Université de Montréal

Game theoretical characterization of the multi-agent network expansion game

par

Flore Caye

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en Recherche opérationnelle

April 30, 2022

Université de Montréal

Faculté des arts et des sciences

Ce mémoire intitulé

Game theoretical characterization of the multi-agent network expansion game

présenté par

Flore Caye

a été évalué par un jury composé des personnes suivantes :

Gauthier Gidel
(président-rapporteur)

Margarida Carvalho
(directrice de recherche)

Sandra Ngueveu
(codirectrice)

Michel Gendreau
(membre du jury)

Résumé

Dans les chaînes d'approvisionnement, les producteurs font souvent appel à des entreprises de transport pour livrer leurs marchandises. Cela peut entraîner une concurrence entre les transporteurs qui cherchent à maximiser leurs revenus individuels en desservant un producteur. Dans ce travail, nous considérons de telles situations où aucun transporteur ne peut garantir la livraison de la source à la destination en raison de son activité dans une région restreinte (par exemple, une province) ou de la flotte de transport disponible (par exemple, uniquement le transport aérien), pour ne citer que quelques exemples. La concurrence est donc liée à l'expansion de la capacité de transport des transporteurs.

Le problème décrit ci-dessus motive l'étude du jeu d'expansion de réseau multi-agent joué sur un réseau appartenant à de multiples transporteurs qui choisissent la capacité de leurs arcs. Simultanément, un client cherche à maximiser le flux qui passe par le réseau en décidant de la politique de partage qui récompense chacun des transporteurs. Le but est de déterminer un équilibre de Nash pour le jeu, en d'autres termes, la strategie d'extension de capacité et de partage la plus rationnelle pour les transporteurs et le client, respectivement. Nous rappelons la formulation basée sur les arcs proposée dans la littérature, dont la solution est l'équilibre de Nash avec le plus grand flux, et nous identifions ses limites. Ensuite, nous formalisons le concept de chemin profitable croissant et nous montrons son utilisation pour établir les conditions nécessaires et suffisantes pour qu'un vecteur de stratégies soit un équilibre de Nash. Ceci nous conduit à la nouvelle formulation basée sur le chemin. Enfin, nous proposons un renforcement du modèle basé sur les arcs et une formulation hybride arc-chemin. Nos résultats expérimentaux soutiennent la valeur des nouvelles inégalités valides obtenues à partir de notre caractérisation des équilibres de Nash avec des chemins croissants rentables. Nous concluons ce travail avec les futures directions de recherche pavées par les contributions de cette thèse.

Mots Clés: Expansion de réseau multi-agent, Théorie algorithmique des jeux, Recherche opérationnelle, Équilibres de Nash, Programmation en nombres entiers mixtes, Inégalités valides, Flux maximum.

Abstract

In supply chains, manufacturers often use transportation companies to deliver their goods. This can lead to competition among carriers seeking to maximize their individual revenues by serving a manufacturer. In this work, we consider such situations where no single carrier can guarantee delivery from source to destination due to its operation in a restricted region (e.g., a province) or the available transportation fleet (e.g., only air transportation), to name a few examples. Therefore, competition is linked to the expansion of transportation capacity by carriers.

The problem described above motivates the study of the multi-agent network expansion game played over a network owned by multiple transporters who choose their arcs' capacity. Simultaneously, a customer seeks to maximize the flow that goes through the network by deciding the sharing policy rewarding each of the transporters. The goal is to determine a Nash equilibrium for the game, in simple words, the most rational capacity expansion and sharing policy for the transporters and the customer, respectively. We recap the arc-based formulation proposed in literature, whose solution is the Nash equilibirum with the largest flow, and we identify its limitations. Then, we formalize the concept of profitable increasing path and we show its use to establish necessary and sufficient conditions for a vector of strategies to be a Nash equilibrium. This lead us to the first path-based formulation. Finally, we propose a strengthening for the arc-based model and a hybrid arc-path formulation. Our experimental results support the value of the new valid inequalities obtained from our characterization of Nash equilibria with profitable increasing paths. We conclude this work with the future research directions paved by the contributions of this thesis.

Keywords: Multi-agent network expansion, Algorithmic game theory, Operations research, Nash Equilibria, Mixed-integer programming, Valid inequalities, Maximum flow.

Contents

List of Tables

List of Figures

List of Acronyms & Abbreviations

MANEG	Multi-agent network expansion game
ARC_{IP}	Arc-based formulation
PATH_{IP}	Path-based formulation
HYB_{IP}	Hybrid formulation
noneg	Non-negative profit cuts
filter	Never-profitable path cuts

Notation

Game description

Parameters and sets

$\mathcal{A}$	finite set of players
$G = (V,E)$	digraph (network) with set of nodes V and set of arcs E
E_u	set of arcs in G of agent $u \in \mathcal{A}$
E_F	set of forward arcs in the residual graph associated to G
E_B	set of backward arcs in the residual graph associated to G
$\mathcal{C} := (c_{ij})_{ij \in E}$	vector of dimension $\|E\|$ where each entry ij provides the unit cost of expanding the capacity of arc $(i,j) \in E$
$\overline{\mathcal{Q}} := (\overline{q}_{ij})_{ij \in E}$	vector of dimension $\|E\|$ where each entry ij provides an upper bound $\overline{q}_{ij}$ on the capacity of the arc $ij \in E$
$\underline{\mathcal{Q}} := (\underline{q}_{ij})_{ij \in E}$	vector of dimension $\|E\|$ where each entry ij provides a lower bound $\underline{q}_{ij}$ on the capacity of the arc $ij \in E$
π	total reward awarded by the customer
o,d	nodes in V representing the origin and the destination
$o \sim d$	path from $o \in V$ to $d \in V$
$\mathcal{P}$	set of all $o \sim d$ paths
K	$\max_{ij \in E} \overline{q}_{ij}$
F_{max}	theoretical maximum flow in the network

Acknowledgment

First and foremost, I want to thank my supervisors, Pr. Carvalho and Ngueveu. They provided an excellent scientific environment, but also a lot of their time and patience. Their kind mentorship helped me to not only finish this master thesis, but more importantly to develop a steady basis on which I can confidently start a Ph.D. For their investment and trust, I am forever grateful.

I want to thank all my fellow students, for always making me feel like I belong and I could do it. Despite working remotely most of the time, we managed to create an caring and stimulating dynamic which I hope to benefit from in my future years as a student. Carl, Federico, Justine, Warley: you made this lab the best environment to succeed!

Finally, I thank my family and friends who supported me throughout this process, despite being on the other side of the ocean in the middle of a pandemic. I will fly back to you soon!

Introduction

Context

In Operations Research, the vehicle routing problem is one of the most classical and challenging optimization problems studied. Typically, this problem involves a single decision maker controlling fully the planning of the routes of its fleet in a given network. However, in some real-world applications, carriers may not have access to the full network or, in particular, they may not be able to make a delivery between an origin and a destination specified by a customer. This limitation may be caused by the fact that the networks encompass different modes of transportation or authorized operating locations. Consequently, the study of problems involving different transportation agents operating in a network is motivated. In addition, consideration of non-cooperative game-theoretic aspects is also motivated, since the decisions of the various transportation agents operating in a network are expected to influence each other.

In this work, we study the game proposed by Chaabane et al. [2017], the multi-agent network expansion game (MANEG), where different carriers compete for the flow of a customer in a network. The decisions (strategies) of the carriers (players) pertain to investments to increase the capacity of their arcs. In turn, the customer (also a player of the game) decides the sharing policy, i.e., how to split a reward among the carriers. All players are self-interested which means they aim to maximize their own profits. While each carrier profit is given by the revenue received from the sharing policy minus the investment costs on capacity expansion, the customer profit is the maximum flow going through the network from a given origin to a destination.

To analyze a game, we must anticipate its outcome, i.e., the decisions that we expect the players to select. In this way, we can evaluate various elements of the game such as whether there are players that will leave the game (e.g., due to null profits), players that dominate the game, and the social welfare associated with it, to name a few. Such analyzes support policy-makers in their task of regulating competition. Thus, the goal of our work is to determine and characterize the outcome of MANEG. This lead us to concentrate on the well-known game solution concept called *Nash equilibria*. Roughly speaking, a Nash

equilibirum specifies a vector of strategies (i.e., it fixes their decisions) for the players such that no player has incentive to unilaterally deviate from it. Therefore, it is reasonable to expect that a game outcome is a Nash equilibrium.

Contributions

Chaabane et al. [2017] showed that MANEG has always a (pure) Nash equilibrium and that the problem of determining the equilibrium achieving the largest flow is NP-hard. For the latter problem, they propose a mixed-integer linear program which we designate by arc-based formulation (ARC_{IP}).

Our contribution is the characterization of Nash equilibria through a necessary and sufficient condition based on the formalization of the concept of *profitable increasing paths*. This lead us to a novel path-based formulation (PATH_{IP}). As expected, the path-based formulation has the issue of requiring, as a preprocessing step, the computation of all paths between a pre-defined origin and destination. To overcome this problem, our third contribution is the enhancement of both ARC_{IP} and PATH_{IP} through valid inequalities based on profitable increasing paths and the fine-tuning of the big-M parameter fot ARC_{IP}. We also combine the formulations in a hybrid arc-path based formulation. Fourth, we provide extensive computational experiments allowing us to evaluate the different ingredients of our contributions and to prescribe different formulations accordingly with the instance at hand.

Our work also contributes to close the gap between the Game Theory and Operations Research communities by tackling a game through mathematical programming tools.

Organization of the thesis

In Chapter 1, we revise the literature related to the game tackled in this thesis. The goal is to position our work within the literature in game theory using mixed-integer programming techniques and on similar graph games. Chapter 2 provides background on the arc-based formulation for MANEG as well as the identification of two of its limitations. In Chapter 3, we provide a novel characterization of equilibria for MANEG based on the new concept of *profitable increasing paths*. This leads to an intuitive and simple to describe path-based formulation. In Chapter 4, *(i)* we provide a fine-tuning of the big-M parameters for the arc-based formulation, *(ii)* we devise two new families of valid inequalities for both arc and path-based formulations with the aim to overcome their respective drawbacks, and *(iii)* we propose a strengthened hybrid arc-path formulation. In Chapter 5, through extensive empirical experiments, we show that the theoretical contributions of the previous Chapter

lead to better performance than the state-of-the-art formulation. Finally, we summarize this thesis contributions and suggest future research directions in Chapter 6.

Chapter 1

Literature review

The field of game theory formalizes concepts aiming at modelizing and analizing situations where different agents interact and are self-driven. Such concepts are widely used to exploit classical problems such as electricity markets or kidney exchange (e.g., in Carvalho et al. [2017]) to name a few. In this literature review, we describe some links between mathematical programming, particularly, integer programming, game theory, and networks. In Section 1.1, we focus on integer programming techniques used to solve non-cooperative games, i.e., games in which players are self-interested and no bidding agreements can be established. Then in Section 1.2, we review relevant work concerning games played in networks to position the one tackled in this thesis.

1.1. Integer programming techniques for games

The use of integer programming tools to solve games dates back to Sandholm et al. [2005] and Sagratella [2016] and it is more recently prominent for tackling Integer Programming Games (IPGs), defined first by Köppe et al. [2011], then by Carvalho [2016]. An integer programming (IP) game with n players is a non-cooperative game where the set of strategies for each player is a set of integer points inside a polytope. Nash equilibria, defined in Nash [1950], are a common solution concept to compute for games, as they guarantee stability. In the case of IPGs, it has been proven by Carvalho et al. [2018] that the problem of deciding if an IPG has a Nash equilibrium is Σ_2^p-complete. As a consequence, recent research is interested into exploiting the integrality property of such games. Traditional mathematical (integer) programming frameworks are explored to fit the specific context of games and facilitate Nash equilibria computation. We focus here on three main approaches: Column-row generation, Branch and Cut and convex approximation.

First, in Carvalho et al. [2022] an algorithm, Cut and Play, is presented. Its core concept is to use a decomposition framework, column (and row) generation (Gilmore and Gomory [1961]), to compute only relevant strategies and associated constraints. By using only a

subset of the feasible set of strategies, it becomes easier to compute Nash equilibria. Another approach, deeply rooted in integer programming, is Branch and Cut. Carvalho et al. [2021] uses well studied families of cuts (e.g., Gomory Mixed-Integer as well as Mixed-Integer Rounding, and Knapsack Cover Cuts) and game-tailored cuts (value cuts) to compute equilibria in IPGs polyhedral representable games (e.g., IPGs and multi-leader bilevel games). A third angle is convexification of games. Harks and Schwarz [2022] prove that there exists a convexified instance of generalized IPGs such that for a fixed situation, every player has a linear cost function and a polyhedral strategy. This result allows new characterizations of the existence and computability of generalized Nash equilibria. Other forms of reformulations are proposed by Guo et al. [2021], using Burer [2009] methods and cutting planes to compute pure NE. In this work, they develop specific KKT conditions for equilibria of games. Lastly, Dragotto and Scatamacchia [2021] develop a cutting plane algorithm for the computation of pure Nash equilibria of IPGs, and show great performances on network games, which we will introduce in the following section.

1.2. Games on networks

Networks are present in numerous real life situations such as transport networks. Some of these situations involve more than one agent, which naturally calls for a game theory insight. This game theoretical perspective on network problems like congestion games is well established, and developed in Roughgarden [2007] as well as Tardos and Wexler [2007]. We present here some examples of games on networks that we consider relevant for the following work as well as we review the works closely related to ours. To this aim, we focus exclusively on non-cooperative games.

First, we consider a classic network problem, the network design, tackled with a game theoretical approach in Chen and Roughgarden [2006]. In this work, focused on the price of stability (roughly, measure of social welfare benefit), different concepts of equilibria are developed, such as approximate Nash equilibria. Then, we consider Sagratella et al. [2020], where the non-cooperative fixed charge transportation problem (NFCTP) is studied. This is the extension of the fixed charge transportation problem to multiple players. Here, all players solve a fixed charge transportation problem simultaneously. This problem is formalized as a mixed integer program, and the associated properties are exploited to prove the existence of Nash equilibria as well as numerical methods to compute them.

Finally, we mention a game on networks closely related to the multi-agent network expansion game (MANEG): the multi-agent project scheduling problem (with controllable

processing times) presented in Agnetis et al. [2015]. In such problem, the objective is to minimize the make span of a project. As in the MANEG, the activity of each agents is modeled with the arcs in the network representing the different activities to complete a project. The project scheduling problem can be view as a min-cost-flow problem while the MANEG is a max-flow on similar networks. In Agnetis et al. [2015], a characterization of Nash equilibria in terms of the existence of certain types of cuts, modeled after the network structure, is proposed. At last, the following work takes its roots into Chaabane et al. [2017], for which the formulation of MANEGs as a mixed-integer program is developped in Chapter 2. Along with Chaabane et al. [2017]'s formulation, a proof of existence for MANEGs' Nash equilibria is given.

Chapter 2

Multi-agent network expansion game

In this chapter we provide the background and the state-of-the-art mathematical programming formulation of the multi-agent network expansion game (MANEG). In Section 2.1, we formally present the game and specify the notations used in this work. After some basic definitions and properties, we go into more detail into the concept of residual graph. In Section 2.2, we first detail how this residual graph is used to model Nash equilibria in a MANEG. We then explain the mathematical formulation by Chaabane et al. [2017] of the maximum flow Nash equilibrium in a MANEG, designated by ARC_{IP}. Finally, in Section 2.3, a small example is used to show some limitations of ARC_{IP}, and thus motivate the remaining work.

2.1. Definitions and notation

We are considering a game on a network, as described in Chaabane et al. [2017]. There are two sets of players, a customer and a set of agents (carriers), taking decisions simultaneously. The goal of the customer is to obtain the highest possible flow going from a pre-defined origin to a pre-defined destination of the network. The customer controls the reward proportion paid to each agent. The goal of each agent is to maximize their individual profits. The agents manage the arcs' capacity in the network. The difficulty arises from the fact that the profit for each agent depends (in part but not only) on the maximum flow going through the network and the split of the reward decided by the customer. Moreover, the maximum flow depends on the capacities of the arcs decided by the agents. This underscores the interaction between the players and the challenge of anticipating the most *rational* strategies for the game.

Next, we describe the components defining our game.

Definition 2.1.1. *A multi-agent network is given by the following tuple:*

$$< \mathcal{A}, G, \underline{\mathcal{Q}}, \overline{\mathcal{Q}}, \mathcal{C}, \pi > ,$$

where

- $\mathcal{A} = \{A_1, A_2, \ldots, A_m, A_{m+1}\}$ *is the set of players with player* A_{m+1} *denoting the customer;*
- $G = (V,E)$ *is a digraph with a set of nodes* V*, including an origin* o *and a destination* d*, and a set of arcs* E *given by the union of the disjoint sets* E_u *for each* $u \in \mathcal{A}$ *(where* $E_{A_{m+1}} = \emptyset$*);*
- $\overline{\mathcal{Q}} := (\overline{q}_{ij})_{ij \in E}$ *is a vector representing the maximum capacity of each arc of* G*;*
- $\underline{\mathcal{Q}} := (\underline{q}_{ij})_{ij \in E}$ *is a vector representing the minimum capacity of each arc of* G*;*
- $\mathcal{C} := (c_{ij})_{ij \in E}$ *is a vector representing the unit capacity expansion cost of each arc of* G*;*
- π *is the total reward given by* A_{m+1}*, to be split between the* m *transportation agents.*

All game parameters are integer.

Figure 2.1 provides an example of a multi-agent network. In our example, there are 3 carriers: one owning the dotted arcs, one owning the dashed arcs and one owning the full arcs. In the figure, for each arc (i,j), we provide in parenthesis their minimum and maximum capacities, $\left[\underline{q}_{ij}, \overline{q}_{ij}\right]$, as well as the unit capacity expansion cost c_{ij}\$. Given that all arcs have a minimum capacity of 0, no flow can go through the network without the players investing in expanding the capacity of their arcs. To completely define the game, the total reward π and the origin and destination nodes would need to be defined.

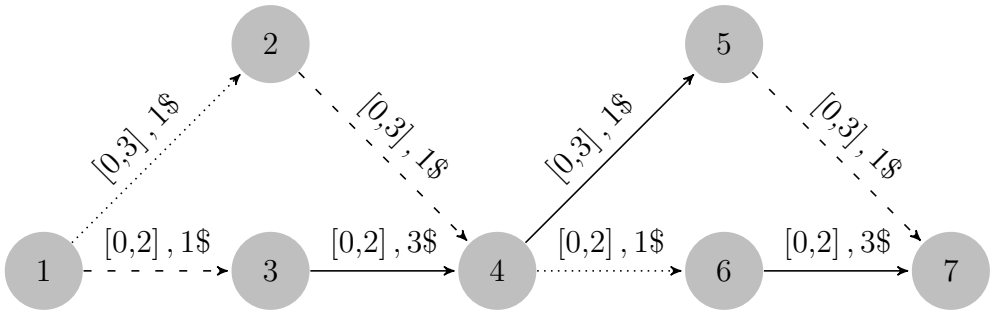


Figure 2.1. Example with 4 players: 3 carriers and 1 customer

Before preceding with the definition of strategies and utilities of the players in a MANEG, we make the simplifying assumptions:

Assumptions

- No agent owns a full path $o \sim d$ by herself.
  Since such a path would not be related to the other agents' strategies, we can safely make this assumption without altering the game.
- The vector of minimum capacities $\underline{\mathcal{Q}}$ is $\mathbf{0}$.

The last assumption above allows us to not take into consideration the *free* flow that could go through the network, i.e., the flow that could cross the network with no investment on capacity expansion. In addition, it does not compromise the use of the instances from Chaabane et al. [2017] since $\underline{\mathcal{Q}}$ is set to $\mathbf{0}$. Moreover, as discussed later in Chapter 6, the concepts and valid inequalities that we are presenting here can be adapted to fit non-trivial minimum capacity vectors by representing the cost of each arc as a stair function instead of a fixed value.

As mentioned initially, the goal of the customer is to maximize the flow going from an origin to a destination of the network. Thus, the profit of the customer in the game is the associated maximum flow. This leads to an immediate simplification observed by Chaabane et al. [2017]: the customer may not be considered a player like the others in the sense that any change in the reward sharing policy of the customer has no direct effect on her profit; in other words, the customer cannot exert a direct influence on the maximum flow since she does not control the capacity of the arcs. On the other hand, a carrier can directly influence the flow going through the network. Due to this status difference, even though all players are playing simultaneously, we allow ourselves to shorten and define a strategy vector as follows:

Definition 2.1.2. *A strategy vector for a multi-agent network* $\left\langle \mathcal{A}, G, \underline{\mathcal{Q}}, \overline{\mathcal{Q}}, \mathcal{C}, \pi \right\rangle$ *is given by*

$$\mathcal{S} = (S_1, \ldots, S_m),$$

where $S_u = (q_{ij})_{ij \in E_u}$ *is the vector of capacities over the arcs of player* $u \leq m$*. The strategy vector can be completed with a corresponding sharing policy (namely, the customer's strategy)* $S_{m+1} = (w_u)_{u=1,\ldots,m}$ *providing the proportion of* π *allocated to each carrier.*

Now, we have all the elements to define the set of feasible strategies for each player and trivial strategy.

Definition 2.1.3. *Given a multi-agent network* $\left\langle \mathcal{A}, G, \underline{\mathcal{Q}}, \overline{\mathcal{Q}}, \mathcal{C}, \pi \right\rangle$*, a strategy vector* $\mathcal{S} = (S_1, \ldots, S_m, S_{m+1})$ *is feasible if the following conditions hold*

(1) For each arc $(i,j) \in E$

$$q_{ij} \in \mathbb{N}.$$

(2) For each arc $(i,j) \in E$

$$\underline{q}_{ij} \leq q_{ij} \leq \overline{q}_{ij}.$$

(3) The customer's strategy is a sharing policy, i.e., $\sum_{u=1}^{m} w_u = 1$ *and* $w_u \geq 0$ *for* $u = 1, \ldots, m$.

We remark that in the original definition of feasible strategies by Chaabane et al. [2017], the decisions (strategies) of the carriers are not (explicitly) restricted to integer values. However, in their mathematical programming formulation to solve the game, they impose arcs' capacity expansions to be integer. Thus, we make this restriction explicit in Definition 2.1.3.

Definition 2.1.4. *Given a multi-agent network* $\left\langle \mathcal{A}, G, \underline{\mathcal{Q}}, \overline{\mathcal{Q}}, \mathcal{C}, \pi \right\rangle$*, a strategy vector*

$$\underline{\mathcal{S}} = \left((\underline{q}_{ij})_{ij \in E_u} \right)_{u \in \{1,\dots,m\}}$$

is called a trivial strategy.

Note that in a trivial strategy, no agent increases any of her arc's capacity. We also remark that a trivial strategy vector remains feasible for any sharing policy. It would be more correct (yet heavier) to write that there exists an infinity of trivial feasible strategies. Due to the assumption made in the beginning of this section, the trivial strategy is equal to zero, corresponding to the case for which the maximum flow is zero. For any strategy $\mathcal{S}$, we denote by $F(\mathcal{S})$ the maximum flow going through the network G from $o \in V$ to $d \in V$ under the capacities given by the strategy $\mathcal{S}$.We denote $\underline{v} = F(\underline{\mathcal{S}})$.

Definition 2.1.5. *Given a multi-agent network* $\left\langle \mathcal{A}, G, \underline{\mathcal{Q}}, \overline{\mathcal{Q}}, \mathcal{C}, \pi \right\rangle$*, a profit vector is*

$$Z(\mathcal{S}) = \left(Z_1(\mathcal{S}), \dots, Z_{m+1}(\mathcal{S}) \right),$$

where

- *For agents* $u \leq m$*:* $Z_u(\mathcal{S}) = w_u \pi \left(F(\mathcal{S}) - \underline{v} \right) - \sum_{ij \in E_u} c_{ij} \left(q_{ij} - \underline{q_{ij}} \right)$.
- *For customer:* $Z_{m+1} = F(\mathcal{S})$.

Finally, we can mathematically define MANEG:

Definition 2.1.6. *A Multi-Agent Network Expansion Game (MANEG) is composed of:*

- *A multi-agent network* $\left\langle \mathcal{A}, G, \underline{\mathcal{Q}}, \overline{\mathcal{Q}}, \mathcal{C}, \pi \right\rangle$*;*
- *A strategy vector* $\mathcal{S}$ *and a sharing policy* $(w_u)_{u=1,\dots,m}$*;*
- *A profit vector* $Z(\mathcal{S})$.

The broadly accepted concept of solution for a game is *Nash equilibria*. A Nash equilibrium specifies a strategy vector for each player such that none has incentive to unilaterally deviate. Under a Nash equilibrium, all players select their optimal strategies given the fixed strategies of the opponents. Chaabane et al. [2017] showed that MANEG has always a pure Nash equilibrium and thus, we focus on this solution concept:

Definition 2.1.7. *A strategy vector* $\mathcal{S}$ *is a (pure) Nash equilibrium of* $\left\langle \mathcal{A}, G, \underline{\mathcal{Q}}, \overline{\mathcal{Q}}, \mathcal{C}, \pi \right\rangle$ *if for each player* $u = 1, \dots, m$*:*

$$Z_u(\mathcal{S}) \geq Z_u(S_1, \dots, S_{u-1}, S'_u, S_{u+1}, \dots, S_{m+1}) \quad \forall S'_u \text{ feasible}, \tag{2.1.1}$$

and for the customer $m+1$*:*

$$Z_{m+1}(\mathcal{S}) \geq Z_{m+1}(S_1,\ldots,S_m,S'_{m+1}) \quad \forall S'_{m+1} \text{ feasible.} \tag{2.1.2}$$

Definition 2.1.7 makes it clear that the customer can never improve her profit by herself (i.e., by unilaterally deviating from the strategy vector). Hence, to characterize a Nash equilibrium in a MANEG, it is enough to consider that no transportation agent has incentive to deviate. Thus, we only need to consider Inequalities (2.1.1). The key point to represent an equilibrium strategy in Chaabane et al. [2017] is to define for each transportation agent u a residual graph G_r^u, and check in that graph if player u can make savings (i.e., decrease costs).

Definition 2.1.8. *A strategy vector* $\mathcal{S}$ *in* $\left\langle \mathcal{A},G,\underline{\mathcal{Q}},\overline{\mathcal{Q}},\mathcal{C},\pi \right\rangle$ *is a non-poor strategy if there is no agent who can increase her profit unilaterally without changing the flow going through the network. In a non-poor strategy, all arcs are saturated.*

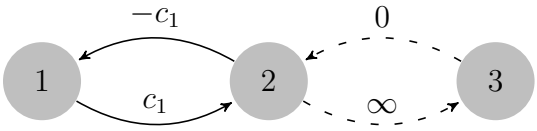
Definition 2.1.9. *Consider a multi-agent network* $\left\langle \mathcal{A},G,\underline{\mathcal{Q}},\overline{\mathcal{Q}},\mathcal{C},\pi \right\rangle$*, a feasible strategy* $\mathcal{S}$ *and a transportation agent* $u \in \mathcal{A}$*. The corresponding residual graph* $G_r^u(\mathcal{S}) = (V_r^u, E_F \cup E_B)$ *is a network such that:*

- *The nodes are the same as* $G(V = V_r^u)$*;*
- *If* $(i,j) \in E$*, then* $(i,j) \in E_F$ *(forward arc) and* $(j,i) \in E_B$ *(backward arc); moreover,* δ_F^{iju} *(resp.* δ_B^{jiu}*) denote the costs for forward (resp. backward) arcs in the residual graph.*

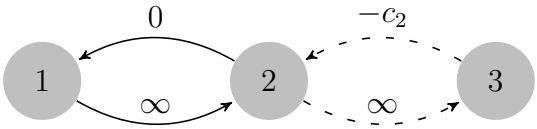
As we will see next, Chaabane et al. [2017] set the costs of each arc in a residual graph so that it reflects whether the capacity can be increased or decreased, and whether or not such deviation can affect agent u. In Figure 2.2, we provide an example of a residual graph associated with a strategy vector $\mathcal{S}$; we will get back to it later in this section.

Remark 2.1.10. *A direct consequence of the assumption on the lower bound capacities is that a non-poor strategy is now equivalent to a strategy where all arcs are saturated in the case of a maximum flow.*

(a) MANEG with 2 transports and strategy $\mathcal{S}$ given by the left-most number in each arc label



(b) Residual graph associated with the current strategy $\mathcal{S}$ for the full arc agent



(c) Residual graph associated with the current strategy $\mathcal{S}$ for the dash arc agent

Figure 2.2. Example of residual graphs

Next, we present the residual costs by Chaabane et al. [2017]. We note that our exposition is a simplified version of the one presented in the original paper given our initial assumption on $\underline{\mathcal{Q}} = \mathbf{0}$.

More precisely, here is how the residual costs for a strategy $\mathcal{S}$ are defined in the case of a MANEG under the trivial lower bounds assumption. For each $(i,j) \in E$:

- if $(i,j) \in E_u, q_{ij} < \overline{q}_{ij}$: Agent u has rights on this arc and can increase its capacity with respect to the bounds. Here, $\delta_F^{iju} = c_{ij}$;
- if $(i,j) \in E_u, q_{ij} = \overline{q}_{ij}$: Agent u has rights on this arc but cannot increase its capacity with respect to the bounds. Here, $\delta_F^{iju} = M$ where M is a large constant;
- if $(i,j) \notin E_u$: Agent u has no rights on this arc Here $\delta_F^{iju} = M$;
- if $(j,i) \in E_u, q_{ij} > 0$: Agent u has rights on this arc and can decrease its capacity with respect to the bounds. Here, $\delta_B^{jiu} = -c_{ij}$;
- if $(j,i) \in E_u, q_{ij} = 0$: Agent u has rights on this arc but cannot decrease its capacity with respect to the bounds. Here, $\delta_B^{jiu} = M$;
- if $(i,j) \notin E_u$: Agent u has no rights on this arc. Here, $\delta_B^{jiu} = 0$.

In a MANEG, a Nash equilibrium can be described as a strategy where no agent has incentive to unilaterally invest or desinvest in any arc. Since the strategy we are dealing with are non-poor, no agent has incentive to increase her capacities. Thus, for a strategy to be a Nash equilibrium, it only has to have *no saving paths*. To do so, we are looking for a least expensive $d \sim o$ path in the residual graph. The cost of such a path in the residual

graph represents the savings an agent would make by decreasing her capacities along this path. For a strategy to be a Nash equilibrium, the cost of this path must be strictly lower than the benefits from earning one unit of flow.

In our example in Figure 2.2, the cost for the full arc agent of such path would be $-c_1$ (see Figure 2.2b). For the dash arc agent it would be $-c_2$ (see Figure 2.2b).

Remark 2.1.11. *Because of how the residual costs are defined for the backward arcs with the trivial lower bound assumption, the least expensive path for u in the residual graph is the same as the most expensive path for agent u in the network G taken backward.*

2.2. Arc-based formulation

We will refer to the mathematical program formulated in Chaabane et al. [2017] as the *arc-based formulation*. The main idea of their model is

- to define a feasible strategy as in Definition 2.1.3;
- to describe the residual graph for each player u and the associated strategy vector;
- to make sure that in each residual graph, even the most expensive invested path is profitable (hence, no player has incentive to decrease her expansion).

The last condition, central to obtain a Nash equilibrium, does not seem linear; recall, Definition 2.1.5 for the profit of the transportation agents which depends on computing the maximum flow. Chaabane et al. [2017] propose a set of primal-dual constraints to express it.

We can define for each G_r^u, a min-cost flow problem (LP_u) associated with a demand of 1 unit of flow between nodes d and o. Let φ be the decision vector of the flow in the residual graph, for each $u \in \mathcal{A}$, we have:

$$
\begin{aligned}
(LP_u) \quad & \min_{\varphi^u} \sum_{ij \in E_F(G_u)} \delta_F^{iju} \varphi_{ij}^u + \sum_{ji \in E_B(G_u)} \delta_B^{jiu} \varphi_{ji}^u \\
& \text{s.t.} \sum_{ij \in E_F \cup E_B} \varphi_{ij}^u - \sum_{ji \in E_F \cup E_B} \varphi_{ji}^u = \begin{cases} 0, \forall i \neq o,d \\ -1, i = o \\ 1, i = d \end{cases}, \qquad \forall i \in V(G_u) \\
& \varphi_{ij}^u \geq 0, \qquad \forall ij \in E(G_u).
\end{aligned}
$$

We can write the associated dual (DP_u):

$$\begin{aligned}
(DP_u) \quad \max_t \ & t_o^u - t_d^u & \\
\text{s.t. } & t_j^u - t_i^u \leq \delta_F^{iju}, & \forall ij \in E_F(G_u) \\
& t_i^u - t_j^u \leq \delta_B^{jiu}, & \forall ji \in E_B(G_u) \\
& t_i^u \in \mathbb{R}, & \forall i \in V(G_u).
\end{aligned}$$

By strong duality, we know that if (LP_u) has an optimum $\varphi^\star$, then (DP_u) is feasible and

$$Z(DP_u) = Z(LP_u),$$

where the operator $Z(LP_u)$ and $Z(DP_u)$ represent the optimal value the problems in parenthesis.

Bringing this back to our problem, it means that:

- If we can design an optimal solution for each (LP_u) (i.e., Constraint (2.2.16) holds and φ is optimal), then the dual is feasible (i.e., Constraints (2.2.14) and (2.2.15) hold for all $u \in \mathcal{A}$);
- Moreover, if we can determine a feasible solution for each (LP_u), then we have by weak duality:

$$t_o^u - t_d^u \leq \sum_{ij \in E_F(G_u)} \delta_F^{iju} \varphi_{ij}^u + \sum_{ji \in E_B(G_u)} \delta_B^{jiu} \varphi_{ji}^u, \forall u \in \mathcal{A}.$$

  By strong duality, this inequality only holds with equality for optimal solutions. This is the configuration we are aiming at in the arc-based formulation.

To summarize, we aim at building a path $d \sim o$ with a minimal residual cost, (i.e., the most expensive combination of arcs for an agent u) in the residual graph of each transportation agent. This cost is carried to dual variables t and compared to the benefits obtained by agent u from one unit of flow. If by disinvesting one unit in such path, the agent is saving more than the benefits reached from one unit of flow, then she has incentive to unilaterally deviate, and hence the current strategy is not a Nash equilibrium.

There are two auxiliary parameters in the arc-based formulation: ϵ and M. As advised in Chaabane et al. [2017], we set ϵ to 1. M is supposed to be a rebarbative cost; we discuss in Section 2.3 the challenges associated with this parameter. Finally, we can provide the arc-based formulation (ARC_{IP}) whose solution is a Nash equilibrium achieving the maximum flow:

$$\max\ F - \frac{\sum_{ij\in E} c_{ij}q_{ij}}{1+\sum_{ij\in E} c_{ij}\overline{q}_{ij}}$$

The next three constraints assure that a strategy is feasible for the game

$$\sum_{ij\in E} q_{ij} - \sum_{ji\in E} q_{ji} = \begin{cases} 0, i \neq s,t \\ F, i = s \\ -F, i = t \end{cases}, \quad \forall i \in V \tag{2.2.1}$$

$$q_{ij} \leq \overline{q}_{ij}, \quad \forall (i,j) \in E \tag{2.2.2}$$

$$\sum_{u\in\mathcal{A}} w_u = 1 \tag{2.2.3}$$

The next constraints are setting binary values in order to define the residual costs

$$x_{ij} \leq \overline{q}_{ij} - q_{ij} \leq \overline{q}_{ij}x_{ij}, \quad \forall (i,j) \in E \tag{2.2.4}$$

$$y_{ij} \leq q_{ij} \leq \overline{q}_{ij}y_{ij}, \quad \forall (i,j) \in E \tag{2.2.5}$$

$$(1-x_{ij})\overline{q}_{ij} \leq q_{ij} \leq (1-\alpha_{ij})\overline{q}_{ij} - \epsilon x_{ij}, \quad \forall (i,j) \in E \tag{2.2.6}$$

$$\epsilon(\beta_{ij}+y_{ij}) \leq q_{ij} \leq y_{ij}\overline{q}_{ij}, \quad \forall (i,j) \in E \tag{2.2.7}$$

$$\alpha_{ij} \leq x_{ij}, \quad \forall (i,j) \in E \tag{2.2.8}$$

$$y_{ij} + \beta_{ij} \leq 1, \quad \forall (i,j) \in E \tag{2.2.9}$$

The next constraints are defining the residual costs

$$\delta_F^{ij,u} = c_{ij} - (c_{ij}-M)(1-x_{ij}) - c_{ij}\alpha_{ij}, \quad \forall (i,j) \in E_u, \forall u \in \mathcal{A} \tag{2.2.10}$$

$$\delta_F^{ij,u} = M(2-x_{ij}-\alpha_{ij}), \quad \forall (i,j) \in E \setminus E_u, \forall u \in \mathcal{A} \tag{2.2.11}$$

$$\delta_B^{ji,u} = (1-\beta_{ij}-y_{ij})M - c_{ij}y_{ij}, \quad \forall (i,j) \in E_u, \forall u \in \mathcal{A} \tag{2.2.12}$$

$$\delta_B^{ji,u} = (1-\beta_{ij}-y_{ij})M, \quad \forall (i,j) \in E \setminus E_u, \forall u \in \mathcal{A} \tag{2.2.13}$$

The next constraints describe the feasibility set of (DP_u)

$$t_j^u - t_i^u \leq \delta_F^{ij,u}, \quad \forall (i,j) \in E, \forall u \in \mathcal{A} \tag{2.2.14}$$

$$t_i^u - t_j^u \leq \delta_B^{ji,u}, \quad \forall (i,j) \in E, \forall u \in \mathcal{A} \tag{2.2.15}$$

The next constraint describes the feasibility set of (LP_u)

$$\sum_{ij \in E_F \cup E_B} \varphi_{ij}^u - \sum_{ji \in E_F \cup E_B} \varphi_{ji}^u = \begin{cases} 0, \forall i \neq s,t \\ -1, i = s \\ 1, i = t \end{cases}, \quad \forall i \in V, \forall u \in \mathcal{A} \tag{2.2.16}$$

The next constraint imposes the necessary strong duality condition

$$t_0^u - t_{n+1}^u \geq \sum_{ij \in E_F(G_u)} \delta_F^{iju} \varphi_{ij}^u + \sum_{ji \in E_B(G_u)} \delta_B^{jiu} \varphi_{ji}^u, \quad \forall u \in \mathcal{A} \tag{2.2.17}$$

This last constraint makes sure that the optimal solution to (LP_u) is not a saving path for each agent u

$$\begin{gathered} t_{n+1}^u - t_0^u < w_u \pi, \quad \forall u \in \mathcal{A} \\ x_{ij}, \alpha_{ij}, y_{ij}, \beta_{ij}, \varphi_{ij}^u, \varphi_{ji}^u \in \mathbb{B}, \quad \forall (i,j) \in E, \forall u \in \mathcal{A} \\ F \geq 0 \\ w_u \geq 0, \forall u \in \mathcal{A} \\ q_{ij} \in \mathbb{N}, \forall (i,j) \in E \\ \delta_B^{jiu}, \delta_F^{iju}, t_i^u, t_j^u \in \mathbb{R}, \forall (i,j) \in E, \forall u \in \mathcal{A} \end{gathered} \tag{2.2.18}$$

The objective function maximizes the flow and the second penalization term aims to avoid non-poor strategies; we refer the reader to Chaabane et al. [2017] for details.

We remark that the right-hand side of Constraints (2.2.17), is not linear. Its linearization can be done through standard techniques and thus, we provide the equivalent linear reformulation in Appendix A.

2.3. Identification of limitations

Now that the arc-based formulation has been detailed, we use the following simple instance to analyze the performance of solving ARC_{IP}:
As in the previous examples, each arc (i,j) is labeled $\left[\underline{q}_{ij}, \overline{q}_{ij}\right], c_{ij}\$$. We call *player 1* the agent managing the full arcs, *player 2* the agent owning the dashed arcs, and *players 3* the agent controlling the dotted ones. The origin is node 1 and the destination node 7. This instance is used to motivate the work in the upcoming chapters.

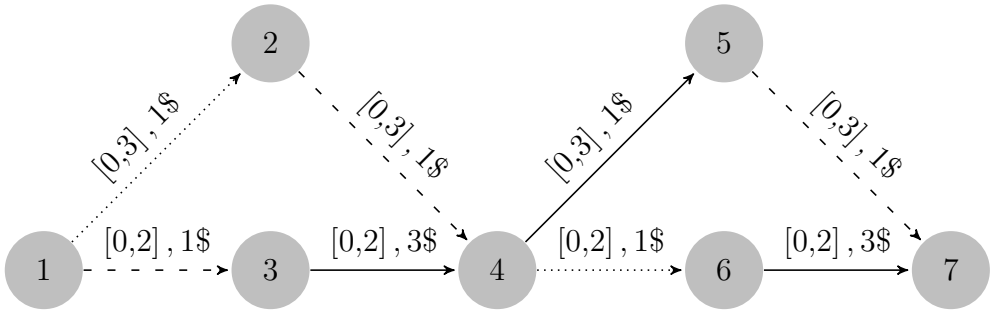


Figure 2.3. Example with 3 carriers

For this brief computational results, we use the same setup as for all our experiments, detailed in Chapter 5 so here we refrain from repeating those details. In Table 2.1, we present our results for two instances based on the illustrated network where only the reward π differs. For each of these instances, we solved ARC_{IP} for two values of the big-M auxiliary parameter. The table provides in the third column the maximum flow of the computed equilibrium, in the fourth column the time to solve ARC_{IP} in seconds, and in the fifth column the number of nodes explored by the solver.

π	M	Flow F	runtime (s)	nodes
7	14	2	0.015	1
	10^6	4	0.024	17
1	5	0	0.005	0
	10^6	3	0.024	33

Table 2.1. Influence of M in the performance

π	VI	Flow F	runtime (s)	nodes
7	none	4	0.024	17
	noneg	2	0.025	1
	filter	4	0.026	17
1	none	3	0.024	33
	noneg	0	0.004	0
	filter	0	0.004	0

Table 2.2. Influence of the valid inequalities (VI) with (a large) $M = 10^6$

In both instances, large values of M result in more nodes explored by the MIP solver. This is not surprising as the use of big-M parameters is known to contribute to non-tight

linear relaxations and thus to result in large branch-and-bound trees. More importantly, for the large values of M (i.e., $M = 10^6$), the model produces incorrect solutions, i.e., solutions that do not represent a Nash equilibrium. This is especially visible when $\pi = 1$. In this case, due to a reward lower than any unitary capacity expansion cost, only the trivial strategy is a Nash equilibrium. Hence any solution with a flow higher than zero, will lead to non-Nash strategies. Remark that ARC_{IP} with $M = 10^6$ provides a solution with flow equal to 3, demonstrating that the solution is not correct. Please note that this does not invalidate the theoretical correctness of ARC_{IP}; the issue described here is associated with numerical issues introduced by the large values of M.

These observations leads us to try to improve the model, by making it more robust to big-M variations, to assure that the trivial solutions are found fast when they are the unique Nash equilibrium, and to improve the associated linear relaxation.

To that aim, we devise a combinatorial formulation based on *o-d* paths in Chapter 3. This model inspires the development of valid inequalities for the arc-based formulation. We anticipate through Table 2.2, the impact of our valid inequalities, namely, *noneg* for *non-negative profit cuts*, and *filter* for *never-profitable path cuts*. Indeed, when the reward is (high) $\pi = 7$, the non-negative profit inequalities correct the error induced by the big-M. When the reward is (low) $\pi = 1$, the never-profitable path inequalities fix the solution correctness and improve efficiency.

In Chapter 4, we propose an hybrid reformulation, joining both arc and path-based formulations. Finally, we present in Chapter 5, our instances and discuss the performances of all three formulations.

Chapter 3

Set partitioning reformulation

In the domain of integer programming, mathematical programming formulations based on decision variables associated with the arcs of a graph can frequently be reformulated through paths in that graph. In this chapter, we follow this research direction and we present the first path-based formulation for MANEG.

In Section 3.1, we provide a necessary and sufficient condition for the characterization of Nash equilibria for MANEG using the definition of profitable increasing paths. Our path-based formulation is described in Section 3.2. We end this chapter by discussing in Section 3.3 the advantages and disadvantages of our new formulation in comparison with the arc-based formulation.

3.1. Profitable increasing paths

In what follows, we define the notation of profitable increasing paths and combination of $o \sim d$ paths. This allow us to describe, using paths, a Nash equilibrium for MANEG. In a sightly abuse of notation, for path a p in a graph $G = (V,E)$ and $E_u \subset E$, we define $p \cap E_u$ as the set of arcs belonging to E_u used by the path p.

Definition 3.1.1 (*Profitable increasing path*)**.** *Consider a multi-agent network* $\left\langle \mathcal{A}, G, \underline{\mathcal{Q}}, \overline{\mathcal{Q}}, \mathcal{C}, \pi \right\rangle$ *and a sharing policy* w*. A path* p *from* o *to* d *is called a profitable increasing path if for each player* $u \in \mathcal{A}$*:*

$$\sum_{ij \in p \cap E_u} c_{ij} \leq w_u \pi.$$

Definition 3.1.2 (*Combination of paths*)**.** *Consider a multi-agent network* $\left\langle \mathcal{A}, G, \underline{\mathcal{Q}}, \overline{\mathcal{Q}}, \mathcal{C}, \pi \right\rangle$*. A combination of paths is a set of paths* $\{p_1, \ldots, p_n\}$ *, all non-null, in the graph from* o *to* d *associating a strictly positive number* λ_i*, to each path* p_i *in the set. Such* λ *is called the path multiplicity.*

The idea behind the definition of combination of paths is the following: a strategy vector $\mathcal{S}$ for a MANEG can be described by a combination of paths such that for each arc $(i,j) \in E$

the sum of multiplicities associated with the paths passing in that arc is equal to the q_{ij} given by the strategy vector $\mathcal{S}$.

Theorem 3.1.3. *In a multi-agent network expansion game* $\left\langle \mathcal{A}, G, \underline{\mathcal{Q}}, \overline{\mathcal{Q}}, \mathcal{C}, \pi \right\rangle$*, a non-poor strategy* $\mathcal{S}$ *is a pure Nash equilibrium if, and only if, it can only be expressed with a combination of profitable increasing paths.*

PROOF. **($\Longrightarrow$, by contradiction)** We prove the following statement: *a non-poor strategy* $\mathcal{S}$ *that can be expressed as a combination of paths, not all profitable increasing, is not a pure Nash equilibrium.*

Let $\mathcal{S}$ be a non-poor strategy vector, with a corresponding combination of n paths $\{p_1, \ldots, p_n\}$ with multiplicities λ_i for $i = 1, \ldots, n$, all non-null (guaranteed by definition). Without loss of generality, let p_1 be a non-profitable increasing path for an agent $u \in \mathcal{A}$. Then such player has incentive to unilaterally deviate, e.g., by decreasing the capacity expansion of an arc in $p \cap E_u$. To see this, note that the profit of player u is

$$Z_u(\mathcal{S}) = w_u \pi F(\mathcal{S}) - \sum_{ij \in E_u} c_{ij} q_{ij} = w_u \pi \sum_{k=1}^{n} \lambda_k - \sum_{k=1}^{n} \sum_{ij \in p_k \cap E_u} c_{ij} \lambda_k = \sum_{k=1}^{n} \lambda_k \left(w_u \pi - \sum_{ij \in p_k \cap E_u} c_{ij} \right).$$

Remark that for $k = 1$, the term in parenthesis for the last equality is negative since p_1 is not a profitable increasing path. If player u decreases by one unit the capacity of an arc in $\in p_1 \cap E_u$ (note that this is possible since all arcs of player u along p_1 have capacity at least λ_1), then the multiplicity of path p_1 would be $\lambda_1 - 1$ decreasing the value of the negative term in the profit of player u.

Therefore $\mathcal{S}$ is not a Nash equilibrium.

($\Longleftarrow$, by contradiction) Next, we prove the following statement: *if a non-poor strategy vector* $\mathcal{S}$ *is not a pure Nash equilibrium, then it can be expressed as a combination of paths including at least one non-profitable path.*

Let $\mathcal{S}$ be a non-poor strategy vector, such that agent 1 (w.l.o.g.) has incentive to deviate.

Let us start by characterizing the deviations from a non-poor strategy by a player. Recalling Chapter 2, we assumed that no player controls all the arcs of an $o \sim d$ path (essentially, such path can be removed from the game since the related flow is controlled by a single player). Therefore, if a player has incentive to deviate, then there is an arc whose increase or decrease by one unit of capacity results in a strict increase of the player profit. Moreover, since $\mathcal{S}$ is non-poor, the incentive is always to decrease capacity as all arcs are saturated; otherwise, if the player increases the capacity, there is no flow increase and the player increases costs.

Using the considerations above, we suppose that player 1 has incentive to decrease by one unit the capacity of $e \in E_1$, resulting in the vector of strategies $\hat{\mathcal{S}} = (\hat{\mathcal{S}}_1, \mathcal{S}_{-1})$ where $\mathcal{S}_{-1}$ is vector $\mathcal{S}$ without $\mathcal{S}_1$, i.e., it represents the unilateral deviation of player 1 from $\mathcal{S}$. Therefore, it holds $Z_1(\mathcal{S}) < Z_1(\hat{\mathcal{S}})$.

The strategy vector $\hat{\mathcal{S}}$ is not a non-poor strategy anymore. However, we can derive a non-poor strategy $\tilde{\mathcal{S}}$ from it such that $F(\hat{\mathcal{S}}) = F(\tilde{\mathcal{S}})$. Such strategy $\tilde{\mathcal{S}} = (\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_{-1})$ is built the following way:

- $\tilde{\mathcal{S}}_1$:
  Because we removed capacity on e to build $\hat{\mathcal{S}}$, we can choose the most expensive path for player 1 from o to d containing e, i.e., $\tilde{p} \in \arg\max_{p \in \mathcal{P} | e \in p} \sum_{ij \in p \cap E_1} c_{ij}$. Then $\tilde{\mathcal{S}}_1$ is the strategy of player 1 when all arcs in E_1 along $\tilde{p}$ see their capacities decrease by one unit from $\mathcal{S}_1$ (including e). Then all arcs managed by player 1 are saturated.

- $\tilde{\mathcal{S}}_{-1}$ is the strategy on arcs $E \setminus E_1$ where each arc is saturated.

Since all arcs in E_1 and all arcs in $E \setminus E_1$ are saturated, the strategy $\tilde{\mathcal{S}}$ is non-poor.

Then, player 1 profit is:

$$\begin{aligned} Z_1(\tilde{\mathcal{S}}) &= w_1\pi\left(F(\mathcal{S}) - 1\right) - \sum_{ij \in E_1} c_{ij} q_{ij} + \sum_{ij \in \tilde{p} \cap E_1} c_{ij} \\ &= Z_1(\mathcal{S}) - \left(w_1\pi - \sum_{ij \in \tilde{p} \cap E_1} c_{ij} \right). \end{aligned}$$

By construction,

$$Z_1(\mathcal{S}) < Z_1(\hat{\mathcal{S}}) \leq Z_1(\tilde{\mathcal{S}}).$$

Thus:

$$\begin{aligned} & Z_1(\mathcal{S}) < Z_1(\tilde{\mathcal{S}}) \\ \iff & Z_1(\mathcal{S}) < Z_1(\mathcal{S}) - \left(w_1\pi - \sum_{ij \in \tilde{p} \cap E_1} c_{ij} \right) \\ \iff & w_1\pi - \sum_{ij \in \tilde{p} \cap E_1} c_{ij} < 0 \\ \iff & w_1\pi < \sum_{ij \in \tilde{p} \cap E_1} c_{ij} \end{aligned}$$

Hence $\tilde{p}$ is a non-profitable path along which there is a positive flow under $\mathcal{S}$. Thus, $\tilde{p}$ can be used together with other paths to form a combination of paths describing $\mathcal{S}$.

□

This theorem provide us a complete characterization of a pure Nash equilibrium through the use of combinations of paths under certain conditions (i.e., being all profitable). The following formulation aims at exploiting this result by producing such combinations.

3.2. Path-based reformulation

As previously explained, it is straightforward to see that any vector of strategies can be represented through a combination of paths (together with their multiplicities). With Theorem 3.1.3, we obtain a strong link between $o \sim d$ paths and a Nash equilibrium. This motivated the path-based formulation that we present next.

Notation. Given a MANEG and the set $\mathcal{P}$ of all $o \sim d$ paths in the network, we define K as

$$K = \max_{ij \in E} \left(\overline{q}_{ij} \right).$$

This parameter K is the theoretical maximum multiplicity possible for any $o \sim d$ path in the network. We introduce also a new parameter, d_p^u for each path $p \in \mathcal{P}$ and agent $u \in \mathcal{A}$ such that:

$$d_p^u = |p \cap E_u|.$$

Essentially, d_p^u is the number of arcs owned by player u in the path p.

In the following reformulation of the Nash equilibrium attaining the maximum flow for the multi-agent network expansion game, we use a binary variable x_p^k to express the exploitation of a path $p \in \mathcal{P}$ with a multiplicity of $k = 1, \ldots, K$; in other words, x_p^k equals 1 if there is at least a capacity expansion (or equivalently, flow) of k along path p, and 0 otherwise. This definition imposes a ordering of the x variables associated with a path p: if $x_p^k = 1$, then $x_p^i = 1$ for all $i < k$. Variables w_u for each $u \in \mathcal{A}$ have the same meaning as in the arc-based formulation (sharing policy).

We also introduce two new sets of (auxiliary) binary variables, z_p^u and y_{ij}, linked by a third set of variables, $t_p^{iju} = y_{ij} \times z_p^u$, for each $(i,j) \in E, u \in \mathcal{A}, p \in \mathcal{P}$. The variable z_p^u is 1 when a path p satisfies the condition of profitable increasing path for agent u, and 0 otherwise. The decision variable y_{ij} is actually the same as in ARC_{IP}: y_{ij} is 1 if arc $(i,j) \in E$ is used (i.e., there is $p \in \mathcal{P}$ such that $(i,j) \in p$ and $x_p^1 > 0$), and 0 when $x_p^1 = 0, \forall p \in P$ with

$(i,j) \in p$. Finally, t_p^{iju}, defined as a product of two binary variables, represents the fact that an arc $(i,j) \in E$ is exploited by player u through a path p.

The goal of our formulation is to find a combination of profitable increasing paths, while making sure that no non-profitable path is formed inadvertently.

Path-based formulation. In order to describe the mathematical program, we have to respect the maximum allowed capacity expansion on each arc. We also need to make sure that all the paths forming the combination of paths in the solution (i.e., the variables x_p^k) are profitable increasing paths, and that no non-profitable path can be formed by recombining the arcs exploited. Finally, we need to make sure that a path p is exploited at a level k only if it is exploited at a level $k-1$, for $k = 1, \ldots, K$. This leads us to the following reformulation (PATH_{IP}) of the maximum flow Nash equilibrium problem in a MANEG:

$$\max \sum_{p \in \mathcal{P}} \sum_{k=1}^{K} x_p^k$$

$$\text{st.} \sum_{p \in \mathcal{P} | ij \in p} \sum_{k=1}^{K} x_p^k \leq \overline{q}_{ij}, \qquad \forall ij \in E \tag{3.2.1}$$

$$\sum_{u \in \mathcal{A}} w_u = 1 \tag{3.2.2}$$

$$x_p^{k+1} - x_p^k \leq 0, \qquad \forall p \in \mathcal{P}, \forall k = 1, \ldots, K-1 \tag{3.2.3}$$

$$z_p^u \sum_{ij \in p \cap E_u} c_{ij} - w_u \pi \leq 0, \qquad \forall p \in \mathcal{P}, \forall u \in \mathcal{A} \tag{3.2.4}$$

$$x_p^1 - z_p^u \leq 0, \qquad \forall p \in \mathcal{P}, \forall u \in \mathcal{A} \tag{3.2.5}$$

$$d_p^u x_p^1 - \sum_{ij \in p \cap E_u} y_{ij} \leq 0, \qquad \forall p \in \mathcal{P}, \forall u \in \mathcal{A} \tag{3.2.6}$$

$$\sum_{ij \in p \cap E_u} \left(y_{ij} - t_p^{iju} \right) \leq d_p^u - \frac{d_p^u}{|p|}, \qquad \forall p \in \mathcal{P}, \forall u \in \mathcal{A} \tag{3.2.7}$$

$$t_p^{iju} - z_p^u \leq 0, \qquad \forall p \in \mathcal{P}, \forall ij \in E, \forall u \in \mathcal{A} \tag{3.2.8}$$

$$t_p^{iju} - y_{ij} \leq 0, \qquad \forall p \in \mathcal{P}, \forall ij \in E, \forall u \in \mathcal{A} \tag{3.2.9}$$

$$y_{ij} - t_p^{iju} + z_p^u \leq 1, \qquad \forall p \in \mathcal{P}, \forall ij \in E, \forall u \in \mathcal{A} \tag{3.2.10}$$

$$x_p^k, z_p^u, y_{ij}, t_p^{iju} \in \{0,1\}, \qquad \forall p \in \mathcal{P}, \forall k = 1, \ldots, K, \forall ij \in E \tag{3.2.11}$$

$$w_u \geq 0, \qquad \forall u \in \mathcal{A}. \tag{3.2.12}$$

While considering a MANEG and a Nash equilibrium, a first necessary criterion is for a strategy to be feasible. This is guaranteed by Constraints (3.2.1) and (3.2.2), together

with the domain Constraints (3.2.11) and (3.2.12). Concretely, Constraints (3.2.1) assure that the capacity expansions chosen by the agents respect the maximum capacity bound; Constraints (3.2.2) enforce that the customer's strategy is a sharing policy. We add Constraint (3.2.3) to impose the ordering of the paths multiplicity. Due to Theorem 3.1.3, we are only interested in combinations of profitable increasing paths. Constraints (3.2.4) verify if a path satisfies the profitable increasing path constraint for a player $u \in \mathcal{A}$ by activating and deactivating the Boolean variable z_p^u, while Constraint (3.2.5) forbids the use of a non-profitable path in the path combination. Finally, we set the value of y_{ij} for $(i,j) \in E$ through Constraint (3.2.6): if a path is used, then all the arcs along it are used. These values are then used in Constraints (3.2.7) to make sure that no profitable path has been formed. This constraint is actually equivalent to the following:

$$\begin{aligned}
&\sum_{ij \in p \cap E_u} \left(y_{ij} - t_p^{iju}\right) \leq d_p^u - \frac{d_p^u}{|p|} \\
\iff &\sum_{ij \in p \cap E_u} \left(y_{ij} - z_p^u \times y_{ij}\right) \leq d_p^u - \frac{d_p^u}{|p|} \\
\iff &\left(1 - z_p^u\right) \sum_{ij \in p \cap E_u} y_{ij} \leq d_p^u - \frac{d_p^u}{|p|}.
\end{aligned}$$

The right-hand side needs a more detailed explanation: we want to ensure that if a path is not profitable increasing for an agent u, at least one of its arcs is not used. It might seem like we could have use the right-hand side $d_p^u - 1$, but this would have lead to infeasibility when $d_p^u = 0$. This specific case is covered with this more complex right-hand side.
Finally, Constraints (3.2.8) to (3.2.10) force t_p^{iju} to be equal to the product $z_p^u \times y_{ij}$.

The objective function sums the multiplicity of all used paths, which is the value of the total flow going through the network from o to d.

Combinatorial formulations like this one, with almost only binary variables, and an exponential number of both variables and constraints, are often challenging to solve and do not scale well. On the other hand, a lot of information from a MANEG can be exploited before the optimization. We will discuss in the next section the theoretical strengths and weaknesses of PATH_{IP} relatively to ARC_{IP}.

3.3. Discussion of formulations strengths and weaknesses

The most obvious flaw for PATH_{IP} is its potential large size, both in terms of variables and constraints. Large integer programs are generally slower to solve and, in particular, the enumeration of paths can become unpractical for large graphs. Then, one might advocate that generating all the paths and related data (e.g., d_p^u) requires too much preprocessing effort. However, that is true only for large and/or dense networks. For small and/or sparse graphs, we can gather the model parameters and enumerate paths efficiently, allowing us to see possible advantages of PATH_{IP} over ARC_{IP}.

Regardless, ARC_{IP} has a few weaknesses that PATH_{IP} does not have. We have shown in Table 2.1 that the use of a large value for the parameter M in ARC_{IP} can lead to numerical inaccuracies and eventually produce strategies that are not Nash equilibria. In PATH_{IP}, we do not have such big-M constraints. Furthermore, ARC_{IP} relies on a set of strict constraints, Constraints 2.2.18, which are also known to lead to numerical issues while solving mixed-integer programs.

Taking all these observations into consideration, we aim at counteract the drawbacks of each formulation and exploit all of their strengths as much as possible. This is the aim of the next chapter, where this objective is sought in the form of valid inequalities of two different types, as well as an hybrid formulation, merging both ARC_{IP} and PATH_{IP} in one single mathematical program denoted by HYB_{IP}.

Chapter 4

Improvements: Taking the best of both formulations

We ended Chapter 2 with a small computational example. At this occasion, we noticed the importance of fine tuning the parameter M for ARC_{IP}. In Section 4.1, we will finally discuss the tuning of this parameter. Then, we propose a set of valid inequalities to limit the impact of a badly chosen M on the correctness of the optimal solution. We will also propose valid inequalities (which can also be seen as a preprocessing step) to reduce the size of PATH_{IP}, its main weakness. In Section 4.2, we detail a third and last formulation, HYB_{IP}, formed by merging the arc and path-based mathematical programs.

4.1. Enhanced formulations

We start this section by discussing the tuning of the big-M parameter for ARC_{IP} in Section 4.1.1. Then, we present two key ideas to create valid inequalities for our problem. We give their explicit form for each formulation, ARC_{IP} and PATH_{IP}. The first one, described in Section 4.1.2, can be seen as a way to reduce the sensitivity of ARC_{IP} to numerical inaccuracies induced by big-M constraints. Our second idea, presented in Section 4.1.3, aims at reducing the size of PATH_{IP} as much as possible by exploiting all the MANEG data gathered during a prepossessing step. Even if these concepts were conceived to improve each specific formulation, we show in Chapter 5 that ARC_{IP} and PATH_{IP} can benefit from both families of valid inequalities.

4.1.1. Tuning M

In mathematical programming, big-M constraints are known to cause numerical imprecisions and to contribute to loose linear relaxations. This is what we observed in Table 2.1. As detailed in Section 2.2, M is designed to be a rebarbative cost on an arc in a residual graph. For example, let us consider a strategy $\mathcal{S}$, an agent $u \in \mathcal{A}$ and the existence of an arc

$(i,j) \in E_u$ with a null capacity expansion. In the residual graph associated with agent u, G_r^u, even though (j,i), the backward arc associated, exists, its residual cost will be, according to Constraint (2.2.12) of ARC_{IP}, M.

Actually, M is a parameter designed to make sure that paths going through impossible arcs (i.e., arcs with a capacity of 0, or forward arcs owned by a different agent) in the residual graphs are very expensive (and thus, not considered in a Nash equilibrium). We recall that for each $u \in \mathcal{A}$, (LP_u) is a min-cost flow in the residual graph. Thus, a theoretically correct M must have a value strictly greater than the most expensive path in the MANEG's network[1]. With this in mind, we can devise tight suitable M values. Depending on how much computational effort one would like to put into analyzing a MANEG, we could choose (from the tightest to the most computationally efficient) :

- For a player u, the highest cost for a path plus one, i.e., $M_u = 1 + \max_{p \in \mathcal{P}} \sum_{ij \in p \cap E_u} c_{ij}$;
- the cost of the most expensive path plus one $M = 1 + \max_{p \in \mathcal{P}} \sum_{ij \in p} c_{ij}$;
- For a player u, the sum of all her expansion costs plus one, i.e., $M_u = 1 + \sum_{ij \in E_u} c_{ij}$;
- the sum of all expansion costs[2], i.e., $M = \sum_{ij \in E} c_{ij}$.

4.1.2. Valid inequalities: No player has a negative profits

One of the main drawbacks of ARC_{IP} came from the big-M constraints, as highlighted in Table 2.1; in the associated example, we observed that the use in ARC_{IP} of a very large M in a MANEG where the trivial strategy is the unique Nash equilibrium, led to an incorrect solution. Puzzled by this technical challenge (theoretically, the use of a big-M is correct), we focus on the trivial strategy, which is a Nash equilibrium in any MANEG:

Proposition 4.1.1. *Any agent can unilaterally reach a reward of at least 0*

Proof. Due to our assumption stating that the minimum capacity for all arcs is zero, any player $u \in \mathcal{A}$ can set $q_{ij} = 0$ for $(i,j) \in E_u$. In this way, the player has a null cost. □

Corollary 4.1.2. *For each agent, the profit obtained from a strategy vector $\mathcal{S}$ corresponding to a Nash equilibrium achieving the maximum flow is non-negative.*

By Corollary 4.1.2, we can affirm that a constraint inferring that all agents have a positive or null profit is a valid inequality. Next, we lift the space of variables and we represent for

[1]Indeed, there is no need to consider path composed of both forward and backward arcs under the 0 lower bound expansion condition.

[2]Here we do not need to add a plus one, since we assumed no player owned a full path

both formulations the benefit achieved by each agent $u \in \mathcal{A}$ (i.e., the total part of reward the agent earns) with a new variable f_u. We start by describing the new inequalities in the context of the formulation ARC_{IP}. After detailing them, we propose an alternative way to express the same inequalities within PATH_{IP}.

In ARC_{IP}, the variables q_{ij} for each arc $(i,j) \in E$ provide us a direct access to each player strategy. In this way, we devise the following set of inequalities:

$$\pi w_u \overline{F} - f_u \geq 0, \qquad \forall u \in \mathcal{A} \tag{4.1.1}$$

$$f_u - \sum_{ij \in E_u} c_{ij} q_{ij} \geq 0, \qquad \forall u \in \mathcal{A} \tag{4.1.2}$$

$$\sum_{u \in \mathcal{A}} f_u - \pi F = 0 \tag{4.1.3}$$

$$f_u \geq 0, \qquad \forall u \in \mathcal{A}.$$

The revenue of each player $u \in \mathcal{A}$ in the game is $\pi w_u F$, thus the goal of the inequalities above is to approximate f_u to that value. Sadly, making $f_u = \pi w_u F$ is not desirable as it creates the bilinear term $w_u F$. To thwart this, we upper bound our variables (revenue) f_u with the maximum theoretical flow $\overline{F}$ in Constraints (4.1.1).[3] We also lower bound the revenue through the costs of the expansion strategy in Constraints (4.1.2); this forbids negative profits for the players. Furthermore, since $\sum_{u \in \mathcal{A}} w_u = 1$, we can precise the bounds on each revenue variable f_u with Constraints (4.1.3).

While Constraints (4.1.1) and (4.1.3) can be added directly to PATH_{IP}, this is not the case for Constraint (4.1.2) since PATH_{IP} has no variables q_{ij} for $(i,j) \in E$. Instead, we must use the binary variables describing providing the multiplicity of the paths. Constraint (4.1.2) is thus equivalent to

$$f_u - \sum_{ij \in E_u} c_{ij} \left(\sum_{p \in \mathcal{P} | ij \in p} \sum_{k=1}^{K} x_p^k \right) \geq 0, \qquad \forall u \in \mathcal{A}. \tag{4.1.4}$$

We designate the valid inequalities developed in this section by *non-negative profit cuts* (*noneg*).

4.1.3. Valid inequalities: Filtering

The main weakness of PATH_{IP} is its size, highly related with the preprocessing step enumerating the set of paths for the formulation. Our goal is to exploit as much as possible this step in order to reduce the size of the model. The idea is to forbid the exploitation

[3]The maximum theoretical flow is the maximum $o \sim d$ flow achieve when $q_{ij} = \overline{q}_{ij}, \forall (i,j) \in E$.

of any path that is non-profitable for any feasible sharing policy. A path $p \in \mathcal{P}$ fits this non-profitability characteristic if there is one agent $u \in \mathcal{A}$ such that:

$$\sum_{ij \in p \cap E_u} c_{ij} > \pi. \tag{4.1.5}$$

Since this condition was initially formulated with PATH_{IP} in mind. We start by describing the valid inequalities inspired by this condition for PATH_{IP}s. We thus propose the following inequalities for all paths p respecting Condition (4.1.5):

$$x_p^1 \leq 0. \tag{4.1.6}$$

In practice, we can simply remove from the definition of $\mathcal{P}$ for PATH_{IP}, the paths for which Condition (4.1.5) holds.

Now, we show how to convert Constraint (4.1.6) to ARC_{IP}. Remark that in PATH_{IP}, we use a set of variables, y, in common with ARC_{IP}. Recall that in PATH_{IP}, we use this set of variables to forbid some paths with Constraints (3.2.7). It is this constraint in PATH_{IP} that provide us a valid inequality for ARC_{IP}:

$$\sum_{ij \in p \cap E_u} y_{ij} \leq d_p^u - \frac{d_p^u}{|p|} \quad \forall p \in \mathcal{P}, \forall u \in A \text{ such that Condition (4.1.5) holds.} \tag{4.1.7}$$

We designate the valid inequalities developed in this section by *filtering the never-profitable paths cuts* (*filter*).

Even though filtering through all the paths can be tedious, we expect - for instances with a small reward π - these inequalities to reduce the significantly the feasible set, making the problem almost trivial. We saw on Table 2.2 that the valid inequalities described here, *noneg* and *filter*, could actually improve the performances.

4.2. Hybrid formulation

The second set of valid inequalities that we defined, i.e., the *filter* cuts, highlights an important relationship between the two formulations, ARC_{IP} and PATH_{IP}. Furthermore, we thought the previous valid inequalities to adorn specific extreme cases (namely low reward π and big parameter M). We also noticed during our experiments (detailed in Chapter 5) that PATH_{IP} out-performs ARC_{IP} for instances with a small reward, while the opposite occurs for high rewards. Our idea is that for instances with medium rewards, merging both formulation into one could be more efficient than having to choose between the two, and taking the risk of not picking the best one.

With that in mind, we define an hybrid formulation HYB_{IP} as follow[4]:

$$
\begin{aligned}
\max \quad & F \\
\text{s.t. } & (2.2.1) \text{ to } (2.2.18) \\
& (3.2.1) \text{ to } (3.2.10) \\
x_{ij},\alpha_{ij},y_{ij},\beta_{ij},\varphi^u_{ij},\varphi^u_{ji} &\in \mathbb{B}, \quad \forall (i,j) \in E, \forall u \in \mathcal{A} \\
F &\geq 0 \\
w_u &\geq 0, \forall u \in \mathcal{A} \\
q_{ij} &\in \mathbb{N}, \forall (i,j) \in E \\
\delta^{jiu}_B, \delta^{iju}_F, t^u_i, t^u_j &\in \mathbb{R}, \forall (i,j) \in E, \forall u \in \mathcal{A} \\
x^k_p, z^u_p, y_{ij}, t^{iju}_p &\in \{0,1\}, \qquad \forall p \in \mathcal{P}, \forall k = 1,\ldots,K, \forall ij \in E \\
w_u &\geq 0, \qquad \forall u \in \mathcal{A}.
\end{aligned}
$$

Now, we have two different formulations, ARC_{IP} and PATH_{IP}, as well as a third one combining them, HYB_{IP}. We also formulated two sets of valid inequalities applicable to all three formulations. In the next chapter, we demonstrate the value of all the individual enhancements as well as their combination described here on a set of instances based on the one used in Chaabane et al. [2017].

[4]All variables keep the same meaning as in the arc and path-based formulations.

Chapter 5

Computational experiments

In Chapter 3, we presented an alternative formulation for the computation of a Nash equilibrium achieving the maximum flow in a MANEG, PATH_{IP}. In Chapter 4, we described an hybrid formulation, HYB_{IP}, joining both the alternative formulation, PATH_{IP}, and the state-of-the-art one, ARC_{IP}. We also detailed two new families of valid inequalities in order to improve the performances related to solving those three formulations.

In Section 5.1, we detail implementation specifications and how we build a meaningful set of instances for the computational experiments. We then provide and discuss the results of our implementations in Section 5.2.

This chapter validates empirically the value of our contributions, enabling us to identify their individual advantages and joint synergies.

5.1. Experimental setup

Instances. Our work departs from the arc-based formulation and observations by Chaabane et al. [2017]. In this context, we mimic the generation of a set of instances similar to the one in Chaabane et al. [2017] by using the same generative tool RanGen 1, as developed in Demeulemeester et al. [2003].

To use RanGen 1, we must input the value of a parameter designated by order of strength (OS) and the number of nodes. As in Chaabane et al. [2017], each network has OS equal to 0.5. This parameter controls the number of arcs in the network. We build two sets of 30 networks, with $|V| \in \{10,50\}$. We focus on instances with 2 transport agents (i.e., $m = 2$).

Once a network is generated, the origin and destination nodes are determined as the first and last nodes generated. Moreover, for each $(i,j) \in E$ of a generated network, we assign uniformly at random in the indicated domains, the following parameters:

- The maximum capacity expansion $\overline{q}_{ij} \in [0,10]$;
- The unitary cost of expansion $c_{ij} \in [0,80]$;
- The ownership of the arc by agent $u \in \mathcal{A}$.

Each network produced as described above leads to 5 instances, differing on the value assigned to the total reward π:

$$\pi = \alpha \times c_{\max}$$

with $\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and $c_{\max} = \max_{p \in \mathcal{P}} \sum_{ij \in p} c_{ij}$, i.e., the most expensive $o \sim d$ path. In summary, for each graph size $|V| \in \{10{,}50\}$, we produce $150 = 5 \cdot 30$ instances.

Remark: The computational experiments in Chaabane et al. [2017] show that the computational time to solve ARC_{IP} increases with the number of players. This is expected since the formulation size increases with the number of players (particularly, the number of extra auxiliary variables needed in the linearization described in Appendix A). In fact, except for $\alpha = 0.1$, the authors report average computational times significantly higher than our time limit of 1800 seconds. Similarly, both the size of PATH_{IP} and HYB_{IP} increases with the number of players since the very definition of profitable increasing path depends on the set of players. This motivates our focus on instances with $m = 2$ in this chapter. Nevertheless, in Appendix B, we take the 150 instances with $|V| = 10$, we redistribute uniformly at random the arcs among $m = 5$ transport agents, and we provide detailed computational results. This allow us to see that the same trend of advantages and limitations reported for our contributions hold when the number of players increases, although with an unsurprising deterioration of the computational time.

Computational environment. All our code is implemented in Julia 1.7.1, and optimization problems are solved with Gurobi 9.5.0. We ran our experiments on an Intel Xeon Gold 6226 CPU processor at 2.70 GHz, running under Oracle Linux Server, restricting the optimization solver to one thread and a time limit of 1800 seconds.

Before proceeding with the experiments, we have to describe the computation of auxiliary elements for our formulations. First, we compute the big-M parameter for ARC_{IP}, as detailed in Section 4.1.1. In particular, we set $M = c_{\max} + 1$, i.e., the cost of the most expensive path from o to d plus one. Second, we compute a few more parameters to build PATH_{IP} and the valid inequalities. Those take a a negligible amount of resources in comparison to the optimization (and thus, they are not included within our time limit):

- Set $\mathcal{P}$ of all paths from o to d;
- For each path $p \in \mathcal{P}$ and each player $u \in \mathcal{A}$: $d_p^u = |p \cap E_u|$;
- The theoretical maximum flow $\overline{F}$ (needed for the *no-neg* valid inequalities).

5.2. Experimental results

In this section , we aim to evaluate the advantages and the limits of the contributions developed in this thesis. In Section 5.2.1, we first compare the three mixed-integer linear programming formulations, ARC_{IP}, PATH_{IP} and HYB_{IP}, and we identify their individual strengths. Then, in Section 5.2.2, we analyze under which conditions the valid inequalities are useful, i.e., they speed up computations.

In the following sections, we show the results obtained on our instances with $|V| = 50$. In the Appendix B, we provide detailed tables of results for all instances, including $|V| = 10$. The reason why we focus on these larger instances in the main body of the thesis is because they are harder to solve, consequently, allowing a more effective evaluation of our contributions, and because similar performance trends are observed for the remaining instances.

5.2.1. Formulations

At first, we compare the behavior of each formulation over our 5 categories, i.e., according to the generation of π. Recall that the generation of π depends on 5 possible values of the parameter α. For each of the 5 categories, we have 30 instances. We categorize instances according to the setup of the reward π because of the expected high impact of this value on the number of profitable increasing paths. In particular, the higher the parameter α, the higher the total reward is.

Figure 5.1 provides the performance plots for each formulation and value of α. Table 5.1 complements the performance plots with additional details on average computational time (CPU), average number of explored nodes in the branch-and-bound procedure (B & B nodes), and on the percentage of instances solved (% solved).

In Figure 5.1a, concerning ARC_{IP}, we see that as the reward (or α) increases, the more time is needed to solve the problem, with even some instances not being solved within the time limit. Interestingly, this behavior is not monotonic. When $\alpha = 0.9$, more instances are solved than for $\alpha \in \{0.5, 0.7\}$; actually the line corresponding to this category is the one solving the most instances within short running times. We hypothesize that when the reward is very high, the theoretical maximum flow can be achieved and such equilibria are easier to determine; indeed, the experiments by Chaabane et al. [2017], show that for such instances the flow corresponding to the determined equilibrium is close to the theoretical maximum flow. For PATH_{IP} and HYB_{IP}, we observe a significantly different behavior in Figures 5.1b and 5.1c. For these two formulations, the computational time increases as the reward increases. The easiest and fastest category has $\alpha = 0.1$. This behavior is expected

as low rewards must allow the formulations using paths to eliminate several (non-profitable) paths.

Note the variability in performance is lower for ARC_{IP} than for PATH_{IP} and HYB_{IP}. Thus, even though PATH_{IP} and HYB_{IP} are more efficient for instances with a small reward (i.e., more instances are solved within the time limit, as detailed in Table 5.1), ARC_{IP} appears as a better choice overall since it is less susceptible to be affected by α variations. Lastly, even if the performances of PATH_{IP} and HYB_{IP} seem similar in a lot of aspects, it is important to notice that during the branch-and-bound method, HYB_{IP} has a very small amount of nodes explored (this is visible in Table 5.1). This is because HYB_{IP} has a good linear relaxation in comparison with the other formulations. This is logical since HYB_{IP} is based on the arc and path-formulations.

Nodes		CPU			B&B nodes			% of instances solved		
$\|V\|$	α	ARC_{IP}	PATH_{IP}	HYB_{IP}	ARC_{IP}	PATH_{IP}	HYB_{IP}	ARC_{IP}	PATH_{IP}	HYB_{IP}
50	0.1	209.31	24.17	3.23	5885.86	932.70	0.00	0.97	1.00	1.00
	0.3	388.88	511.54	309.53	10767.71	920.46	0.70	0.80	0.80	0.90
	0.5	689.16	703.81	707.30	32273.91	885.00	3.00	0.37	0.20	0.27
	0.7	429.71	1182.01	684.08	46558.63	732.00	1.00	0.27	0.03	0.03
	0.9	24.08	NaN	678.89	1197.83	NaN	1.00	0.60	0.00	0.13

Table 5.1. Computational summary for each formulation

To complement the discussion on how to decide which formulation to use, in Figure 5.2, we present the performance profiles for each category of instances. This allows us to identify which formulation is better suited for each category. We can conclude that, as shown in Figures 5.2a and 5.2b, HYB_{IP} and PATH_{IP} are more efficient than ARC_{IP} for $\alpha = 0.1$ or $\alpha = 0.3$. Starting at $\alpha = 0.5$, the trend is reversed. It is very clear then in Figure 5.2d that ARC_{IP} gives better results for a high α.

In summary, the results presented here show that we already have quite fast results to solve instances with small rewards using PATH_{IP} and HYB_{IP}. Next, we aim to show that our enhancements presented in Chapter 4 improve the performance of the formulations over instances with medium-high rewards.

5.2.2. Valid inequalities

In Chapter 4, we introduced two families of valid inequalities, *noneg* and $filter$. In this section, we aim at showing how those valid inequalities can improve the performances of our three formulations. Detailed results can be found in Tables B.1 and B.2 of Appendix B where

we use Boolean variables to indicate the inequalities activated and the remaining rows follow the same meaning as in the previous section. In what follows, we take particular instance categories and formulations to support our analysis, referring the reader to Appendix B for additional results.

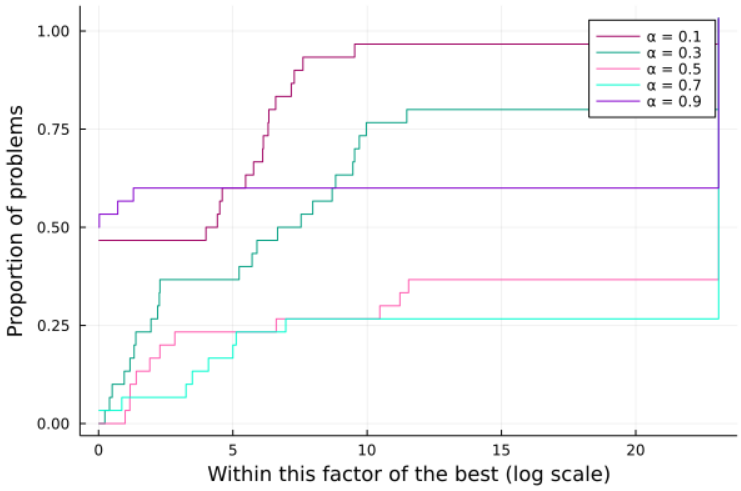
Figure 5.3 showcases the impact of each family of valid inequalities in each formulation. Since similar result trends are observed among the different categories, we provide use Figure 5.3 to showcase the impact of each family of valid inequalities in each formulation for the category of instances with $\alpha = 0.3$. Although the $filter$ inequalities were motivated as an improvement for PATH_{IP}, Figure 5.3b shows that the optimization of PATH_{IP} is not improved by any of the inequalities. Neither the proportion of problems solved, nor the solving time was improved. In Figure 5.3c, we observe the same tendency. However, ARC_{IP} seems to benefit significantly from all valid inequalities, as shown in Figure 5.3a. Both the solving time and the proportion of problems solved is increased.

Nodes $\lvert V\rvert$	α	noneg	filter	CPU ARC_{IP}	B & B nodes ARC_{IP}	% of instances solved ARC_{IP}
50	0.1	0	0	209.31	5885.86	0.97
		1	0	0.35	1.00	1.00
		0	1	0.21	0.17	1.00
		1	1	0.20	0.03	1.00
	0.3	0	0	388.88	10767.71	0.80
		1	0	50.25	7779.33	0.90
		0	1	4.08	1051.13	1.00
		1	1	3.43	869.93	1.00
	0.5	0	0	689.16	32273.91	0.37
		1	0	409.82	88993.70	0.33
		0	1	384.41	24200.24	0.70
		1	1	378.91	45968.90	0.67
	0.7	0	0	429.71	46558.63	0.27
		1	0	171.44	25764.57	0.23
		0	1	204.10	11101.00	0.47
		1	1	229.16	21628.20	0.50
	0.9	0	0	24.08	1197.83	0.60
		1	0	19.16	1422.50	0.60
		0	1	177.21	13947.95	0.73
		1	1	231.54	15577.43	0.77

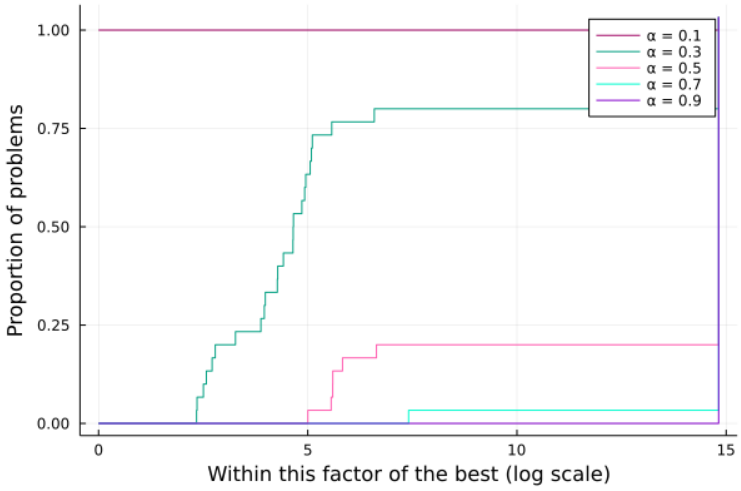
Table 5.2. Impact of valid inequalities on ARC_{IP}

In Table 5.2, we obtain additional insights on the impact of valid inequalities on ARC_{IP}. For α up to 0.7, the valid inequalities all together, and specifically the $filter$, decrease the

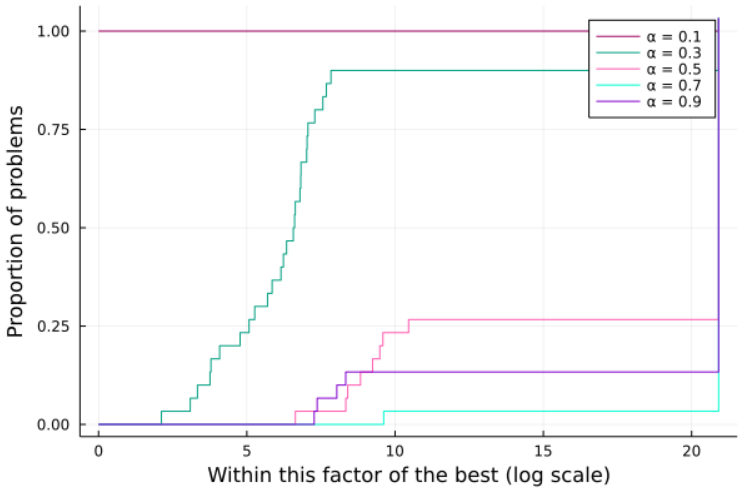
number of nodes in the branch and bound algorithm. For $\alpha = 0.9$, the opposite occurs, although one must keep in mind that the percentage of instances solved within the time limit increases.



(a) ARC_{IP} profile plot

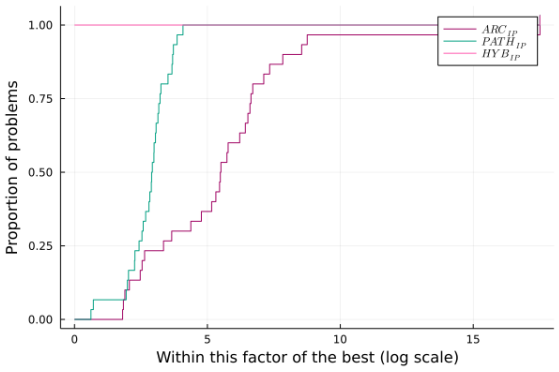


(b) PATH_{IP} profile plot

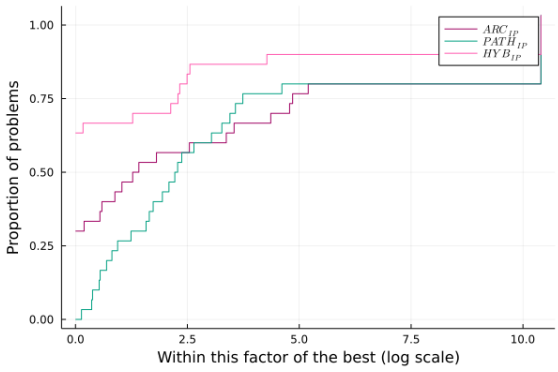


(c) HYB_{IP} profile plot

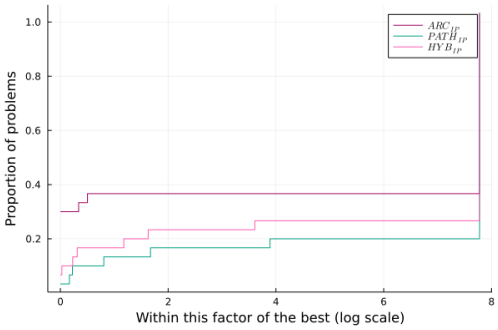
Figure 5.1. Performance profiles for each formulation



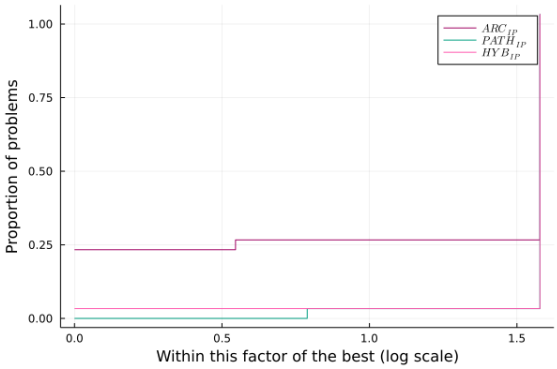
(a) Comparison of formulations for $\alpha = 0.1$



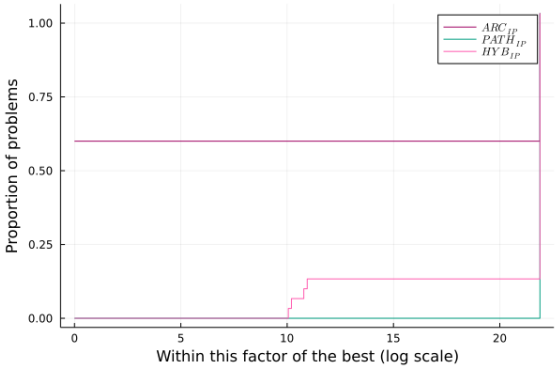
(b) Comparison of formulations for $\alpha = 0.3$



(c) Comparison of formulations for $\alpha = 0.5$

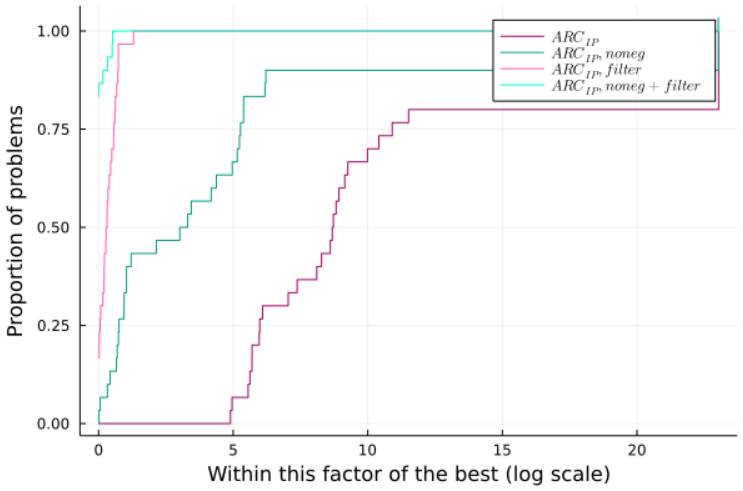


(d) Comparison of formulations for $\alpha = 0.7$

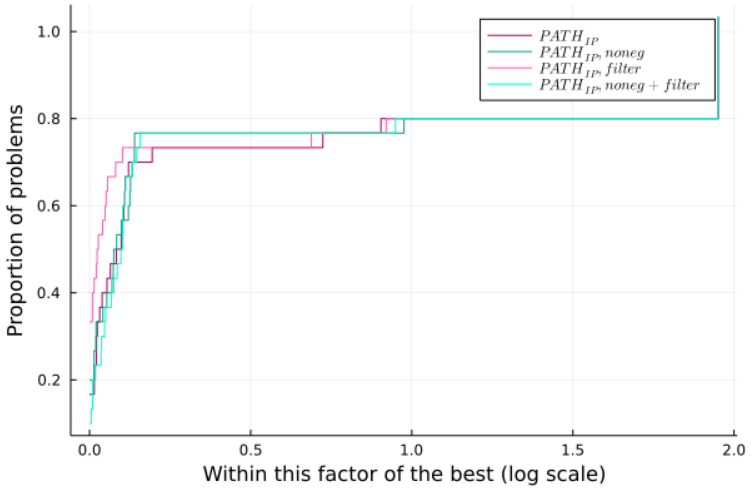


(e) Comparison of formulations for $\alpha = 0.9$

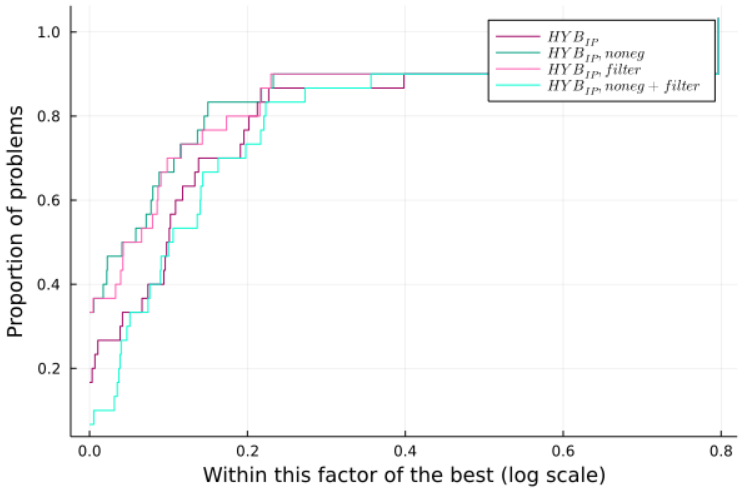
Figure 5.2. Performance profiles for each α



(a) ARC_{IP} profile plot



(b) PATH_{IP} profile plot



(c) HYB_{IP} profile plot

Figure 5.3. Performance profiles for each formulation and activation set of valid inequalities

Chapter 6

Conclusions and future work

Conclusions. In this thesis, we tackle the multi-agent network expansion game. This is a game played by set of carriers and a customer. The customer aims to maximize the flow traversing a network from an origin to a destination and she controls the proportion of reward allocated to each player. In turn, each carrier decides the capacity expansion of each of her arcs and aims to maximize individual profits. Our goal is to compute a strategy vector for the players corresponding to a Nash equilibrium attaining the maximum flow.

We recapped the state-of-the-art arc-based formulation, ARC_{IP}, and we identified its drawbacks. Then, we formalized the concept of profitable increasing path and we showed sufficient and necessary conditions based on it to characterize Nash equilibria. This led us to a novel path-based formulation (PATH_{IP}). Motivated by the exponential size of the latter, we developed a new family of valid inequalities for both formulations to filter paths that are never profitable. We also proposed a new set of valid cuts on the lifted space of profits with the aim of modeling that each player revenue must be superior or equal to the incurred costs. Additionally, we described procedures to fine-tune the big-M in ARC_{IP} and we devised an hybrid arc-path based formulation (HYB_{IP}). Finally, we evaluated through computational experiments, the three formulations as well as the valid inequalities. We were able to show categories of instances where HYB_{IP} and ARC_{IP} dominate. With regards to the valid inequalities, ARC_{IP} was by far the formulation benefiting the most from their inclusion. Indeed, not only were instances solved faster previously un-solved instances became solvable within the time limit. Interestingly, the reward seems to one of the main elements controlling the game complexity given its direct effect on the performance of the different formulations.

In conclusion, the contributions provided in this thesis bring not only new theoretical characterizations of the game but also lead to new integer programming formulations that can be solved more efficiently in practice. Moreover, our work has the potential to provide building stones for future work as we describe next.

Future work. The current work strongly relies on the assumption that $\underline{\mathcal{Q}} = \mathbf{0}$. However, the concepts developed here can be enriched to fit more general problems, namely problems free from this assumption, by representing the cost expansion with stair functions. Additionally, considering MANEGs to be sequential games, making the customer the leader in a Stackelberg fashion, seems like an interesting direction to follow. This angle could allow not only to increase the flow, but also to better model the practical situation between the customer and agents. Finally, the inequalities added to PATH_{IP} make it suitable for the application of a column generation framework, generating profitable increasing paths to solve its linear relaxation. The pricing problem is based a linear program obtained by dropping the non-unprofitable path constraints in the linear relaxation of PATH_{IP}, and using the paths reduced costs in the objective. These results are useful for instances where the root gap is small, which happens to be the instances were the path-based formulation struggles relatively to the arc-based formulation. It can also be a powerful tool for large instances, as such algorithm does not need all $o \sim d$ paths as parameters.

Bibliography

A. Agnetis, C. Briand, J.-C. Billaut, and P. Sůcha. Nash equilibria for the multi-agent project scheduling problem with controllable processing times. *Journal of Scheduling*, 18(1):pp.15–27, 2015. doi: 10.1007/s10951-014-0393-x. URL `https://hal.archives-ouvertes.fr/hal-01231028`.

S. Burer. On the copositive representation of binary and continuous nonconvex quadratic programs. *Mathematical Programming*, 120(2):479–495, Sept. 2009. ISSN 1436-4646. doi: 10.1007/s10107-008-0223-z. URL `https://doi.org/10.1007/s10107-008-0223-z`.

M. Carvalho. *Computation of equilibria on integer programming games*. PhD thesis, Faculdade de Ciências da Universidade do Porto, 2016.

M. Carvalho, A. Lodi, J. P. Pedroso, and A. Viana. Nash equilibria in the two-player kidney exchange game. *Mathematical Programming*, 161(1):389–417, Jan 2017. ISSN 1436-4646. doi: 10.1007/s10107-016-1013-7. URL `https://doi.org/10.1007/s10107-016-1013-7`.

M. Carvalho, A. Lodi, and J. P. Pedroso. Existence of Nash Equilibria on Integer Programming Games. In A. I. F. Vaz, J. P. Almeida, J. F. Oliveira, and A. A. Pinto, editors, *Operational Research*, Springer Proceedings in Mathematics & Statistics, pages 11–23, Cham, 2018. Springer International Publishing. ISBN 9783319715834. doi: 10.1007/978-3-319-71583-4_2.

M. Carvalho, G. Dragotto, A. Lodi, and S. Sankaranarayanan. The Cut and Play Algorithm: Computing Nash Equilibria via Outer Approximations. *arXiv:2111.05726 [cs, math]*, Nov. 2021. URL `http://arxiv.org/abs/2111.05726`. arXiv: 2111.05726.

M. Carvalho, A. Lodi, and J. P. Pedroso. Computing equilibria for integer programming games. *European Journal of Operational Research*, Mar. 2022. ISSN 0377-2217. doi: 10.1016/j.ejor.2022.03.048. URL `https://www.sciencedirect.com/science/article/pii/S0377221722002727`.

N. Chaabane, C. Briand, M.-J. Huguet, and A. Agnetis. Finding a Nash equilibrium and an optimal sharing policy for multiagent network expansion game. *Networks*, 69(1):94–109, 2017. doi: 10.1002/net.21711. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/net.21711`. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/net.21711.

H.-L. Chen and T. Roughgarden. Network design with weighted players. In *Proceedings of the eighteenth annual ACM symposium on Parallelism in algorithms and architectures*, SPAA '06, pages 29–38, New York, NY, USA, July 2006. Association for Computing Machinery. ISBN 9781595934529. doi: 10.1145/1148109.1148114. URL https://doi.org/10.1145/1148109.1148114.

E. Demeulemeester, M. Vanhoucke, and W. Herroelen. RanGen: A Random Network Generator for Activity-on-the-Node Networks. *Journal of Scheduling*, 6(1):17–38, Jan. 2003. ISSN 1099-1425. doi: 10.1023/A:1022283403119. URL https://doi.org/10.1023/A:1022283403119.

G. Dragotto and R. Scatamacchia. ZERO Regrets Algorithm: Optimizing over Pure Nash Equilibria via Integer Programming. *arXiv:2111.06382 [cs, math]*, Nov. 2021. URL http://arxiv.org/abs/2111.06382. arXiv: 2111.06382.

P. C. Gilmore and R. E. Gomory. A Linear Programming Approach to the Cutting-Stock Problem. *Operations Research*, 9(6):849–859, Dec. 1961. ISSN 0030-364X. doi: 10.1287/opre.9.6.849. URL https://pubsonline.informs.org/doi/abs/10.1287/opre.9.6.849.

C. Guo, M. Bodur, and J. A. Taylor. Copositive Duality for Discrete Markets and Games. *arXiv:2101.05379 [econ, math]*, Jan. 2021. URL http://arxiv.org/abs/2101.05379. arXiv: 2101.05379.

T. Harks and J. Schwarz. Generalized Nash Equilibrium Problems with Mixed-Integer Variables. *arXiv:2107.13298 [cs, math]*, Feb. 2022. URL http://arxiv.org/abs/2107.13298. arXiv: 2107.13298.

M. Köppe, C. T. Ryan, and M. Queyranne. Rational Generating Functions and Integer Programming Games. *Operations Research*, 59(6):1445–1460, Dec. 2011. ISSN 0030-364X. doi: 10.1287/opre.1110.0964. URL https://pubsonline.informs.org/doi/abs/10.1287/opre.1110.0964.

J. F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36(1):48–49, 1950. doi: 10.1073/pnas.36.1.48. URL https://www.pnas.org/doi/abs/10.1073/pnas.36.1.48.

T. Roughgarden. *Routing Games*, page 461–486. Cambridge University Press, 2007. doi: 10.1017/CBO9780511800481.020.

S. Sagratella. Computing all solutions of nash equilibrium problems with discrete strategy sets. *SIAM Journal on Optimization*, 26(4):2190–2218, 2016. doi: 10.1137/15M1052445. URL https://doi.org/10.1137/15M1052445.

S. Sagratella, M. Schmidt, and N. Sudermann-Merx. The noncooperative fixed charge transportation problem. *European Journal of Operational Research*, 284(1):373–382, July 2020. ISSN 0377-2217. doi: 10.1016/j.ejor.2019.12.024. URL https://www.sciencedirect.com/science/article/pii/S0377221719310483.

T. Sandholm, A. Gilpin, and V. Conitzer. Mixed-integer programming methods for finding nash equilibria. In *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 2*, AAAI'05, page 495–501. AAAI Press, 2005. ISBN 157735236x.

E. Tardos and T. Wexler. *Network Formation Games and the Potential Function Method*, page 487–516. Cambridge University Press, 2007. doi: 10.1017/CBO9780511800481.021.

Appendix A

Linearization of Constraints (2.2.17)

In the arc based formulation proposed by Chaabane et al. [2017], the strong duality constraint describing the cost in the min-cost flow residual graph is non-linear.

$$t_0^u - t_{n+1}^u \geq \sum_{ij \in E_F} \varphi_{ij}^u \delta_F^{iju} + \sum_{ji \in E_B} \varphi_{ji}^u \delta_B^{jiu}, \quad \forall u \in \mathcal{A}. \tag{2.2.17}$$

Since φ are binary variables, and the residual costs δ are bounded, we can use a standard linearization methods. First, we set

$$\begin{aligned} \mu_{ij}^u &= \varphi_{ij}^u \delta_F^{iju}, && \forall (i,j) \in E_F, u \in \mathcal{A} \\ \nu_{ji}^u &= \varphi_{ji}^u \delta_B^{jiu}, && \forall (j,i) \in E_B, u \in \mathcal{A}. \end{aligned}$$

Furthermore, we know:

$$\begin{aligned} 0 \leq \delta_F^{iju} \leq 2M, &\forall (i,j) \in E_F, u \in \mathcal{A} \\ -c_{ij} \leq \delta_B^{jiu} \leq M, &\forall (j,i) \in E_B, u \in \mathcal{A}. \end{aligned}$$

Therefore Constraints (2.2.17) can be replaced with:

$$\begin{aligned} t_0^u - t_{n+1}^u \geq \sum_{ij \in E_F} \mu_{ij}^u + \sum_{ji \in E_B} \nu_{ji}^u, & \quad \forall u \in \mathcal{A} \\ \mu_{ij}^u \leq 2M \varphi_{ij}^u & \quad \forall (i,j) \in E, \forall u \in \mathcal{A} \\ \mu_{ij}^u \leq \delta_F^{ij,u} & \quad \forall (i,j) \in E, \forall u \in \mathcal{A} \\ \delta_F^{ij,u} - \mu_{ij}^u \leq 2M(1 - \varphi_{ij}^u), & \quad \forall (i,j) \in E, \forall u \in \mathcal{A} \\ \nu_{ji}^u \leq M \varphi_{ji}^u, & \quad \forall (i,j) \in E, \forall u \in \mathcal{A} \\ \nu_{ji}^u \leq \delta_B^{ji,u}, & \quad \forall (i,j) \in E, \forall u \in \mathcal{A} \\ \delta_B^{ji,u} - \nu_{ji}^u \leq M(1 - \varphi_{ji}^u), & \quad \forall (i,j) \in E, \forall u \in \mathcal{A}. \end{aligned}$$

Appendix B

Detailed results

Nodes				CPU			B & B nodes			% of instances solved		
$\|V\|$	α	noneg	filter	ARC_{IP}	$PATH_{IP}$	HYB_{IP}	ARC_{IP}	$PATH_{IP}$	HYB_{IP}	ARC_{IP}	$PATH_{IP}$	HYB_{IP}
10	0.1	0	0	0.02	0.00	0.01	0.93	14.00	0.00	1.00	1.00	1.00
		1	0	0.01	0.00	0.01	0.00	14.00	0.00	1.00	1.00	1.00
		0	1	0.01	0.00	0.01	0.00	14.00	0.00	1.00	1.00	1.00
		1	1	0.01	0.00	0.01	0.00	14.00	0.00	1.00	1.00	1.00
	0.3	0	0	0.03	0.02	0.02	5.30	14.00	0.00	1.00	1.00	1.00
		1	0	0.01	0.02	0.01	0.00	14.00	0.00	1.00	1.00	1.00
		0	1	0.01	0.02	0.01	0.90	14.00	0.00	1.00	1.00	1.00
		1	1	0.01	0.02	0.01	0.00	14.00	0.00	1.00	1.00	1.00
	0.5	0	0	0.06	0.04	0.05	13.00	14.00	0.73	1.00	1.00	1.00
		1	0	0.06	0.04	0.05	7.70	14.00	0.70	1.00	1.00	1.00
		0	1	0.03	0.05	0.05	2.53	14.00	0.70	1.00	1.00	1.00
		1	1	0.03	0.05	0.05	2.20	14.00	0.70	1.00	1.00	1.00
	0.7	0	0	0.08	0.07	0.10	18.70	14.00	1.00	1.00	1.00	1.00
		1	0	0.06	0.07	0.10	11.23	14.00	1.00	1.00	1.00	1.00
		0	1	0.05	0.08	0.10	18.10	14.00	1.00	1.00	1.00	1.00
		1	1	0.05	0.09	0.10	5.53	14.00	1.00	1.00	1.00	1.00
	0.9	0	0	0.03	0.14	0.12	4.00	14.00	1.00	1.00	1.00	1.00
		1	0	0.02	0.12	0.12	2.33	14.00	1.00	1.00	1.00	1.00
		0	1	0.03	0.14	0.12	3.27	14.00	1.00	1.00	1.00	1.00
		1	1	0.02	0.12	0.13	2.33	14.00	1.00	1.00	1.00	1.00
50	0.1	0	0	209.31	24.17	3.23	5885.86	932.70	0.00	0.97	1.00	1.00
		1	0	0.35	25.34	3.14	1.00	932.70	0.00	1.00	1.00	1.00
		0	1	0.21	25.04	3.33	0.17	932.70	0.00	1.00	1.00	1.00
		1	1	0.20	25.66	3.09	0.03	932.70	0.00	1.00	1.00	1.00
	0.3	0	0	388.88	511.54	309.53	10767.71	920.46	0.70	0.80	0.80	0.90
		1	0	50.25	506.00	304.66	7779.33	920.46	0.74	0.90	0.80	0.90
		0	1	4.08	506.66	311.23	1051.13	920.46	0.70	1.00	0.80	0.90
		1	1	3.43	504.19	318.58	869.93	920.46	0.74	1.00	0.80	0.90
	0.5	0	0	689.16	703.81	707.30	32273.91	885.00	3.00	0.37	0.20	0.27
		1	0	409.82	789.68	764.44	88993.70	843.60	0.88	0.33	0.17	0.27
		0	1	384.41	830.18	739.35	24200.24	885.00	4.50	0.70	0.20	0.27
		1	1	378.91	806.47	731.35	45968.90	843.60	0.88	0.67	0.17	0.27
	0.7	0	0	429.71	1182.01	684.08	46558.63	732.00	1.00	0.27	0.03	0.03
		1	0	171.44	1281.40	561.66	25764.57	732.00	1.00	0.23	0.03	0.03
		0	1	204.10	1189.29	649.62	11101.00	732.00	1.00	0.47	0.03	0.03
		1	1	229.16	1329.76	549.18	21628.20	732.00	1.00	0.50	0.03	0.03
	0.9	0	0	24.08	NaN	678.89	1197.83	NaN	1.00	0.60	0.00	0.13
		1	0	19.16	NaN	766.08	1422.50	NaN	1.00	0.60	0.00	0.17
		0	1	177.21	NaN	803.80	13947.95	NaN	1.00	0.73	0.00	0.17
		1	1	231.54	NaN	845.32	15577.43	NaN	1.00	0.77	0.00	0.17

Table B.1. Detailed results with $m = 2$

Nodes				CPU			B & B nodes			% of instances solved		
$\|V\|$	α	noneg	filter	ARC_{IP}	PATH_{IP}	HYB_{IP}	ARC_{IP}	PATH_{IP}	HYB_{IP}	ARC_{IP}	PATH_{IP}	HYB_{IP}
10	0.1	0	0	0.08	0.02	0.01	4.77	0.00	0.00	1.00	1.00	1.00
		1	0	0.02	0.02	0.01	0.00	0.00	0.00	1.00	1.00	1.00
		0	1	0.01	0.02	0.01	0.00	0.00	0.00	1.00	1.00	1.00
		1	1	0.01	0.02	0.01	0.00	0.00	0.00	1.00	1.00	1.00
	0.3	0	0	0.13	0.11	0.05	15.10	0.00	0.00	1.00	1.00	1.00
		1	0	0.03	0.11	0.05	1.00	0.00	0.00	1.00	1.00	1.00
		0	1	0.05	0.17	0.05	2.53	0.00	0.00	1.00	1.00	1.00
		1	1	0.02	0.17	0.05	0.03	0.00	0.00	1.00	1.00	1.00
	0.5	0	0	0.19	0.18	0.19	74.47	1.00	0.77	1.00	1.00	1.00
		1	0	0.19	0.18	0.20	52.17	1.00	1.00	1.00	1.00	1.00
		0	1	0.15	0.18	0.19	47.90	1.00	0.77	1.00	1.00	1.00
		1	1	0.16	0.19	0.20	35.33	1.00	1.00	1.00	1.00	1.00
	0.7	0	0	0.29	0.36	0.55	197.17	1.00	1.00	1.00	1.00	1.00
		1	0	0.23	0.38	0.64	84.93	1.00	1.00	1.00	1.00	1.00
		0	1	0.29	0.36	0.55	195.40	1.00	1.00	1.00	1.00	1.00
		1	1	0.24	0.38	0.65	88.23	1.00	1.00	1.00	1.00	1.00
	0.9	0	0	0.25	0.73	0.60	63.43	1.00	1.00	1.00	1.00	1.00
		1	0	0.20	0.68	0.75	30.67	1.00	1.00	1.00	1.00	1.00
		0	1	0.25	0.73	0.59	63.43	1.00	1.00	1.00	1.00	1.00
		1	1	0.20	0.68	0.74	30.67	1.00	1.00	1.00	1.00	1.00

Table B.2. Detailed results with $m = 5$