}) + w_1(\{1,3\}), w_2(\{0,3\}) + w_2(\{1,3\}) + \alpha_1 + \alpha_3\} \end{aligned}$$

$$= c_t(\{1,3\})$$

To prove $x_2 + x_3 \leq c_t(\{2,3\})$ we distinguish 2 cases: either agent 1 and agent 2 both construct an edge directly to the source or one of them connects to the source through other agent. In the first case we have $\alpha_1, \alpha_2 \geq r$. Therefore $x_1 = c_t(\{1\})$ and thus

$$\begin{aligned} x_2 + x_3 &= c_t(N) - x_1 \\ &= c_t(N) - c_t(\{1\}) \\ &\leq c_t(\{2,3\}) \end{aligned}$$

Let us now look at the second case in which we assume agent 2 connects to the source via agent 1 (the same holds when agent 1 connects to the source via agent 2). We have $e_2 = w_1(\{0,1\})$, $e_3 = w_1(\{1,2\})$, $\alpha_2 \geq r, \alpha_3 < r$ and either $\alpha_1 \geq r$ or $\alpha_1 < r$. Because $\alpha_2 \geq r$ and $\alpha_3 < r$ we have 4 possibilities for agent 2 and agent 3 to cooperate, therefore

$$\begin{aligned} c_t(\{2,3\}) = \min\{&w_1(\{0,2\}) + w_2(\{0,3\}) + \alpha_3, w_1(\{0,2\}) + w_2(\{2,3\}) + \alpha_3, \\ &w_1(\{0,3\}) + w_1(\{2,3\}), w_2(\{0,3\}) + w_2(\{2,3\}) + \alpha_2 + \alpha_3\} \end{aligned}$$

If $x_1 = c_t(\{1\})$, then the same holds as for in the first case. If $x_1 \neq c_t(\{1\})$, let us then consider the four different possibilities for $c_t(\{2,3\})$ seperately.

- $c_t(\{2,3\}) = w_1(\{0,2\}) + w_2(\{0,3\}) + \alpha_3$:
  If $x_1 = w_1(\{0,3\}) + w_1(\{1,3\}) - x_3$ then we have

  $$\begin{aligned} x_2 + x_3 &= c_t(N) - x_1 \\ &\leq w_1(\{0,3\}) + w_1(\{0,2\}) + w_1(\{1,3\}) - w_1(\{0,3\}) - w_1(\{1,3\}) + x_3 \\ &= w_1(\{0,2\}) + x_3 \\ &\leq c_t(\{2,3\}) \end{aligned}$$

  where the first inequality comes from the fact that $c_t(N)$ is minimal and the second inequality comes from the fact that $x_3 \leq w_2(\{0,3\}) + \alpha_3$. For the same reason, when $x_1 = w_2(\{0,3\}) + w_2(\{1,3\}) + \alpha_1 + \alpha_3 - x_3$ we have

  $$\begin{aligned} x_2 + x_3 &= c_t(N) - x_1 \\ &\leq w_1(\{0,2\}) + w_2(\{0,3\}) + w_2(\{1,3\}) + \alpha_1 + \alpha_3 - w_2(\{0,3\}) \\ &\quad - w_2(\{1,3\}) - \alpha_1 - \alpha_3 + x_3 \\ &= w_1(\{0,2\}) + x_3 \\ &\leq c_t(\{2,3\}) \end{aligned}$$

- $c_t(\{2,3\}) = w_1(\{0,2\}) + w_2(\{2,3\}) + \alpha_3$:
  Can be proved in the same way as for the first possibility.
- $c_t(\{2,3\}) = w_1(\{0,3\}) + w_1(\{2,3\})$:

  If $x_1 = w_1(\{0,3\}) + w_1(\{1,3\}) - x_3$ then we have $x_1 = w_1(\{1,3\}) + r - \alpha_3$ if $e_1 = \{0,3\}$, $x_1 = w_1(\{0,3\}) + r - \alpha_3$ if $e_1 = \{1,3\}$ and $w_1(\{0,3\}) + w_1(\{1,3\}) - w_2(\{2,3\}) - \alpha_3$ if $e_1 = \{2,3\}$. In the first case we have

$$\begin{aligned} x_2 + x_3 &= c_t(N) - w_1(\{1,3\}) - r + \alpha_3 \\ &\leq w_1(\{0,3\}) + w_1(\{1,3\}) + w_1(\{2,3\}) - w_1(\{1,3\}) - r + \alpha_3 \\ &= w_1(\{0,3\}) + w_1(\{2,3\}) - r + \alpha_3 \\ &\leq w_1(\{0,3\}) + w_1(\{2,3\}) \\ &= c_t(\{2,3\}) \end{aligned}$$

  where the first inequality comes from the fact that $c_t(N)$ is minimal and the second inequality comes from the fact that $\alpha_3 < r$. For the same reasons and because $w_1(\{1,3\}) \leq w_1(\{0,3\})$ when $e_1 = \{1,3\}$, in the second case we have

$$\begin{aligned} x_2 + x_3 &= c_t(N) - w_1(\{0,3\}) - r + \alpha_3 \\ &\leq w_1(\{0,3\}) + w_1(\{1,3\}) + w_1(\{2,3\}) - w_1(\{0,3\}) - r + \alpha_3 \\ &\leq w_1(\{0,3\}) + w_1(\{2,3\}) - r + \alpha_3 \\ &\leq w_1(\{0,3\}) + w_1(\{2,3\}) \\ &= c_t(\{2,3\}) \end{aligned}$$

  and in the third case we have

$$\begin{aligned} x_2 + x_3 &= c_t(N) - w_1(\{0,3\}) - w_1(\{1,3\}) + w_1(\{2,3\}) + \alpha_2 + \alpha_3 \\ &\leq w_1(\{1,3\}) + w_1(\{0,3\}) + w_1(\{2,3\}) \\ &\quad - w_1(\{0,3\}) - w_1(\{1,3\}) + w_2(\{2,3\}) + \alpha_3 \\ &\leq w_1(\{0,3\}) + w_1(\{2,3\}) - w_1(\{2,3\}) + w_2(\{2,3\}) + \alpha_3 \\ &= w_1(\{0,3\}) + w_1(\{2,3\}) - r + \alpha_3 \\ &\leq w_1(\{0,3\}) + w_1(\{2,3\}) \\ &= c_t(\{2,3\}) \end{aligned}$$

  where the first inequality is because $c_t(N)$ is minimal, the second inequality comes from the fact that $w_1(\{0,3\}) \geq w_1(\{2,3\})$, the second equality comes from the fact that $w_1(\{2,3\}) - w_2(\{2,3\}) = r$ and the third inequality is because $\alpha_3 < r$.

If $x_1 = w_2(\{0,3\}) + w_2(\{1,3\}) + \alpha_1 + \alpha_3$ then we have $x_1 = w_2(\{1,3\}) + \alpha_1$ if $e_1 = \{0,3\}$, $x_1 = w_2(\{0,3\}) + \alpha_1$ if $e_1 = \{1,3\}$ and $x_1 = w_2(\{0,3\}) + w_2(\{1,3\}) - w_2(\{2,3\}) + \alpha_1$ if $e_1 = \{2,3\}$. In the first case we have

$$\begin{aligned}
x_2 + x_3 &= c_t(N) - w_2(\{1,3\}) - \alpha_1 \\
&\leq w_1(\{0,3\}) + w_1(\{1,3\}) + w_1(\{2,3\}) - w_2(\{1,3\}) - \alpha_1 \\
&= w_1(\{0,3\}) + w_1(\{2,3\}) + r - \alpha_1 \\
&\leq w_1(\{0,3\}) + w_1(\{2,3\}) \\
&= c_t(\{2,3\})
\end{aligned}$$

In the second case we have

$$\begin{aligned}
x_2 + x_3 &= c_t(N) - w_2(\{0,3\}) - \alpha_1 \\
&\leq w_1(\{0,3\}) + w_1(\{1,3\}) + w_1(\{2,3\}) - w_2(\{0,3\}) - \alpha_1 \\
&= w_1(\{1,3\}) + w_1(\{2,3\}) + r - \alpha_1 \\
&\leq w_1(\{0,3\}) + w_1(\{2,3\}) + r - \alpha_1 \\
&\leq w_1(\{0,3\}) + w_1(\{2,3\}) \\
&= c_t(\{2,3\})
\end{aligned}$$

and in the third case we have

$$\begin{aligned}
x_2 + x_3 &= c_t(N) - w_2(\{0,3\}) - w_2(\{1,3\}) + w_2(\{2,3\}) - \alpha_1 \\
&\leq w_1(\{0,3\}) + w_1(\{1,3\}) + w_1(\{2,3\}) - w_2(\{0,3\}) \\
&\quad - w_2(\{1,3\}) + w_2(\{2,3\}) - \alpha_1 \\
&\leq w_1(\{0,3\}) + w_1(\{2,3\}) + r - w_2(\{2,3\}) + w_2(\{2,3\}) - \alpha_1 \\
&\leq w_1(\{0,3\}) + w_1(\{2,3\}) \\
&= c_t(\{2,3\})
\end{aligned}$$

where the second inequality comes from the fact that $w_1(\{1,3\}) - w_2(\{1,3\}) = r$ and $w_2(\{1,3\}) \geq w_2(\{2,3\})$ since in this case $e_1 = \{2,3\}$.

- $c_t(\{2,3\}) = w_2(\{0,3\}) + w_2(\{2,3\}) + \alpha_2 + \alpha_3$:

  If $x_1 = w_1(\{0,3\}) + w_1(\{1,3\}) - x_3$ then we have $x_1 = w_1(\{1,3\}) + r - \alpha_3$

if $e_1 = \{0,3\}$, $x_1 = w_1(\{0,3\}) + r - \alpha_3$ if $e_1 = \{1,3\}$ and $w_1(\{0,3\}) + w_1(\{1,3\}) - w_2(\{2,3\}) - \alpha_3$ if $e_1 = \{2,3\}$. In the first case we have

$$\begin{aligned}
x_2 + x_3 &= c_t(N) - w_1(\{1,3\}) - r + \alpha_3 \\
&\leq w_1(\{0,3\}) + w_1(\{1,3\}) + w_2(\{2,3\}) + \alpha_2 - w_1(\{1,3\}) \\
&\quad - r + \alpha_3 \\
&= w_2(\{0,3\}) + w_2(\{2,3\}) + \alpha_2 + \alpha_3 \\
&= c_t(\{2,3\})
\end{aligned}$$

where the second equality comes from the fact that $w_1(\{0,3\}) - r = w_2(\{0,3\})$. In the second case we have

$$\begin{aligned}
x_2 + x_3 &= c_t(N) - w_1(\{0,3\}) - r + \alpha_3 \\
&\leq w_1(\{0,3\}) + w_1(\{1,3\}) + w_2(\{2,3\}) + \alpha_2 - w_1(\{0,3\}) \\
&\quad - r + \alpha_3 \\
&= w_1(\{1,3\}) + w_2(\{2,3\}) + \alpha_2 - r + \alpha_3 \\
&= w_2(\{1,3\}) + w_2(\{2,3\}) + \alpha_2 + \alpha_3 \\
&\leq w_2(\{0,3\}) + w_2(\{2,3\}) + \alpha_2 + \alpha_3 \\
&= c_t(\{2,3\})
\end{aligned}$$

and in the third case we have

$$\begin{aligned}
x_2 + x_3 &= c_t(N) - w_1(\{0,3\}) - w_1(\{1,3\}) + w_2(\{2,3\}) + \alpha_3 \\
&\leq w_1(\{0,3\}) + w_1(\{1,3\}) + w_2(\{2,3\}) + \alpha_2 - w_1(\{0,3\}) \\
&\quad - w_1(\{1,3\}) + w_2(\{2,3\}) + \alpha_3 \\
&= w_2(\{2,3\}) + w_2(\{2,3\}) + \alpha_2 + \alpha_3 \\
&\leq w_2(\{0,3\}) + w_2(\{2,3\}) + \alpha_2 + \alpha_3 \\
&= c_t(\{2,3\})
\end{aligned}$$

If $x_1 = w_2(\{0,3\}) + w_2(\{1,3\}) + \alpha_1 + \alpha_3$ then we have $x_1 = w_2(\{1,3\}) + \alpha_1$ if $e_1 = \{0,3\}$, $x_1 = w_2(\{0,3\}) + \alpha_1$ if $e_1 = \{1,3\}$ and $x_1 = w_2(\{0,3\}) + w_2(\{1,3\}) - w_2(\{2,3\}) + \alpha_1$ if $e_1 = \{2,3\}$. In the first case we have

$$\begin{aligned}
x_2 + x_3 &= c_t(N) - w_2(\{1,3\}) - \alpha_1 \\
&\leq w_2(\{0,3\}) + w_2(\{1,3\}) + w_2(\{2,3\}) + \alpha_1 + \alpha_2 + \alpha_3 \\
&\quad - w_2(\{1,3\}) - \alpha_1
\end{aligned}$$

$$\begin{aligned} &= w_2(\{0,3\}) + w_2(\{2,3\}) + \alpha_2 + \alpha_3 \\ &= c_t(\{2,3\}) \end{aligned}$$

In the second case we have

$$\begin{aligned} x_2 + x_3 &= c_t(N) - w_2(\{0,3\}) - \alpha_1 \\ &\leq w_2(\{0,3\}) + w_2(\{1,3\}) + w_2(\{2,3\}) + \alpha_1 + \alpha_2 + \alpha_3 - w_2(\{0,3\}) - \alpha_1 \\ &= w_2(\{1,3\}) + w_2(\{2,3\}) + \alpha_2 + \alpha_3 \\ &\leq w_2(\{0,3\}) + w_2(\{2,3\}) + \alpha_2 + \alpha_3 \\ &= c_t(\{2,3\}) \end{aligned}$$

and in the third case we have

$$\begin{aligned} x_2 + x_3 &= c_t(N) - w_2(\{0,3\}) - w_2(\{1,3\}) + w_2(\{2,3\}) - \alpha_1 \\ &\leq w_2(\{0,3\}) + w_2(\{1,3\}) + w_2(\{2,3\}) + \alpha_1 + \alpha_2 + \alpha_3 \\ &\quad - w_2(\{0,3\}) - w_2(\{1,3\}) + w_2(\{2,3\}) - \alpha_1 \\ &= w_2(\{2,3\}) + w_2(\{2,3\}) + \alpha_2 + \alpha_3 \\ &\leq w_2(\{0,3\}) + w_2(\{2,3\}) + \alpha_2 + \alpha_3 \\ &= c_t(\{2,3\}) \end{aligned}$$

Hence, x is an element of the core of the associated mcsttr game.

(iv) Without loss of generality there are 4 possibilities in which the agents can connect to the source: (1) all agents connect to the source by a direct edge, (2) agent 1 and agent 3 connect to the source by a direct edge and agent 2 connects to the source through agent 1, (3) agent 1 connects to the source by a direct edge, agent 2 connects to the source through agent 1 and agent 3 connects to the source through agent 2 and (4) agent 1 connects directly to the source and agent 2 and agent 3 both connect to the source through agent 1, see figure A.2.1. Let us consider every case seperately

(1) Because every agent is connected to the source by a direct edge we have $\alpha_i \geq r$ for $i \in \{1,2,3\}$. Therefore, $x_1 = w_1(\{0,1\}) = c_t(\{1\}), x_2 = w_1(\{0,2\}) = c_t(\{2\})$ and $x_3 = w_1(\{0,3\}) = c_t(\{3\})$. Also $x_1 + x_2 = c_t(\{1,2\}), x_1+x_3 = c_t(\{1,3\}), x_2+x_3 = c_t(\{2,3\})$ and $x_1+x_2+x_3 = c_t(N)$ by definition of Prim's algorithm.

(2) Let $E = (e_1, e_2, e_3)$ be the sequence of edges constructed by Algorithm 10. Algorithm 10 can construct this mcstt in the following three ways:

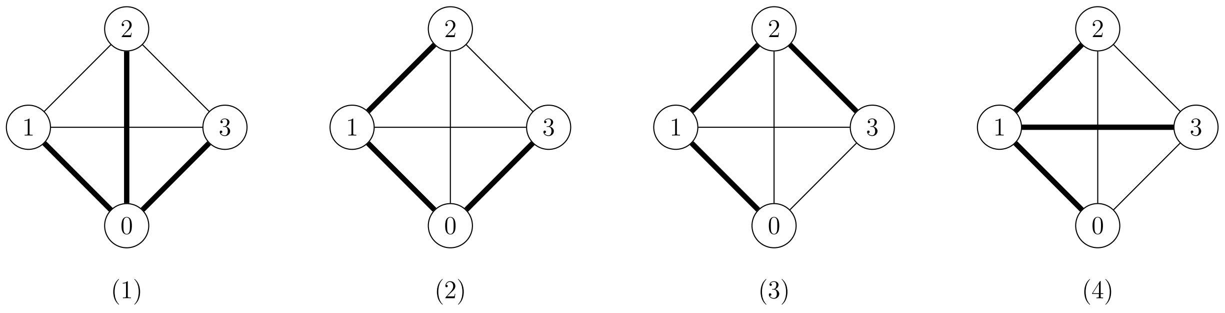


Figure A.2.1: Shows the 4 possibilities for the three agents to all connect to the source in the mcsttr problem at $t = 1$.

$E = (\{0,1\},\{0,3\},\{1,2\}), E = (\{0,1\},\{1,2\},\{0,3\})$ or $E = (\{0,3\},\{0,1\},\{1,2\})$. Notice that $\alpha_2, \alpha_3 \geq r$. In the first case we have

$$x_1 = c_t(\{1\})$$

per definition and

$$\begin{aligned} x_3 &= w_1(\{0,3\}) \\ &= c_t(\{3\}) \end{aligned}$$

because $\alpha_3 \geq r$. Because of the subadditivity of mcstt games, we have

$$\begin{aligned} x_2 &= c_t(N) - x_1 - x_3 \\ &= c_t(N) - c_t(\{1\}) - c_t(\{3\}) \\ &\leq c_t(\{2\}) \end{aligned}$$

$$\begin{aligned} x_1 + x_2 &= c_t(N) - x_3 \\ &= c_t(N) - c_t(\{3\}) \\ &\leq c_t(\{1,2\}) \end{aligned}$$

$$\begin{aligned} x_2 + x_3 &= c_t(N) - x_1 \\ &= c_t(N) - c_t(\{1\}) \\ &\leq c_t(\{2,3\}). \end{aligned}$$

Furthermore, because $w_1(\{1,3\}) \geq \min\{w_1(\{0,1\}), w_1(\{0,3\})$ and $\alpha_3 \geq r$ we have

$$\begin{aligned} c_t(\{1,3\}) &= \min\{w_1(\{0,1\}) + w_1(\{0,3\}), w_2(\{0,1\}) + \alpha_1 + w_1(\{0,3\})\} \\ &= c_t(\{1\}) + w_1(\{0,3\}) \\ &= x_1 + x_3. \end{aligned}$$

In the second case we know $c_t(\{1,2\}) = w_1(\{0,1\}) + w_1(\{1,2\})$ because if there would be a cheaper way, the mcsttr outputted by Algorithm 10 would not be minimal. Therefore

$$x_1 + x_2 = c_t(\{1,2\})$$

Again, we have

$$x_1 = c_t(\{1\})$$

per definition. By subadditivity of mcstt games we have

$$\begin{aligned} x_3 &= c_t(N) - x_1 - x_2 \\ &= c_t(N) - c_t(\{1,2\}) \\ &\leq c_t(\{3\}) \end{aligned}$$

$$\begin{aligned} x_2 + x_3 &= c_t(N) - x_1 \\ &= c_t(N) - c_t(\{1\}) \\ &\leq c_t(\{2,3\}). \end{aligned}$$

Because $c_t(\{1,2\}) = w_1(e_1) + w_1(e_2)$ we have

$$\begin{aligned} x_2 &= c_t(\{1,2\}) - x_1 \\ &\leq c_t(\{2\}) \end{aligned}$$

where the inequality follows from the subadditivity of mcstt games. Furthermore, $w_1(\{1,3\}) \geq \min\{w_1(\{0,3\}), w_1(\{0,1\})\}$ since otherwise the constructed mcstt would not be minimal. Therefore we have

$$\begin{aligned} x_1 + x_3 &= w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{0,3\}) - w_1(\{0,1\}) \\ &\quad - w_1(\{1,2\}) + \min\{w_1(\{0,1\}), w_2(\{0,1\}) + \alpha_1\} \\ &= w_1(\{0,3\}) + \min\{w_1(\{0,1\}), w_2(\{0,1\}) + \alpha_1\} \end{aligned}$$

$$= c_t(\{1,3\})$$

In the third case we have
$$x_3 = c_t(\{3\})$$
per definition of Algorithm 10. Since $\alpha_3 \geq r$ we have

$$\begin{aligned} x_1 &= \min\{w_1(\{0,3\}) + w_1(\{0,1\}), w_1(\{0,3\}) + w_2(\{0,1\}) + \alpha_1, \\ &\quad w_2(\{0,3\}) + w_2(\{0,1\}) + \alpha_1 + \alpha_3\} - w_1(\{0,3\}) \\ &= \min\{w_1(\{0,1\}), w_2(\{0,1\}) + \alpha_1\} \\ &= c_t(\{1\}). \end{aligned}$$

Because $w_1(\{1,3\}) \geq \min\{w_1(\{0,1\}), w_1(\{0,3\})$ since otherwise the constructed mcstt would not be minimal and $\alpha_3 \geq r$, we have

$$x_1 + x_3 = c_t(\{1,3\}).$$

By subadditivity of mcstt games we have

$$\begin{aligned} x_2 &= c_t(N) - x_1 - x_3 \\ &= c_t(N) - c_t(\{1\}) - c_t(\{3\}) \\ &\leq c_t(\{2\}) \end{aligned}$$

$$\begin{aligned} x_1 + x_2 &= c_t(N) - x_3 \\ &= c_t(N) - c_t(\{3\}) \\ &\leq c_t(\{1,2\}) \end{aligned}$$

$$\begin{aligned} x_2 + x_3 &= c_t(N) - x_1 \\ &= c_t(N) - c_t(\{1\}) \\ &\leq c_t(\{2,3\}). \end{aligned}$$

(3) Notice that $\alpha_3 \geq r$. Per definition of Algorithm 10 we have $x_1 = c_t(\{1\})$ and therefore $x_2 + x_3 = c_t(N) - x_1 = c_t(N) - c_t(\{1\}) \leq c_t(\{2,3\})$ by subadditivity. Also $x_1 + x_2 + x_3 = w_1(e_1) + w_1(e_2) + w_1(e_3) = c_t(N)$. We can distinguish 4 cases: $\alpha_1, \alpha_2 \geq r, \alpha_1 \geq r$ and $\alpha_2 < r$, $\alpha_1 < r$ and $\alpha_2 \geq r$ and $\alpha_1, \alpha_2 < r$. Let us consider each case seperately:

- $\alpha_1, \alpha_2 \geq r$:

- $x_2 \leq c_t(\{2\})$:
  We have $x_1 = w_1(\{0,1\}) = c_t(\{1\})$. Therefore

$$\begin{aligned} x_2 &= \min\{w_1(\{1,2\}), w_2(\{1,2\}) + \alpha_2, w_2(\{1,2\}) + \alpha_2 + \alpha_1 - r, \\ &\quad w_1(\{0,2\}) + \alpha_1 - r\} \\ &= w_1(\{1,2\}) \\ &\leq w_1(\{0,1\}) \\ &= c_t(\{2\}) \end{aligned}$$

  since $\alpha_1, \alpha_2 \geq r$ and $w_1(\{0,1\}) \geq w_1(\{1,2\})$ by definition of Prim's algorithm.

- $x_3 \leq c_t(\{3\})$:
  We have

$$\begin{aligned} x_3 &= c_t(N) - x_1 - x_2 \\ &= w_1(\{2,3\}) \\ &\leq w_1(\{0,3\}) \\ &= c_t(\{3\}) \end{aligned}$$

  since if $w_1(\{2,3\}) > w_1(\{0,3\})$ the mcstt constructed by Algorithm 10 would not be minimal and $\alpha_3 \geq r$.

- $x_1 + x_2 \leq c_t(\{1,2\})$:
  $w_1(\{0,1\}), w_1(\{1,2\}) \geq w_1(\{0,2\})$ since otherwise the constructed mcstt would not be minimal. Because $\alpha_1, \alpha_2 \geq r$ we now have $x_1 + x_2 = c_t(\{1,2\})$.

- $x_1 + x_3 \leq c_t(\{1,3\})$:
  We have $x_1 + x_3 = w_1(\{0,1\}) + w_1(\{2,3\})$. Since $w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{2,3\}) \leq w_1(\{0,3\}) + w_1(\{1,2\}) + w_1(\{2,3\})$ we have $w_1(\{0,1\}) \leq w_1(\{0,3\})$. In the same way we have $w_1(\{2,3\}) \leq \min\{w_1(\{1,3\}), w_1(\{1,3\})\}$. Therefore

$$\begin{aligned} x_1 + x_3 &= w_1(\{0,1\}) + w_1(\{2,3\}) \\ &= \min\{w_1(\{0,1\}) + w_1(\{2,3\}), w_1(\{0,1\}) + w_1(\{1,3\}), \\ &\quad w_1(\{0,3\}) + w_1(\{1,3\})\} \end{aligned}$$

$$= c_t(\{1,3\})$$

since $\alpha_1, \alpha_3 \geq r$.

- $\alpha_1 \geq r, \alpha_2 < r$:
  - $x_2 \leq c_t(\{2\})$:
    Because $\alpha_1 \geq r, \alpha_2 < r$ and $w_1(\{1,2\}) \leq w_1(\{0,2\})$ by definition of Prim's algorithm, we have

$$\begin{aligned} x_2 = \min\{&w_1(\{1,2\}), w_2(\{1,2\}) + \alpha_2\, w_2(\{1,2\}) + \alpha_2 + \alpha_1 - r, \\ &w_1(\{0,2\}) + \alpha_1 - r\} \\ = {}& w_2(\{1,2\}) + \alpha_2 \\ \leq {}& w_1(\{0,1\}) \\ = {}& c_t(\{2,3\}) \end{aligned}$$

  - $x_3 \leq c_t(\{3\})$:
    We have

$$= w_1(\{0,1\}) + r - \alpha_2$$

Since $x_1+x_2+x_3 = w_1(\{0,1\})+w_1(\{1,2\})+w_1(\{1,3\}) = c_t(N) \leq w_1(\{0,1\}) + w_2(\{1,2\}) + \alpha_2 + w_1(\{0,3\})$ we have

$$x_3 \leq w_1(\{0,3\}) = c_t(\{3\})$$

  - $x_1 + x_2 \leq c_t(\{1,2\})$:
    Again $w_1(\{0,1\}), w_1(\{1,2\}) \geq w_1(\{0,2\})$. Because $\alpha_1 \geq r$ and $\alpha_2 < r$ we have $c_t(\{1,2\}) = w_1(\{0,1\})+w_2(\{1,2\})+\alpha_2 = x_1+x_2$.

  - $x_1 + x_3 \leq c_t(\{1,3\})$:
    Because $\alpha_1, \alpha_3 \geq r$, looking at coalition $\{1,3\}$ both agents connect to the source at $t = 1$. Therefore $x_1 + x_2 + x_3 = c_t(N) \leq c_t(\{1,3\}) + w_2(\{1,2\}) + \alpha_2 = c_t(\{1,3\}) + x_2$ and thus $x_1 + x_3 \leq c_t(\{1,3\})$.

- $\alpha_1 < r, \alpha_2 \geq r$:

- $x_2 \leq c_t(\{2\})$:
  We have $x_1 = w_2(\{0,1\}) + \alpha_1 = c_t(\{1\})$ and $\alpha_2 \geq r$. Therefore

$$\begin{aligned} x_2 &= \min\{w_1(\{1,2\}) + r - \alpha_1, w_2(\{1,2\}) + \alpha_2 + r - \alpha_1, \\ &\quad w_2(\{1,2\}) + \alpha_2, w_1(\{0,2\})\} \\ &\leq w_1(\{0,2\}) \\ &= c_t(\{2\}) \end{aligned}$$

- $x_3 \leq c_t(\{3\})$:
  We have

$$\begin{aligned} x_3 &= w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{2,3\}) - x_1 - x_2 \\ &= w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{2,3\}) - w_2(\{0,1\}) - \alpha_1 \\ &\quad - \min\{w_1(\{1,2\}) + r - \alpha_1, w_2(\{1,2\}) + \alpha_2, w_1(\{0,2\})\} \\ &= w_1(\{2,3\}) + r - \alpha_1 + \max\{\alpha_1 - r, r - \alpha_2 - w_1(\{0,2\})\} \\ &= \max\{w_1(\{2,3\}), w_1(\{2,3\}) + 2r - \alpha_1 - \alpha_2, \\ &\quad w_1(\{2,3\}) + w_1(\{1,2\}) - w_1(\{0,2\}) - \alpha_1\} \end{aligned}$$

  If $x_3 = w_1(\{2,3\})$ then $x_3 \leq w_1(\{0,3\}) = c_t(\{3\})$ since $w_1(\{2,3\}) \leq w_1(\{0,3\})$ by definition of Prim's algorithm.

  If $x_3 = w_1(\{2,3\}) + 2r - \alpha_1 - \alpha_2$ then $w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{2,3\}) = c_t(N) \leq w_2(\{0,1\}) + \alpha_1 + w_2(\{1,2\}) + \alpha_2 + w_1(\{0,3\})$. Therefore $w_1(\{2,3\}) \leq -2r + \alpha_1 + \alpha_2 + w_1(\{0,3\})$ and thus $w_1(\{2,3\}) + 2r - \alpha_1 - \alpha_2 \leq w_1(\{0,3\}) = c_t(\{3\})$.

  If $x_3 = w_1(\{2,3\}) + w_1(\{1,2\}) - w_1(\{0,1\}) - \alpha_1$ then $w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{2,3\}) = c_t(N) \leq w_1(\{0,2\}) + w_1(\{0,3\}) + w_2(\{0,1\}) + \alpha_1$ implies $w_1(\{2,3\}) + w_1(\{1,2\}) - w_1(\{0,2\}) - \alpha_1 \leq w_1(\{0,3\}) = c_t(\{3\})$
  So $x_3 \leq c_t(\{3\})$.

- $x_1 + x_2 \leq c_t(\{1,2\})$:
  Again $w_1(\{0,1\}), w_1(\{1,2\}) \geq w_1(\{0,2\})$. Because $\alpha_1 < r$ and $\alpha_2 \geq r$ we have

$$\begin{aligned} c_t(\{1,2\}) = \min\{&w_1(\{0,1\}) + w_1(\{1,2\}), w_2(\{0,1\}) + \alpha_1 \\ &+ w_2(\{1,2\}) + \alpha_2, w_2(\{0,1\}) + \alpha_1 + w_1(\{0,2\})\} \end{aligned}$$

So

$$\begin{aligned}
x_1 + x_2 &= \min\{w_1(\{0,1\}) + w_1(\{1,2\}), w_1(\{0,1\}) + w_2(\{1,2\}) \\
&\quad + \alpha_2, w_2(\{0,1\}) + w_2(\{1,2\}) + \alpha_1 + \alpha_2, w_1(\{0,2\}) + w_2(\{0,1\}) + \alpha_1\} \\
&= \min\{w_1(\{0,1\}) + w_1(\{1,2\}), w_1(\{0,1\}) + w_2(\{1,2\}) \\
&\quad + \alpha_2, w_1(\{0,2\}) + w_2(\{0,1\}) + \alpha_1\} \\
&= c_t(\{1,2\})
\end{aligned}$$

- $x_1 + x_3 \leq c_t(\{1,3\})$:
We have

$$\begin{aligned}
c_t(\{1,3\}) = \min\{&w_1(\{0,1\}) + w_1(\{1,3\}), w_2(\{0,1\}) + \alpha_1 \\
&+ w_2(\{1,3\}) + \alpha_3, w_1(\{0,3\}) + w_2(\{0,1\}) + \alpha_1, \\
&w_1(\{0,3\}) + w_2(\{1,3\}) + \alpha_1\}
\end{aligned}$$

since in all other possibilities, at least one of the agents would rather connect to the source at another point in time. Furthermore we have

$$\begin{aligned}
x_1 + x_3 = \max\{&w_2(\{0,1\}) + \alpha_1 + w_1(\{2,3\}), \\
&w_1(\{0,1\}) + w_1(\{2,3\}) + r - \alpha_2\}
\end{aligned}$$

Because $\alpha_1 < r$ we have $w_2(\{0,1\})+\alpha_1 < w_1(\{0,1\})$ and by definition of Prim's algorithm we have $w_1(\{2,3\}) \leq \min\{w_1(\{0,3\}), w_1$ $(\{1,3\})$. Also we have $w_1(\{2,3\}) \leq w_1(\{1,3\}) \leq w_2(\{1,3\}) + \alpha_3$. Furthermore we have $w_i(\{0,1\}) \leq w_i(\{0,3\})$, $i \in \{1,2\}$, by definition of Prim's algorithm. Therefore $w_2(\{0,1\}) + w_2(\{2,3\}) \leq w_2(\{0,3\})+w_2(\{1,3\})$ and thus $w_2(\{0,1\})+w_1(\{2,3\}) \leq w_1(\{0,3\})$ $+ w_2(\{1,3\})$. Hence

$$w_2(\{0,1\}) + \alpha_1 + w_1(\{2,3\}) \leq c_t(\{1,3\})$$

Because $\alpha_2 \geq r$ we have $w_1(\{2,3\})+r-\alpha_2 \leq w_1(\{1,3\})$. Furthermore we have $w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{2,3\}) \leq w_2(\{0,1\}) + \alpha_1 + w_2(\{1,3\}) + \alpha_3 + w_2(\{1,2\}) + \alpha_2$ which implies $w_1(\{0,1\}) + w_1(\{2,3\}) + r - \alpha_2 \leq w_2(\{0,1\}) + \alpha_1 + w_2(\{1,3\}) + \alpha_3$. We have $w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{2,3\}) \leq w_1(\{0,3\}) + w_2(\{0,1\}) + \alpha_1+w_2(\{1,2\})+\alpha_2$ which implies $w_1(\{0,1\})+w_1(\{2,3\})+r-\alpha_2 \leq w_1(\{0,3\})+w_2(\{0,1\})+\alpha_1$ and we have $w_1(\{0,1\})+w_1(\{1,2\})+$

$w_1(\{2,3\}) \leq w_1(\{0,3\}) + w_2(\{1,3\}) + \alpha_1 + w_2(\{1,2\}) + \alpha_2$ which implies $w_1(\{0,1\}) + w_1(\{2,3\}) + r - \alpha_2 \leq w_1(\{0,3\}) + w_2(\{1,3\}) + \alpha_1$. Therefore

$$w_1(\{0,1\}) + w_1(\{2,3\}) + r - \alpha_2 \leq c_t(\{1,3\}$$

and thus $x_1 + x_3 \leq c_t(\{1,3\})$.

- $\alpha_1, \alpha_2 < r$

  - $x_2 \leq c_t(\{2\})$
    We have

    $$x_2 = \min\{w_1(\{1,2\}) + r - \alpha_1, w_2(\{1,2\}) + \alpha_2\}$$

    Because $\alpha_2 < r$ we have $w_2(\{1,2\}) + \alpha_2 < w_1(\{1,2\})$ and since $\alpha_1 < r$ we have $w_2(\{1,2\}) + \alpha_2 < w_1(\{1,2\}) + r - \alpha_1$. Because $w_2(\{1,2\}) \leq w_2(\{0,2\})$ since otherwise the mcstt constructed by Algorithm 9 would not be minimal, we now have

    $$x_2 = w_2(\{1,2\}) + \alpha_2 \leq w_2(\{0,2\}) + \alpha_2 = c_t(\{2\})$$

  - $x_3 \leq c_t(\{3\})$:
    We have

    $$\begin{aligned} x_3 &= c_t(N) - x_1 - x_2 \\ &= w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{2,3\}) - w_2(\{0,1\}) - \alpha_1 \\ &\quad - w_2(\{0,2\}) - \alpha_2 \\ &= w_1(\{2,3\}) + 2r - \alpha_1 - \alpha_2 \end{aligned}$$

    Because $w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{2,3\}) = c_t(N) \leq w_2(\{0,1\}) + \alpha_1 + w_2(\{1,2\}) + \alpha_2 + w_1(\{0,3\})$ we have $w_1(\{2,3\}) + 2r - \alpha_1 - \alpha_2 \leq c_t(\{0,3\}) = c_t(\{3\})$.

  - $x_1 + x_2 \leq c_t(\{1,2\})$:
    Because $\alpha_1, \alpha_2 < r$ and $w_2(\{0,2\}) \geq \max\{w_2(\{0,1\}), w_2(\{1,2\}\}$ by definition of Prim's algorithm we have

    $$c_t(\{1,2\}) = w_2(\{0,1\}) + \alpha_1 + w_2(\{1,2\}) + \alpha_2 = x_1 + x_2$$

- $x_1 + x_3 \leq c_t(\{1,3\})$:
  We have

$$\begin{aligned} x_1 + x_3 &= w_2(\{0,1\}) + \alpha_1 + w_1(\{2,3\}) + 2r - \alpha_1 - \alpha_2 \\ &= w_1(\{0,1\}) + w_1(\{2,3\}) + r - \alpha_2 \end{aligned}$$

  and

$$\begin{aligned} c_t(\{1,3\}) = \min\{&w_1(\{0,1\}) + w_1(\{1,3\}), w_2(\{0,1\}) + \alpha_1 \\ &+ w_2(\{1,3\}) + \alpha_3, w_1(\{0,3\}) + w_2(\{0,1\}) + \alpha_1, \\ &w_1(\{0,3\}) + w_2(\{1,3\}) + \alpha_1\} \end{aligned}$$

  In the previous case we saw that $x_1 + x_3 \leq c_t(\{1,3\})$.

(4) Without loss of generality suppose $w_1(\{1,2\}) \geq w_1(\{1,3\})$. Notice that $\alpha_2, \alpha_3 \geq r$. Per definition of Algorithm 10 we have $x_1 = c_t(\{1\})$ and therefore $x_2 + x_3 = c_t(N) - x_1 = c_t(N) - c_t(\{1\}) \leq c_t(\{2,3\})$ by subadditivity. We can distinguish 2 cases: $\alpha_1 \geq r$ and $\alpha_1 < r$. Let us consider each case seperately:

- $\alpha_1 \geq r$:

  - $x_2 \leq c_t(\{2\})$ and $x_1 + x_2 \leq c_t(\{1,2\})$:
    Because $\alpha_1, \alpha_2 > r$ we have

$$x_2 = w_1(\{1,2\})$$

    and because also Prim's algorithm chooses $\{0,1\}$ and $\{1,2\}$ instead of $\{0,2\}$ we know $w_1(\{0,2\}) \geq \max\{w_1(\{0,1\}), w_1(\{1,2\})\}$ and so

$$x_2 \leq c_t(\{2\})$$

    and

$$c_t(\{1,2\}) = w_1(\{0,1\}) + w_1(\{1,2\}) = x_1 + x_2$$

  - $x_3 \leq c_t(\{3\})$:
    We have

$$\begin{aligned} x_3 &= c_t(N) - w_1(\{0,1\}) - w_1(\{1,2\}) \\ &= w_1(\{1,3\}) \\ &\leq w_1(\{0,3\}) \end{aligned}$$

$$= c_t(\{3\})$$

where the inequality comes from the fact that Prim's algorithm chooses to construct edge $\{1,3\}$ instead of edge $\{0,3\}$.

- $x_1 + x_3 \leq c_t(\{1,3\})$:
  We have
  $$x_1 + x_3 = w_1(\{0,1\}) + w_1(\{1,3\})$$
  and
  $$c_t(\{1,3\}) = \min\{w_1(\{0,1\}) + w_1(\{0,3\}), w_1(\{0,1\}) + w_1(\{1,3\}), w_1(\{0,3\}) + w_1(\{1,3\})\}$$
  since $\alpha_1, \alpha_3 \geq r$. Because Prim's algorithm first constructs the edge $\{0,1\}$ instead of $\{0,3\}$ we have $w_1(\{0,1\}) \geq w_1(\{0,3\})$ and because Prim's algorithm constructs the edge $\{1,3\}$ instead of $\{0,3\}$ we have $w_1(\{1,3\}) \leq w_1(\{1,3\})$. Therefore
  $$x_1 + x_3 \leq c_t(\{1,3\})$$

- $\alpha_1 < r$:

  - $x_2 \leq c_t(\{2\})$:
    We have $x_1 = w_2(\{0,1\}) + \alpha_1$. Therefore
    $$\begin{aligned} x_2 &= \min\{w_1(\{1,2\}) + r - \alpha_1, w_2(\{1,2\}) + \alpha_2 + r - \alpha_1, \\ &\quad w_2(\{1,2\}) + \alpha_2, w_1(\{0,2\})) \\ &\leq w_1(\{0,2\}) \\ &= c_t(\{2\}) \end{aligned}$$

  - $x_3 \leq c_t(\{3\})$:
    We have
    $$\begin{aligned} x_3 &= w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{1,3\}) - \min\{w_1(\{0,1\}) \\ &\quad + w_1(\{1,2\}), w_1(\{0,1\}) + w_2(\{1,2\}) + \alpha_2, w_2(\{0,1\}) \\ &\quad + w_2(\{1,2\}) + \alpha_1 + \alpha_2, w_2(\{0,1\}) + \alpha_1 + w_1(\{0,2\})\} \\ &= \max\{w_1(\{1,3\}), w_1(\{1,3\}) + r - \alpha_2, w_1(\{1,3\}) + 2r \\ &\quad - \alpha_1 - \alpha_2, w_1(\{1,3\}) + w_1(\{1,2\}) - w_1(\{0,2\}) + r - \alpha_1\} \\ &= \max\{w_1(\{1,3\}), w_1(\{1,3\}) + 2r - \alpha_1 - \alpha_2, w_1(\{1,3\}) \end{aligned}$$

$$+ w_1(\{1,2\}) - w_1(\{0,2\}) + r - \alpha_1\}$$

We have

$$w_1(\{1,3\}) \leq w_1(\{0,3\})$$

since Prim's algorithm constructs edge $\{1,3\}$ instead of edge $\{0,3\}$.
We have

$$\begin{aligned} c_t(N) &= w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{1,3\}) \\ &\leq w_1(\{0,3\}) + w_2(\{0,1\}) + \alpha_1 + w_2(\{1,2\}) + \alpha_2 \end{aligned}$$

Therefore

$$w_1(\{1,3\}) + 2r - \alpha_1 - \alpha_2 \leq w_1(\{0,3\})$$

Also

$$\begin{aligned} c_t(N) &= w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{1,3\}) \\ &\leq w_1(\{0,2\}) + w_2(\{0,1\})\alpha_1 + w_1(\{0,3\}) \end{aligned}$$

Therefore

$$w_1(\{1,3\}) + w_1(\{1,2\}) - w_1(\{0,2\}) + r - \alpha_1 \leq w_1(\{0,3\})$$

Since $\alpha_3 \geq r$ we have $c_t(\{3\}) = w_1(\{0,3\})$. Hence $x_3 \leq c_t(\{3\})$.

- $x_1 + x_2 \leq c_t(\{1,2\})$:
We have

$$\begin{aligned} x_1 + x_2 &= \min\{w_1(\{0,1\}) + w_1(\{1,2\}), w_1(\{0,1\}) + w_2(\{1,2\}) \\ &\quad + \alpha_2, w_2(\{0,1\}) + \alpha_1 + w_2(\{1,2\}) + \alpha_2, w_1(\{0,2\}) \\ &\quad + w_2(\{0,1\}) + \alpha_1\} \\ &= \min\{w_1(\{0,1\}) + w_1(\{1,2\}), w_2(\{0,1\}) + \alpha_1 \\ &\quad + w_2(\{1,2\}) + \alpha_2, w_2(\{0,1\}) + \alpha_1 + w_1(\{0,2\})\} \end{aligned}$$

Since $\alpha_1 < r$ and $\alpha_2 \geq r$ we have

$$\begin{aligned} c_t(\{1,2\}) = \min\{&w_1(\{0,1\}) + w_1(\{1,2\}), w_2(\{0,1\}) + \alpha_1 \\ &+ w_2(\{1,2\}) + \alpha_2, w_1(\{0,2\}) + w_2(\{0,1\}) + \alpha_1\} \end{aligned}$$

Notice that $c_t(\{1,2\}) \geq w_1(\{0,2\}) + w_2(\{1,2\}) + \alpha_1$ since $w_1(\{0,2\}) \geq w_1(\{0,1\}) \geq w_2(\{0,1\}) + \alpha_2$. Therefore

$$x_1 + x_2 = c_t(\{1,2\})$$

.

- $x_1 + x_3 \leq c_t(\{1,3\})$:
  We have

$$\begin{aligned} x_1 + x_3 = \max\{&w_1(\{1,3\}) + w_2(\{0,1\}) + \alpha_1, \\ &w_1(\{0,1\}) + w_1(\{1,3\}) + r - \alpha_2, \\ &w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{1,3\}) - w_1(\{0,2\})\} \end{aligned}$$

  Since $\alpha_1 < r$ and $\alpha_3 \geq r$ we have

$$\begin{aligned} c_t(\{1,3\}) = \min\{&w_1(\{0,1\}) + w_1(\{1,3\}), \\ &w_1(\{0,3\}) + w_2(\{0,1\}) + \alpha_1, \\ &w_2(\{0,1\}) + \alpha_1 + w_2(\{1,3\}) + \alpha_3\} \end{aligned}$$

  Notice that $c_t(\{1,3\}) \geq w_1(\{0,3\})+w_2(\{1,3\})+\alpha_1$ since $w_1(\{0,3\}) \geq w_1(\{0,1\}) \geq w_2(\{0,1\}) + \alpha_2$. Because $w_1(\{0,1\}) \geq w_2(\{0,1\}) + \alpha_1$, $w_1(\{1,3\})$ $w_1(\{0,3\})$ and $w_1(\{1,3\}) \leq w_2(\{1,3\})+\alpha_3$ we have

$$w_1(\{1,3\}) + w_2(\{0,1\}) + \alpha_1 \leq c_t(\{1,2\})$$

  Because $\alpha_2 > r$, $w_1(\{0,1\})+w_1(\{1,2\})+w_1(\{1,3\}) \leq w_1(\{0,3\})+ w_2(\{0,1\})+w_2(\{0,1\})+\alpha_1+w_2(\{1,2\})+\alpha_2$ implies $w_1(\{0,1\})+ w_1(\{1,3\})+r-\alpha_2 \leq w_1(\{0,3\})+w_2(\{0,1\})+\alpha_1$ and $w_1(\{0,1\})+ w_1(\{1,2\}) + w_1(\{1,3\}) \leq w_2(\{0,1\}) + \alpha_1 + w_2(\{1,2\}) + \alpha_2 + w_2(\{1,3\})+\alpha_3$ implies $w_1(\{0,1\})+w_1(\{1,3\})+r-\alpha_2 \leq w_2(\{0,1\})+ \alpha_1 + w_2(\{1,3\}) + \alpha_3$ we have

$$w_1(\{0,1\}) + w_1(\{1,3\}) + r - \alpha_2 \leq c_t(\{1,2\})$$

  Because $w_1(\{1,2\}) \geq w_1(\{0,2\})$ since Prim's algorithm constructs edge $\{1,2\}$ instead of $\{0,2\}$, $w_1(\{0,1\})+w_1(\{1,2\})+w_1(\{1,3\}) \leq w_1(\{0,2\}) + w_1(\{0,3\}) + w_2(\{0,1\}) + \alpha_1$ implies $w_1(\{0,1\}) + w_1(\{1,2\})+w_1(\{1,3\})-w_1(\{0,2\}) \leq w_1(\{0,3\})+w_2(\{0,1\})+\alpha_1$ and $w_1(\{0,1\})+w_1(\{1,2\})+w_1(\{1,3\}) \leq w_1(\{0,2\})+w_2(\{0,1\})+ \alpha_1+w_2(\{1,3\})+\alpha_3$ implies $w_1(\{0,1\})+w_1(\{1,2\})+w_1(\{1,3\})- w_1(\{0,2\}) \leq w_2(\{0,1\}) + \alpha_1 + w_2(\{1,3\}) + \alpha_3$ we have

$$w_1(\{0,1\}) + w_1(\{1,2\}) + w_1(\{1,3\}) - w_1(\{0,2\}) \leq c_t(\{1,2\})$$

  Therefore $x_1 + x_3 \leq c_t(\{1,3\})$.

$\square$