On Selfish Network Creation

DISSERTATION

zur Erlangung des akademischen Grades

doctor rerum naturalium (Dr. rer. nat.)
im Fach Informatik

eingereicht an der
Mathematisch-Naturwissenschaftlichen Fakultät II
Humboldt-Universität zu Berlin

von
Dipl.-Inf. Pascal Lenzner

Präsident der Humboldt-Universität zu Berlin:
Prof. Dr. Jan-Hendrik Olbertz

Dekan der Mathematisch-Naturwissenschaftlichen Fakultät II:
Prof. Dr. Elmar Kulke

Gutachter:

1. Prof. Dr. Susanne Albers
2. Prof. Dr. Tobias Harks
3. Prof. Dr. Guido Schäfer

Tag der mündlichen Prüfung: 30. Mai 2014

Abstract

The subject of study in this thesis is a game-theoretic model for decentralized network creation by selfish agents. These agents aim to create a connected network among themselves which maximizes their individual connection quality. Links in the network are costly and therefore agents try to find a trade-off between their cost spent on creating edges and their cost incurred by communicating within the network. This model was proposed a decade ago by Fabrikant, Luthra, Maneva, Papadimitriou and Shenker [FLM+03] with the goal of understanding real networks which emerge from the interaction of selfish entities without explicit central coordination, e.g. the Internet or social networks. We contribute to this research endeavor in many ways by considering these so-called Network Creation Games from three perspectives.

Our first point of view on these games is the approximation perspective. We analyze which networks are created by very simple computationally bounded selfish agents and how these networks compare to networks built by agents having unlimited computational resources. If the individual connection quality of an agent is measured with the sum of shortest path distances to all other agents, then we find that simple agents create networks which are remarkably close to networks created by supernatural agents. On the other hand, if the individual connection quality of an agent is measured with the maximum over the shortest path distances to all other agents, then we get the contrasting result that this only holds for tree networks and that other created networks may be very far away from the optimum.

The second point of view is the dynamics perspective. We turn the model into a sequential version and focus on the process of selfish network creation. For this, we investigate whether natural dynamics like best response dynamics are guaranteed to converge to an equilibrium of the game and if so, how this convergence process may be sped up. The results are diverse: If agents are restricted to performing edge-swaps then we have guaranteed and very fast convergence if the initial network is a tree. On the other hand, if the initial network is not a tree or if agents may perform richer strategy-changes, then we can show that this process may never converge. We contrast these mostly negative theoretical results with a careful empirical study. There, we observe reliable and surprisingly fast convergence towards equilibrium networks. Thus, despite our negative results, such egoistic self stabilization processes may be a promising practical approach for finding equilibrium networks.

We complete the treatment of Network Creation Games with our third point of view: the structure perspective. The individual quality of a selfishly created network for the agents is determined by its structure. Thus, a rigorous understanding of the shape of equilibrium networks seems necessary to understand the deterioration in quality due to the selfishness of the agents - the so-called Price of Anarchy. We provide new structural insights for several equilibrium concepts and introduce new tools which shed light on the structure of equilibrium networks for high edge-cost.

Zusammenfassung

Untersuchungsgegenstand dieser Arbeit ist ein spieltheoretisches Modell für die dezentrale Erzeugung von Netzwerken durch eigennützige Agenten. Diese Akteure verfolgen das Ziel, ein zusammenhängendes Netzwerk aufzubauen, welches ihre individuelle Verbindungsqualität maximiert. Direktverbindungen im Netzwerk haben Kosten, weshalb die Agenten ihre Ausgaben für das Erstellen von Direktverbindungen und die damit erzielten Kommunikationskosten ausbalancieren müssen. Dieses Modell wurde vor einem Jahrzehnt von Fabrikant, Luthra, Maneva, Papadimitriou und Shenker [FLM$^+$03] eingeführt, um reale Netzwerke, welche aus der Interaktion von eigenützigen Parteien entstanden sind, zu verstehen. Zu solchen Netzwerken zählen das Internet und auch soziale Netzwerke. Die vorliegende Arbeit trägt zu diesem Forschungsvorhaben bei, indem die sogenannten Network Creation Games aus drei Perspektiven betrachtet werden.

Die erste Sichtweise ist die Approximationsperspektive. Es wird untersucht, welche Netzwerke von sehr einfachen, in ihrer Berechnungsstärke eingeschränkten Agenten erzeugt werden und wie diese im Vergleich mit Netzwerken von Agenten, die beliebige Berechnungsstärke haben, abschneiden. Wird die individuelle Kommunikationsqualität eines Agenten mit der Summe der Distanzen der kürzesten Pfade von diesem Agent zu allen anderen Agenten gemessen, dann wird zeigt, dass die Netzwerke sehr einfacher Agenten den Netzwerken von unbeschränkten Agenten bemerkenswert nah kommen. Andererseits, falls das Maximum der Distanzen der kürzesten Pfade die Kommunikationsqualität bestimmt, dann wird das kontrastierende Ergebnis, dass nur Baumtopologien diese Eigenschaft haben und andere Netzwerke sehr weit vom Optimum entfernt sein können, bewiesen.

Als zweite Sichtweise wird die Dynamikperspektive betrachtet. Dazu werden sequentielle Versionen des Modells definiert und anhand dieser wird explizit der Prozess der Netzwerkerzeugung untersucht. Die Hauptfragestellung ist, ob unter der natürlichen Annahme, dass Agenten stets ihre Situation verbessern wollen, der Prozess zu einem Gleichgewicht konvergiert und, falls dem so ist, wie dieser Prozess beschleunigt werden kann. Die präsentierten Ergebnisse hierzu sind vielfältig. Falls die Agenten nur Kantenvertauschungen ausführen dürfen, dann ist Konvergenz garantiert, falls mit einer Baumtopologie begonnen wird. Andererseits, falls die Starttopologie kein Baum ist oder falls die Agenten aufwändigere Strategieänderungen ausführen können, dann gibt es keine Konvergenzgarantie. Diese hauptsächlich negativen theoretischen Resultate werden mit einer sorgfältigen empirischen Studie kontrastiert. Diese Studie zeigt in allen Fällen zuverlässige und sehr schnelle Konvergenz zu einem spieltheoretischen Gleichgewicht. Dies legt nahe, dass solche Dynamiken, trotz der fehlenden Konvergenzgarantie, ein vielversprechender Ansatz sind, um Gleichgewichte zu finden.

Die Abhandlung wird mit der dritten Sichtweise, der Strukturperspektive, abgerundet. Die individuelle Kommunikationsqualität eines Agenten in einem eigennützig erzeugten Netzwerk wird durch die Struktur des Netzwerks bestimmt. Deshalb erscheint es notwendig die strukturellen Eigenschaften solcher Netzwerke zu verstehen, um den Qualitätsverlust durch das eigennützige Verhalten der Agenten - den sogenannten Preis der Anarchie - ermitteln zu können. Es werden eine Vielfalt neuer Struktureigenschaften für verschiedene Gleichgewichtskonzepte bewiesen und neue Werkzeuge, die bei der Analyse von Gleichgewichtsnetzwerken mit hohen Direktverbindungskosten hilfreich sind, vorgestellt.

This thesis is based on the following publications:

1. **P. Lenzner:** *On Dynamics in Basic Network Creation Games.*
   4th Symposium on Algorithmic Game Theory (SAGT), 2011.
   [Len11]
2. **P. Lenzner:** *Greedy Selfish Network Creation.*
   8th Workshop on Internet & Network Economics (WINE), 2012.
   [Len12]
3. **B. Kawald and P. Lenzner:** *On Dynamics in Selfish Network Creation.*
   25th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), 2013.
   [KL13]
4. **P. Lenzner:** *New Tools for Network Creation Games.*
   Submitted, 2013.

Acknowledgements

I am grateful to many people who supported me in various ways during my time working on this thesis.

First, I like to thank my advisor Prof. Susanne Albers for giving me the opportunity to work in her research group, for her guidance and support and for giving me the freedom to shape and pursue my own research interests. Furthermore, I like to thank my additional referees Prof. Tobias Harks and Prof. Guido Schäfer for reviewing this thesis and for providing their expert opinion on my work. I am also grateful to André Koschmieder, Prof. Johannes Köbler, Prof. Louchka Popova-Zeugmann and Prof. Klaus Reinhardt for their willingness to serve in my committee.

It was a pleasure to work with all my colleaques at Humboldt University Berlin. I thank Antonios Antoniadis, Caroline Domscheidt, Nils Goldammer, Matthias Hellwig, Chien-Chung Huang, Falk Hüffner, Michael Jung, Bernd Kawald, Matthias Killat, Carsten Moldenhauer, Ralf Oelschlägel, Achim Passen, Yury Person, Eva Sandig and Alexander Souza for their interest in my work, their support and for many enlightening discussions.

I like to thank Sarah Möckel and Achim Passen for proof-reading this thesis and for many suggestions which improved the exposition.

Last but not least, I am indebted to my family for their backup and I am deeply grateful to my beloved spouse Marie for her enduring support, trust and inspiration.

Contents

1. Introduction

1.1. Motivation and Context

This thesis focuses on various properties of communication networks which are built by selfish agents in a decentralized way without explicit coordination among the agents. Studying communication networks is a classical and still very active field in the realm of Theoretical Computer Science and Operations Research. This area, called Network Design, turned out to be a procreative and influential research direction which led within the last sixty years to many beautiful and non-trivial combinatorial and algorithmic insights in networks. Famous examples are the Max-Flow-Min-Cut Duality, algorithms for computing (approximately) optimal communication structures like Minimum Spanning Trees and Steiner Trees or for finding interesting network nodes, as in Facility Location, k-Median and k-Center Problems.

Because of this research effort it is now well-known how to design networks under various side constraints and under various objective functions. But despite all this huge body of knowledge there is this curious fact: One of the most important communication networks which is increasingly shaping our everyday life – the Internet – cannot be fully explained by classical Network Design theory. The reason is that unlike classical centrally designed and optimized networks the Internet was and still is created by a multitude of selfish agents (e.g. Internet Service Providers), who control and modify varying sized portions of the network structure ("autonomous systems") in a selfish way to suit their needs. This decentralized and egoistic nature is an obstacle to approaching the design and analysis of the Internet as a classical Network Design optimization problem.

But the situation is not hopeless - it turned out that to tackle such problems the focus has to be broadened to the independently established field of Game Theory. Originally studied mostly by economists and sociologists, classical Game Theory provides the tools for analyzing the process and the outcomes of strategically interacting selfish agents.[1] The powerful idea of combining algorithmic insights with game theoretic settings has led to the creation of the now thriving research area called Algorithmic Game Theory.

From an Algorithmic Game Theory point of view the Internet can be considered as an equilibrium state of a strategic game played by selfish agents. This can be seen as follows: We have that each selfish agent faces classical Network Design problems,

[1] This connection is summarized in Papadimitriou's overview article [Pap01].

i.e. minimizing the cost of connecting the *own* network to the rest of the Internet while ensuring a high quality of service. In consequence agents choose strategies to cope with these problems. The Internet itself can then be understood as the outcome of the (repeated) interplay of such local and egoistic strategies.

Within the last decade several such games have been proposed and analyzed. In this thesis, we will focus on the line of works which consider the so-called Network Creation Games, as introduced by Fabrikant, Luthra, Maneva, Papadimitriou and Shenker [FLM+03]. These strategic games are very simple but they contain an interesting trade-off between an agent's investment in infrastructure and her obtained usage quality. Agents aim to invest as little as possible but at the same time they want to achieve a good connection to all other agents in the network.

Understanding such simple models of decentralized selfish network creation without coordination among the agents can be seen as the first step towards rigorously understanding real networks like the Internet or the multitude of existing social networks. We believe that this knowledge will lead on the one hand to an improved measurement and better maintenance of existing networks and on the other hand to better mechanisms which locally guide the agents towards globally better states.

1.2. Structure and Outline of this Thesis

In the following Chapter 2 we briefly introduce some important game-theoretical notions and give a detailed formal definition of the model we study in the rest of this thesis. It is explained how the outcomes of the game-theoretic model are evaluated and which other closely related models have been studied so far.

In Chapter 3 we investigate what happens when very simple agents act in our model. Unfortunately, computing a best possible strategy in the Network Creation Game is NP-hard. Hence, agents which cannot afford exponential running time have to resort to an approximation of the best possible strategy. These weaker strategies induce a weaker solution concept, the Greedy Equilibrium, an we analyze how close networks in Greedy Equilibrium are to networks in Nash Equilibrium.

The dynamics of the network creation process induced by the variants of the Network Creation Games are studied in Chapter 4. There we will consider sequential-move versions of our model where in every step one agent is allowed to change her strategy. The main question is whether such dynamics eventually converge to a (swap-)stable network and if so, how fast. Besides theoretical results, we also present a careful empirical study and discuss the obtained data.

Chapter 5 is dedicated to the study of structural properties of equilibrium networks in the SUM-version for the solution concepts Swap Equilibrium, Asymmetric Swap Equilibrium, Greedy Equilibrium and Nash Equilibrium.

Finally, in Chapter 6 we summarize the main contributions of this thesis and give an overview of some intriguing open questions and further research directions.

2. Model and Basic Definitions

2.1. A Brief Introduction to Game Theory

The research area Game Theory studies the interaction of independent strategic agents in the broadest sense. We do not aim to introduce the whole area, since we only need some key concepts from one of the most important branches of Game Theory, which is called non-cooperative Game Theory. A detailed introduction to all aspects of classical Game Theory can be found in the standard textbooks by Myerson [Mye91] and by Osborne and Rubinstein [OR94]. The relatively young area of Algorithmic Game Theory is excellently introduced in the books by Nisan, Roughgarden, Tardos and Vazirani [NRTV07] and by Shoham and Leyton-Brown [SLB09].

In non-cooperative Game Theory we focus on analyzing the interaction of agents which are *rational* and *selfish*.

The term "rational" means that agents act in a consistent way according to their own interests. These egoistic interests are modeled by Utility Theory, which was introduced by von Neumann and Morgenstern [VNM44] in a very influential book.[1] The interests of an agent are modeled by an *utility function* which simply maps states of the world as seen from the agent's perspective to a real number with the property that states which are more preferred by the agent have a higher utility value and vice versa.[2] We will get more formal below.

By "selfish" we mean that agents evaluate any state of the world from their own egoistic perspective and act primarily to achieve their own goals, that is, to maximize their own utility value. Agents are ignorant of the preferences, that is, the utility functions, of other agents and there is no explicit coordination among agents. However, coordination may arise whenever this suits all participating agents individually.

We are left to specify what it means when agents "act" to achieve their goals. In *finite strategic games* it is assumed that every agent individually has a finite number of different actions to choose from and that there is only a finite number of agents. In the following this number will be n. Let the set of actions for agent i be $A_i = \{a_1, \ldots, a_k\}$. Thus, the set $\mathcal{A}$ of all possible combinations of actions of agents

[1] This book [VNM44] is often addressed as the birth of Game Theory.

[2] Von Neumann and Morgenstern have shown that based on some simple axioms about agents' preferences such a function always exists. Thus, astonishingly, this simple idea of mapping an arbitrary complex state to a single number is not an over-simplification.

is the Cartesian product of these sets, that is,

$$\mathcal{A} = A_1 \times A_2 \times \cdots \times A_n \ .$$

We will call any element $a \in \mathcal{A}$ an *action-profile*. But agents are in general not restricted to deterministically choosing one available action. They are allowed to randomize between different actions. This is captured with the notion of a mixed strategy. As we will see, agents act in the strategic game by choosing such a mixed strategy. A *mixed strategy* s_i of agent i is any probability distribution over her set of available actions A_i. We let $s_i(a_j)$ denote the probability that agent i chooses action $a_j \in A_i$ under the probability distribution s_i. Let $\mathcal{S}_i$ denote the set of all probability distributions over A_i, that is, $\mathcal{S}_i$ is the set of all mixed strategies of agent i. If a mixed strategy $s_i \in \mathcal{S}_i$ chooses some action a_j with probability 1, that is, $s_i(a_j) = 1$, then this strategy is called a *pure strategy*. Thus, we have that choosing the pure strategy s_i with $s_i(a_j) = 1$ is equal to choosing the action a_j and we will sometimes refer to actions as pure strategies. The *set of all strategy-profiles* $\mathcal{S}$, also called the *strategy-space*, is the Cartesian product of the sets of mixed strategies of all agents, that is,

$$\mathcal{S} = \mathcal{S}_1 \times \mathcal{S}_2 \times \cdots \times \mathcal{S}_n \ .$$

A *strategy-profile* $s \in \mathcal{S}$ is a n-dimensional vector of strategies, where the i-th component of s, that is, s_i, specifies the strategy chosen by agent i. Let s_{-i} denote the $n-1$-dimensional vector which is obtained from s by removing the i-th entry. Thus s_{-i} specifies the chosen strategies of all agents other than agent i. We will use the convention that $s = (s_i, s_{-i})$, for all $1 \leq i \leq n$. We will sometimes say that in strategy-profile (s_i, s_{-i}) the strategy s_i is agent i's *response* to the strategies s_{-i}. Moreover, with this notation it is easy to express the strategy-profile which results from strategy-profile s if exactly one agent changes her strategy. Assume that agent i changes her strategy from s_i to s_i^* and all other agents stick to their respective strategy in s. The strategy-profile obtained by agent i's strategy-change then is $s' = (s_i^*, s_{-i})$.

Now we can rigorously define the utility function u_i of agent i. As described above this function should map states of the world to real numbers. Thus, we define u_i to be a mapping from the set of all action-profiles[3] to a real numbers, that is,

$$u_i : \mathcal{A} \to \mathbb{R} \ .$$

The crucial point here is, that the value of u_i may depend on the chosen actions of *all* agents. In other words, agent i's utility value $u_i(a)$, that is, her happiness with the state of the world $a \in \mathcal{A}$, may depend on the behavior of the other agents.

We are not yet done. Since agents may choose randomized strategies we have to define the utility of states of the world where possibly all agents choose mixed

[3]Note that an action-profile $a = (a_1, \ldots, a_n)$ is equal to the corresponding pure strategy-profile $s^a = (s_1^a, \ldots, s_n^a)$ with $s_i^a(a_i) = 1$, for all $1 \leq i \leq n$. Thus, the definition of u_i captures the utility of strategy-profiles where all agents choose pure strategies.

strategies. This is straightforward by considering the corresponding expected value. Let $s \in \mathcal{S}$ be any n-dimensional strategy-profile. Agent i's expected utility of s is defined as

$$u_i(s) = \sum_{a \in \mathcal{A}} u_i(a) \prod_{j=1}^{n} s_j(a_j) \ .^4$$

Thus, a finite strategic game can be specified by the set of agents, the set of available actions per agent (which specifies the possible strategies of that agent) and by giving a utility function for each agent.

Such games are usually analyzed by characterizing interesting sets of strategy-profiles defined by the used solution concept. A *solution concept* simply is a subset of the strategy-space $\mathcal{S}$, where all elements in the subset have some property.

Let $s = (s_i, s_{-i})$ be any strategy-profile. If there is a strategy s_i' for agent i such that $u_i(s_i', s_{-i}) > u_i(s_i, s_{-i})$, then we say that s_i' is an *improving response* for agent i. If we have that

$$\forall s_i^* \in \mathcal{S}_i : u_i(s_i', s_{-i}) \geq u_i(s_i^*, s_{-i}) \ ,$$

then we say that strategy s_i' is a *best response* of agent i.

Best response strategies directly lead us to the classical and most famous solution concept: the *Nash Equilibrium* [Nas50]. For the following definition, let N be the set of agents and let $|N| = n$.

Definition 2.1.1 (Nash Equilibrium) *A strategy-profile $s \in \mathcal{S}$ is in* Nash Equilibrium*, if*

$$\forall i \in N \ \ \forall s_i^* \in \mathcal{S}_i : \ u_i(s) = u_i(s_i, s_{-i}) \geq u_i(s_i^*, s_{-i}) \ .$$

This solution concept can be understood as follows: If the strategy-profile s is in Nash Equilibrium, then no agent can strictly increase her utility by unilaterally changing her strategy. That is, if all other agents stick to their strategy, then no agent has an improving response strategy in her current situation. Thus, in any Nash Equilibrium strategy-profile all agents have chosen a best response against each other.

One of the most celebrated results in Game Theory, Nash's Theorem [Nas50], guarantees that any finite strategic game must have a Nash Equilibrium strategy-profile. Thus, in any such game there is a stable state of the world in which no agent unilaterally wants to change her strategy.

However, in many strategic games, for example in the Network Creation Games which we define below, it is not suitable to assume that agents choose randomized strategies. Thus, for such games we have to restrict the agents to pure strategies. The corresponding set of stable strategy-profiles is called the pure Nash Equilibrium.

[4] This is exactly the utility function whose universal existance was proved by von Neumann and Morgenstern [VNM44].

Definition 2.1.2 (pure Nash Equilibrium) *A strategy-profile $s \in \mathcal{S}$ is in* pure Nash Equilibrium*, if for all agents $i \in N$ the strategy s_i is a pure strategy and*

$$\forall i \in N \quad \forall s_i^* \in \mathcal{S}_i^p : \; u_i(s) = u_i(s_i, s_{-i}) \geq u_i(s_i^*, s_{-i}) \;\; ,$$

where $\mathcal{S}_i^p$ is the set of pure strategies available for agent i.

Pure Nash Equilibria are a proper subset of Nash Equilibria. This statement may not be obvious, because of the heavy restriction that in pure Nash Equilibria agents compare their current pure strategy only with their alternative *pure* strategies. Some other *mixed* strategy could potentially outperform all pure strategies of an agent. The reason, why this cannot happen is the usage of the expected value in the definition of the utility of a mixed strategy-profile. Let s_i' be any mixed strategy of agent i and let $Sup(s_i') = \{a_j \in A_i \mid s_i'(a_j) > 0\}$ be the *support* of s_i'. Remember, that pure strategies are equal to the action on which they put all their probability weight. It holds that s_i' is a best response to s_{-i} if and only if all pure strategies in the support of s_i' are best responses to s_{-i}. This can be seen as follows: Assume that some pure strategy a_j in the support of s_i' is not itself a best response to s_{-i}, then agent i could improve on strategy s_i' by reducing the probability weight on action a_j and by distributing this weight to all other actions in the support. The other direction is also easy to see: If all actions in the support of s_i' are best responses, then any probability distribution over $Sup(s_i')$ must be a best response as well. Thus, if agent i has a mixed strategy as best response to s_{-i}, then this agent also has a pure strategy which is a best response to s_{-i}.

A crucial difference to mixed Nash Equilibria is that pure Nash Equilibria are not guaranteed to exist for any finite strategic game. There are simple strategic games, like Matching Pennies [SLB09], which do not have a pure Nash Equilibrium.

Fortunately for our treatment of selfish network creation below, pure Nash Equilibria in Network Creation Games will always exist.

2.2. Modeling Selfish Network Creation

In this section we formally introduce our model of selfish network creation. We start by defining the original model, the Network Creation Game, proposed by Fabrikant, Luthra, Maneva, Papadimitriou and Shenker [FLM+03] a decade ago. Next, we will introduce several solution concepts for this game and explain how variants of the game can be derived from these concepts.

We will assume basic graph-theoretic knowledge and refer the reader to Diestel's standard textbook [Die10] for basic definitions. Throughout this thesis we will use uw or wu for the undirected edge $\{u, w\} \in E$ of a graph $G = (V, E)$. We use the standard notation that $V(G)$ denotes the vertex set and $E(G)$ denotes the edge set of the graph G and we will abbreviate the number of vertices in G, that is $|V(G)|$, with the short-hand $|G|$. Moreover, let $G - u$ be the graph G after vertex $u \in V(G)$ is removed.

2.2.1. The Network Creation Game

We consider the following model of selfish network creation, called the *Network Creation Game* (NCG). We will sometimes call this came *Buy Game* (BG) to emphasize the difference to a variant called Swap Games, where only edge-swaps are allowed.

In Network Creation Games (or Buy Games) there are n selfish agents who seek to build a *connected* network among themselves which suits their individual needs. Agents are associated to vertices in the network and we will use the terms agent and vertex interchangeably. However, we will stick to the term agent, whenever a node in the network is in some way active. Note, that we also use the terms graph and network interchangeably.

Agents can influence the structure of the network by individually choosing a pure strategy. Let V be the set of agents in the network. A pure strategy of an agent u is any subset $S_u \subseteq V \setminus \{u\}$.[5] The pure strategy S_u of an agent u specifies which edges are owned (and have to be paid for) by agent u: She owns the undirected edges ux for all $x \in S_u$. Thus, agents may choose to create links to any subset of other agents in the network. Note, that the above definition implies that only incident edges can be owned. Furthermore, edges have exactly one owner. That is, if $w \in S_u$ and $u \in S_w$ for two agents u and w, then this means that there exist two edges between the vertices u and w in the network.[6]

Any combination of pure strategies of all agents, that is, the induced pure strategy-profile (or action-profile), will then uniquely determine the structure of an undirected graph, that is, which edges of the graph are present and who is the owner of each link. Interestingly, this connection between pure strategy-profiles and networks can be easily reversed: Given any network on n vertices and information on the edge-ownership of all edges then this uniquely determines the pure strategies of all agents in the network. It follows, that we have a bijection between graphs with edge-ownership information and pure strategy-profiles.

In our illustrations we will encode the edge-ownership information by directing edges away from their respective owner. We emphasize that this direction of edges does *not* influence the communication-direction of edges. For communication, all edges in any network will be undirected, that is, they can be traversed in both directions. Figure 2.1 illustrates the bijection between directed networks an the pure strategies of all agents with a toy example.

To get a complete game-theoretic model for selfish network creation, we have to define a utility function for all agents. We will work only with negative utilities and we will call them *cost*. Agents want to maximize their utility value. In terms of

[5] Observe, that this definition explicitly rules out self-loops and multi-edges owned by the same agent in the networks. This is no restriction, since with consideration of the cost function of an agent self-loops or multi-edges can never appear in any equilibrium network. Moreover we use S_u instead of s_u to emphasize that the strategy of an agent is a set.

[6] The used cost function will imply that no multi-edges can appear in any equilibrium network.

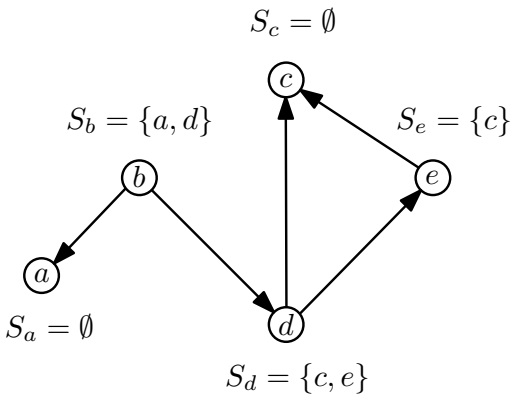


Figure 2.1.: A network with edge-ownership information and the corresponding pure strategies of all agents.

cost this means that agents want to minimize their cost. Thus, instead of a utility function we define a *cost function* for the agents. This is the crucial element of the game, since our selfish agents will choose pure strategies with the goal of minimizing their own cost.

We assume that any edge costs $\alpha > 0$, where α is a fixed parameter of the game. Since this parameter heavily influences the structure of the created networks, we will emphasize this by denoting a network G with parameter α as (G, α).

The cost of agent u in the network (G, α) is defined as follows:

$$c_u(G, \alpha) = e_u(G, \alpha) + \delta_u(G) \ .$$

Here $e_u(G, \alpha)$ is agent u's *edge-cost* within network (G, α) and $\delta_u(G)$ is agent u's *distance-cost* within network (G, α). The edge-cost $e_u(G, \alpha)$ of agent u depends only on the number of edges which are purchased by agent u, that is

$$e_u(G, \alpha) = \alpha |S_u| \ ,$$

where S_u is agent u's strategy in the network (G, α). Thus, if agent u buys k edges in network (G, α), then she has to pay edge-cost of αk.

There are two versions of the distance-cost of an agent:

- In the SUM-version, introduced in [FLM+03], agents try to minimize the sum of their shortest-path distances towards all other agents in the network, that is

$$\delta_u(G) = \begin{cases} \sum_{w \in V(G)} d_G(u, w), & \text{if } (G, \alpha) \text{ is connected} \\ \infty, & \text{otherwise} \ . \end{cases}$$

- In the MAX-version, introduced by Demaine, Hajiaghayi, Mahini and Zadimoghaddam [DHMZ12], agents try to minimize their maximum shortest-path-distance towards any other agent in the network, that is

$$\delta_u(G) = \begin{cases} \max_{w \in V(G)} d_G(u, w), & \text{if } (G, \alpha) \text{ is connected} \\ \infty, & \text{otherwise} \ . \end{cases}$$

In both cases $d_G(u, w)$ denotes the number of edges in any shortest path from vertex u to vertex w in the network (G, α). Observe that agent u's distance-cost is independent of the parameter α and depends only on the structure of the created network.

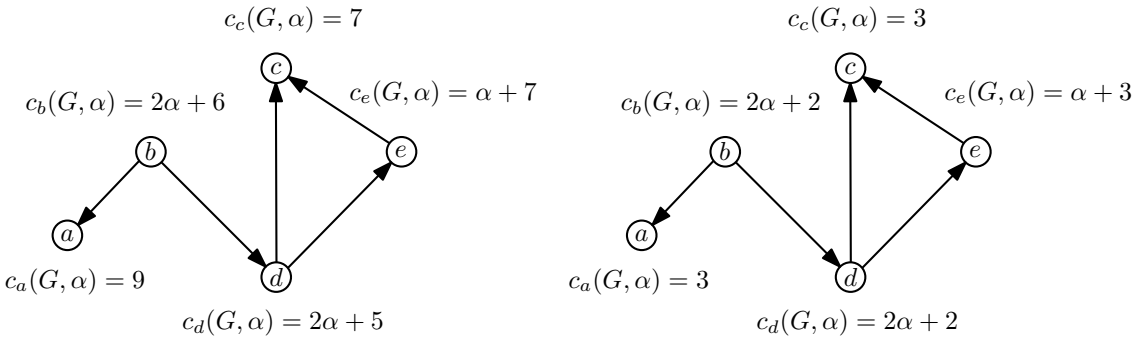
Thus, if agent u plays strategy S_u in the network (G, α) we have that the cost of agent u in (G, α) in the SUM-version is

$$c_u(G, \alpha) = \begin{cases} \alpha|S_u| + \sum_{w \in V(G)} d_G(u, w), & \text{if } G \text{ is connected,} \\ \infty, & \text{otherwise .} \end{cases}$$

In the MAX-version we have

$$c_u(G, \alpha) = \begin{cases} \alpha|S_u| + \max_{w \in V(G)} d_G(u, w), & \text{if } G \text{ is connected,} \\ \infty, & \text{otherwise .} \end{cases}$$

See Figure 2.2 for an example of the agents' cost in both the SUM- and the MAX-version. Note that for the distance-cost the edge-direction has no influence. For communication all edges are undirected.



SUM-version MAX-version

Figure 2.2.: Left: the agents' cost in the SUM-version, right: the agents' cost in the MAX-version.

Note that both cost functions nicely incorporate two conflicting objectives: Agents want to pay as little as possible for being connected to the network while at the same time they want to have good connection quality. Observe that it is easy to minimize either edge-cost or distance-cost alone by either buying no edges or by buying direct links to all other agents. But the cost is defined as the sum of both terms. This conflict between the two terms is the key ingredient which renders this model highly interesting and non-trivial. From an agent's point of view this cost function is realistic: agents like to free-ride but they have an incentive to invest in infrastructure to improve their experienced service quality in the network.

Depending on which version of the distance-cost function is used, we will call the corresponding game SUM-NCG (or SUM-BG) or MAX-NCG (or MAX-BG).

The standard solution concept for the Network Creation Game is the *pure Nash Equilibrium* (NE). Recall that a pure strategy-profile is in pure Nash Equilibrium if no agent can unilaterally change her pure strategy to another pure strategy to strictly decrease her cost. That is, if all other agents stick to their pure strategies, then no agent has an incentive to deviate from her current pure strategy. Since we have a bijection between networks with ownership-information and pure strategy-profiles, we will often say that a network is in pure Nash Equilibrium and we will sometimes call such networks *stable*, since this notion highlights the fact that in pure Nash Equilibrium no agent selfishly wants to change the structure of the corresponding network.

We will slightly abuse notation by denoting the set of all networks in pure Nash Equilibrium of the Sum-NCG as Sum-NE. For the Max-NCG the corresponding set is denoted as Max-NE. We will sometimes omit the reference to the version of the distance-cost function. We do this, whenever a statement holds for both versions or when the version is clear from the context. Moreover, whenever we use the term Nash Equilibrium in the following, it should be clear from the context that we use this as abbreviation for *pure* Nash Equilibrium. Throughout this thesis we do not consider mixed strategies of agents since for the creation of networks it makes no sense to create an edge with some probability.

As mentioned above, pure Nash Equilibria always exist for the Sum- and the Max-NCG. It was shown in [FLM+03], that for $\alpha \leq 1$ the complete network is always in Sum-NE and for $\alpha > 1$ a star is in Sum-NE. For the Max-version it was shown in [MS13], that for $\alpha \leq \frac{1}{n-2}$ a clique on n vertices is in Max-NE and it is easy to see that for $\alpha > \frac{1}{n-2}$ a star on n vertices is in Max-NE.

2.2.2. Other Solution Concepts for NCGs

Besides the pure Nash Equilibrium several other weaker solution concepts have been considered in recent research: The Greedy Equilibrium, which will be introduced in more detail in Chapter 3, the Asymmetric Swap Equilibrium, introduced by Mihalák and Schlegel [MS12], and the Swap Equilibrium, introduced by Alon, Demaine, Hajiaghayi and Leighton [ADHL13].

Definition 2.2.1 (Greedy Equilibrium) *A network (G, α) is in* Greedy Equilibrium *(GE) if no agent can unilaterally strictly decrease her cost by either buying or deleting or swapping* one *own edge. We will denote the set of all networks in Greedy Equilibrium as* Sum-*GE or* Max-*GE, depending on the version of distance-cost.*

Here an edge-swap of an agent u is the operation of replacing one incident edge which is owned by agent u with another non-existing incident edge of which then agent u will be the owner. See Figure 2.3 for an example.

In comparison to the pure Nash Equilibrium the Greedy Equilibrium clearly is a much weaker solution concept for NCGs. We have that NE $\subseteq$ GE, since if no

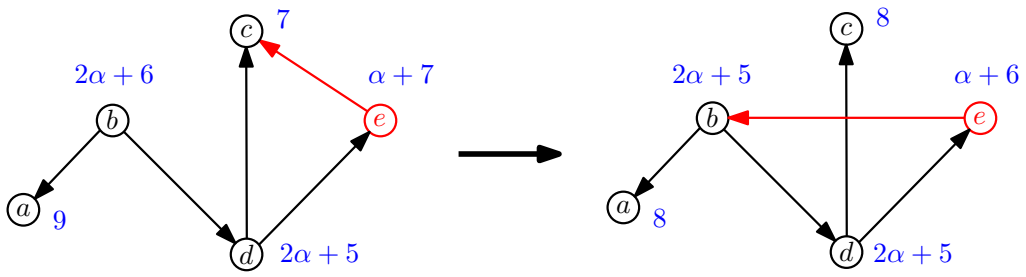


Figure 2.3.: Example of an edge-swap: Agent e swaps her own edge ec to edge eb. The agents' cost in the SUM-version is drawn in blue. Note, that agent e's swap changes the cost of other agents as well. Since the swap strictly decreases agent e's cost, it follows that the left network is not in SUM-GE, which implies that it is also not in SUM-NE.

agent can deviate to any other strategy to improve, then certainly no agent can buy, delete or swap one own edge to improve. Any such move would be a deviation to a valid new strategy.

An even weaker solution concept is the Asymmetric Swap Equilibrium:

Definition 2.2.2 (Asymmetric Swap Equilibrium) *A network (G, α) is in* Asymmetric Swap Equilibrium *(ASE) if no agent can unilaterally strictly decrease her cost by swapping* one *own edge. We will denote the set of all networks in Asymmetric Swap Equilibrium as* SUM*-ASE or* MAX*-ASE, depending on the distance-cost function.*

For example, the right network in Figure 2.3 is in SUM-ASE.

Note, that the edge-cost parameter α has no influence if we consider the ASE as solution concept. This is true since an edge-swap does not change the edge-cost of the swapping or any other agent. Thus, the edge-cost can be simply ignored or be set to 0. Hence, we will mostly omit the parameter α when we analyze networks in Asymmetric Swap Equilibrium.

Clearly, we have that any network which is in GE must be in ASE as well. The converse is not true: The right network in Figure 2.3 is in SUM-ASE but not in SUM-GE for $\alpha = 1.5$. For example, agent a can strictly decrease her cost by buying the edge ad.

The last solution concept, the Swap Equilibrium, is closely related to the ASE, but in Swap Equilibrium edge-ownership is ignored as well.

Definition 2.2.3 (Swap Equilibrium) *A network (G, α) is in* Swap Equilibrium *(SE) if, when all edges have cost* 0 *and edge-ownership is ignored, no agent can unilaterally strictly decrease her cost by swapping* one incident *edge. We will denote the set of all networks in Swap Equilibrium as* SUM*-SE or* MAX*-SE, depending on the distance-cost function.*

Since edge-ownership is ignored, we will often omit the edge-ownership information in illustrations of networks in Swap Equilibrium, that is, we will draw undirected networks.

We emphasize that with focus on Swap Equilibria agents may swap any *incident* edge independently of the edge-ownership. For example, the right network in Figure 2.3 is not in SUM-SE, since agent a can strictly decrease her cost by swapping the edge ab with the edge ad.

Networks in SE are robust against swaps from *both* endpoints of any edge. It follows that the Swap Equilibrium is a stronger solution concept than the Asymmetric Swap Equilibrium and we have that SE $\subseteq$ ASE.[7]

Note that despite their heavy restrictions (Asymmetric) Swap Equilibria are an interesting object of study since they model networks in which the agents locally weigh incident edges against alternative links. Quite surprisingly, despite their innocent statement, such networks tend to have a complicated structure. We survey those structural results in Chapter 5.

So far we have argued, that NE $\subseteq$ GE $\subset$ ASE and SE $\subset$ ASE holds. We will discuss the relationship of these sets in more detail in Chapter 3.

All these equilibria are guaranteed to exist for both the SUM- and the MAX-NCG. For GE and ASE this follows from the existence of NE and from NE $\subseteq$ GE $\subset$ ASE. For Swap Equilibria we will see in Chapter 5 that networks having diameter at most 2 are in SE.

2.2.3. Variants of NCGs

The different solution concepts for the Network Creation Game can be used to define variants of the NCG which have the property that the pure Nash Equilibrium of such a NCG-variant coincides with one of the alternative solution concepts. The trick is to restrict the possible strategy-changes of an agent accordingly. We will define the Greedy Buy Game, the Asymmetric Swap Game and the Swap Game[8]. We use these variants mostly in Chapter 4, where we study their dynamic behavior.

The Greedy Buy Game

The *Greedy Buy Game* (GBG) is very close to the Network Creation Game (or Buy Game). There is only one difference: Not all possible strategies of an agent are admissible. The admissible strategies of an agent u in a network (G, α) will depend on the current strategy of agent u and are restricted to be only slight adaptations of her current strategy.

Let S_u be agent u's strategy in network (G, α). At any time agents are allowed to only buy or delete or swap one own edge. A strategy-change of agent u from

[7] Observe, that the right network in Figure 2.3 shows that SE $\subset$ ASE holds.

[8] The Swap Game will be an exception since we cannot consider the pure NE in these games to obtain the Swap Equilibrium of NCGs.

strategy S_u towards the new strategy S_u^* is admissible for agent u in network (G, α) if

(1) $|S_u^*| = |S_u| + 1$ and $S_u \subset S_u^*$, or

(2) $|S_u^*| = |S_u| - 1$ and $S_u^* \subset S_u$, or

(3) $|S_u| = |S_u^*|$ and $|S_u \cap S_u^*| = |S_u| - 1$.

Here we have that (1) is the creation of one new own edge, (2) is the deletion of one own edge and (3) is the swap of one own edge. All operations can be seen as greedy adaptations of the current strategy, which explains the name of the game.[9]

By design we have that the pure Nash Equilibrium of the Greedy Buy Game coincides with the Greedy Equilibrium in the Network Creation Game.

We will emphasize the used version of the distance-cost function by using the terms SUM-GBG and MAX-GBG.

The Asymmetric Swap Game

If we restrict the agents' admissible strategies even more to only single edge-swaps of own edges, then we end up with the *Asymmetric Swap Game* (ASG).

To put this more formally: Let S_u be the strategy of agent u in the network (G, α). The new strategy S_u^* is admissible for agent u in (G, α) if $|S_u| = |S_u^*|$ and $|S_u \cap S_u^*| = |S_u| - 1$ holds.

The consequence of the heavy restriction on the admissible strategies is that the number of owned edges will remain constant for every agent.[10] Hence, the edge-cost per agent is constant and we will therefore omit it in the agents' cost function. It follows, that the whole cost function of an agent and the resulting (stable) networks will no longer depend on the edge-cost parameter α. This was one of the main reasons why Alon, Demaine, Hajiaghayi and Leighton [ADHL13] have proposed the Swap Game, a close relative of the ASG which was introduced earlier and which we will define below. Mostly, when discussing networks in the ASG, we will omit the parameter α.

Thus, we have that in Asymmetric Swap Games the cost of an agent u in network G in the SUM-version is defined as

$$c_u(G) = \begin{cases} \sum_{w \in V(G)} d_G(u, w), & \text{if } G \text{ is connected,} \\ \infty, & \text{otherwise .} \end{cases}$$

[9] These strategy-changes resemble a simple local search step in the (pure) strategy-space of the Network Creation Game. But we refrain from calling the game "local buy game" since this may convey that only edges to vertices in some close neighborhood can be created - which is clearly not the case.

[10] Another consequence is, that if we consider the process of network creation in the ASG, then this process should start with n-vertex networks having at least $n - 1$ edges. In Chapter 4 we will consider connected initial networks to ensure this and to avoid infinite agents' cost.

In the MAX-version of the ASG we have

$$c_u(G) = \begin{cases} \max_{w \in V(G)} d_G(u,w), & \text{if } G \text{ is connected,} \\ \infty, & \text{otherwise .} \end{cases}$$

Again, by design, we have that the networks in pure Nash Equilibrium of the ASG coincide with the networks which are in Asymmetric Swap Equilibrium for the NCG.

We will highlight the used version of the distance-cost function, by using the terms SUM-ASG and MAX-ASG.

The Swap Game

The *Swap Game* (SG), originally introduced as "Basic Network Creation Game" by Alon, Demaine, Hajiaghayi and Leighton [ADHL13], is similar to the Asymmetric Swap Game defined above, but there is a crucial difference: Edges do not have owners in the Swap Game[11]. This means that both incident agents of an edge may swap this edge. This innocent-looking detail has severe consequences for the model: First of all, the meaning of a pure Strategy S_u of agent u changes. It no longer specifies which edges are owned by agent u, instead it specifies all neighbors of u in the network G. This implies that the pure strategy of agent u influences the current pure strategies of *all* other agents. For agents $x \in S_u$ it follows that $u \in S_x$ and for agents $y \notin S_u$ if follows that $u \notin S_y$. This can be seen as a conceptual drawback of this model.[12] However, this has no severe consequences because the agents' cost function is exactly the same as in the ASG, that is, the edge-cost term is omitted.

The swap-stable networks in the Swap Game are exactly the Swap Equilibria (SE) for NCGs. Note that we cannot use the term "stable networks" or pure Nash Equilibrium here. The pure Nash Equilibrium is not a suitable solution concept for Swap Games, since with the precise definition of a pure NE we have that all networks are stable in the Swap Game. The reason is, that no agent can perform any strategy-change if all other agents stick to their strategy. This is true because the strategies depend on each other, that is, for any strategy-change some agents must cooperate. Instead we have to consider a solution concept where agents only care about their own cost as in the pure NE but affected agents will always cooperate to help an agent change her strategy.

As mentioned above, Swap Games were introduced to remove the intricate dependence on the parameter α and to get a weaker solution concept for the NCG. But, as first observed by Mihalák and Schlegel [MS13], for the NCG it is not true that any network in pure Nash Equilibrium is in Swap Equilibrium. This carries over to the Greedy Equilibrium as well. In Chapter 3 we give examples for networks

[11] Or, equivalently, edge-ownership will be ignored in the Swap Game.

[12] Curiously, the authors of [ADHL13] do not define the strategy of an agent. They only define the Swap Equilibrium from a graph theoretic point of view and discuss the differences to the pure NE in NCGs.

which are in NE but not in SE and networks which are in GE but not in SE for the NCG.

2.3. Measuring (In-)Efficiency

We want to measure the quality of networks created by selfish agents. For this, we first have to define an objective function, which quantitatively measures how good a created network is for the whole society of all participating agents. This objective function is called the *social cost*. For defining this function we adopt what is usually called the utilitarian approach: The social cost $c(G, \alpha)$ of a network (G, α) is the sum of the cost of all agents in (G, α). That is

$$c(G, \alpha) = \sum_{u \in V(G)} c_u(G, \alpha) \ .$$

With this objective function we can now ask how good a n-vertex network (G, α) is compared to some other n-vertex network (G', α). For this, we can simply consider the ratio of $c(G, \alpha)$ over $c(G', \alpha)$. For example, if this ratio is greater than 1, then network (G, α) is socially worse than network (G', α).

Our object of study are networks which are created by selfish agents without explicit coordination. A very natural and influential idea is to ask how the selfish behavior of the agents and the lacking of any centralized coordination affects the overall quality of the created networks.[13] To study this question for the Network Creation Game we can simply compare the social cost of equilibrium networks having n-vertices with the social cost of a best possible n-vertex network which may not be in equilibrium. Clearly, to have a fair comparison, the edge-cost parameter α should be the same for all compared networks.

A socially best possible n-vertex network with parameter α is a network which has minimum social cost among all n-vertex networks with parameter α. There may be many best possible n-vertex networks for parameter α, but all have the same minimum social cost. Note, that a best possible network might not be in equilibrium. Let $\mathcal{G}_n^\alpha$ denote the set of all n-vertex networks with parameter α.

We can now choose any solution concept for the NCG and can compare the social cost of the corresponding n-vertex networks with some n-vertex network having minimum social cost. Let S_n^α be any set of n-vertex networks with parameter α selected by the chosen solution concept in the NCG.[14] We can now compare the worst or best network in S_n^α to the best possible n-vertex network. This leads us to the Price of Anarchy, introduced by Koutsoupias and Papadimitriou [KP99] and the Price of Stability, introduced by Anshelevich, Dasgupta, Kleinberg, Tardos, Wexler and Roughgarden [ADK+08].

[13] This idea was suggested by Koutsoupias and Papadimitriou [KP99]. This work turned out to be of the most influential papers in Algorithmic Game Theory.

[14] That is, S_n^α is any subset of the pure strategy-space of the game. Thus, $S_n^\alpha \subseteq \mathcal{G}_n^\alpha$.

Definition 2.3.1 (Price of Anarchy) *The* Price of Anarchy *of the Network Creation Game with n agents under solution concept $S_n^\alpha \subseteq \mathcal{G}_n^\alpha$ is*

$$\frac{\max_{(G,\alpha)\in S_n^\alpha} c(G,\alpha)}{\min_{(G,\alpha)\in \mathcal{G}_n^\alpha} c(G,\alpha)} \ .$$

Definition 2.3.2 (Price of Stability) *The* Price of Stability *of the Network Creation Game with n agents under solution concept $S_n^\alpha \subseteq \mathcal{G}_n^\alpha$ is*

$$\frac{\min_{(G,\alpha)\in S_n^\alpha} c(G,\alpha)}{\min_{(G,\alpha)\in \mathcal{G}_n^\alpha} c(G,\alpha)} \ .$$

The Price of Anarchy is very close in spirit to the approximation ratio from the area of Approximation Algorithms [Vaz01] and to the competitive ratio, introduced by Sleator and Tarjan [ST85], from the area of Online Algorithms [Alb03]. It takes a worst-case perspective and characterizes how far from optimum selfishly created networks can be. The Price of Stability is more optimistic. It focuses on the best possible equilibrium networks and thus characterizes how close to optimum selfish agents may get.

We will sometimes restrict the set S_n^α even more by focusing on n-vertex equilibrium networks having a certain structure, e.g. tree networks. In this case we will call this the Price of Anarchy for the restricted class, e.g. the Price of Anarchy on trees.

The above definitions can be easily carried over to the GBG, ASG or SG. Note, that for the NCG and GBG the Price of Anarchy and the Price of Stability depend heavily on the chosen version of the distance-cost function and on the edge-cost parameter α. For the ASG and SG only the version of the distance-cost function matters. We will always use the pure Nash Equilibrium as solution concept for the NCG, GBG and ASG and the Swap Equilibrium for the SG whenever we refer to the Price of Anarchy or Stability in these games. Note that since NE, GE, ASE and SE are guaranteed to exist, the Price of Anarchy or Stability for all mentioned versions of the NCG is well-defined.

2.4. A Brief Survey of other Models

We will briefly summarize related work on other models for game-theoretic network creation settings. Since this is a rich and diverse research direction, we will only focus on models which are very close to Network Creation Games. Excellent sources for a broader overview are Jackson's survey [Jac03] and his book [Jac10].

One of the earliest ancestors of the Network Creation Game is the "connections model", proposed by Jackson and Wolinsky [JW96]. In this model links represent social relationships between agents and linked agents benefit from these connections. Agents have to pay for establishing these links. Each agent chooses a subset of other

agents she wants to connect to and a link is formed if both agents are willing to connect to each other and to pay for this link. The benefit is measured with a parameter $\delta < 1$ and diminishes with increasing distance.[15] If two agents have distance k in the created network, then their mutual benefit is δ^k. The utility of an agent i in network G is defined as

$$u_i(G) = \sum_{j \neq i \in V(G)} \delta^{d_G(i,j)} - \sum_{j:ij \in E(G)} c_{ij} \ ,$$

where c_{ij} is the cost of link ij. Thus, agents strive to be close to all other agents, but, since links are costly, they have to choose carefully whom to connect to. This is essentially the same feature which also renders the Network Creation Game interesting and realistic. Links in the connections model can only be created by concerted action of two agents. Thus, unilateral solution concepts like the Nash Equilibrium are not suitable for characterizing interesting outcomes of this game.[16] Instead, Jackson and Wolinsky propose "pairwise stability" as suitable solution concept. A network G is pairwise stable, if each existing link is wanted by both incident agents and for each non-existing link there is at least one incident agent who is not willing to pay for it. This concept is particularly interesting because stability can be checked locally by inspecting all the existing and non-existing links independently. The authors of [JW96] study the structure of networks which maximize the social utility and of networks which are pairwise stable for all ranges of δ when all edges have uniform cost of c. As one of the first works on selfish network creation, they explicitly compare stable networks to socially optimal networks. Watts [Wat01] analyzed the dynamics of the connections model when links are proposed uniformly at random and the incident agents can decide to create or sever the proposed link. The Price of Anarchy and the structure of pairwise stable networks for some ranges of δ was studied by Baumann and Stiller [BS08].

A unilateral, purely non-cooperative version of the connections model was proposed and studied by Bala and Goyal [BG00]. In this version agents selfishly choose which links to buy and no action is needed from the other incident agent for link creation. The authors consider both the version with directed links as well as the version where links are undirected. The utility function of an agent is similar to the connections model. Besides considering the networks for parameter $\delta < 1$, they also focus on the special case where $\delta = 1$. In this case, if edge-cost is low enough, agents try to maximize the number of agents in the network to which they have a (directed) path. For all these versions, the authors give characterizations of stable and social utility maximizing networks and also study the dynamics of a corresponding network creation process.

The Network Creation Game can be understood as a modification of Bala and Goyal's model with undirected links. The crucial difference is that agents in the

[15] The parameter may even depend on the specific pair of agents.

[16] For example, the empty network would always be in pure Nash Equilibrium.

NCG want to be connected to *all* other agents in the network and that the length of the shortest path in the network measures the benefit of the connection. Moreover, decreasing benefit with distance is rephrased as increasing cost with distance which allows for the very elegant and simple cost function of an agent.

Corbo and Parkes [CP05] considered a bilateral version of the NCG. In this model, agents have to agree on creating edges and both endpoints have to pay half of the edge-cost α for the edge. As for the connections model, the pure Nash Equilibrium is not a suitable solution concept. Therefore, the authors consider Jackson and Wolinski's pairwise stability and a minimal coalitional refinement of the pure Nash Equilibrium which allows for concerted action by two-agent coalitions. This solution concept is called pairwise Nash Equilibrium. The authors show that these two concepts and another solution concept proposed by Myerson [Mye91] are equivalent and they study the Price of Anarchy. We will encounter the bilateral version of the NCG in Section 4.6 where we study its dynamic behavior.

Another related model was proposed by Brautbar and Kearns [BK11] and is called the clustering coefficient network formation game. This model is motivated by the empirical observation that social networks typically have densely clustered sub-networks and that agents strive to belong to such a "social clique". Friedship in social networks is often transitive, that is, the friends of friends are mutual friends as well. Thus, such networks contain many triangles. One of the standard measures for analyzing clustering in networks is the clustering coefficient. This coefficient for an agent v in a network G is defined as $CC_v(G) = \Delta_v(G)/\binom{deg_v(G)}{2}$, where $\Delta_V(G)$ is the number of triangles which contain agent v and $deg_v(G)$ is the number of incident edges of v. A high clustering coefficient for an agent yields that this agent is part of an important social clique in the network. In the corresponding game-theoretic model, agents balance edge-cost against the clustering coefficient. Similarly to the Network Creation Game, agents unilaterally choose which edges to create and agents have to pay $\alpha > 0$ for their owned edges. The utility function of agent i in network G is $u_i(G) = CC_i(G) - \alpha|S_i|$, where S_i is agent i's pure strategy in G. The authors study the structure of the corresponding stable networks and also analyze the Price of Anarchy.

3. Approximating Equilibria

As mentioned in Section 1.1, the Internet can be understood as the result of the interplay of local strategies of the agents, that is, it can be considered as an equilibrium state of a game played by selfish agents.

The classical and most popular solution concept of stratgic games is the (pure) Nash Equilibrium [Nas50], which is a stable state, where no agent unilaterally wants to change her current (pure) strategy. However, (pure) Nash Equilibria (NE) have their difficulties. Besides their purely descriptive, non-algorithmic nature, there are two problems:

(1) With NE as solution concept agents only care if there is a better strategy and will perform radical strategy-changes even if they yield only a tiny improvement.

(2) In some games it is computationally hard even to decide if a stable state is reached because computing the best possible strategy of an agent is hard. Thus, for such games NE only predict stable states found by supernatural agents.

But what solutions are actually found by more realistic players, i.e. by agents who prefer smooth strategy-changes and who can only perform polynomial-time computations? And what impact on the stability has this transition from supernatural to realistic players?

In this chapter, we take the first steps towards answering these questions for the Sum- and Max-version of the Network Creation Game which we have defined in Section 2.2. Remember that agents model ISPs who create links towards other ISPs while minimizing cost and maximizing their quality of network usage. It seems reasonable that ISPs prefer greedy refinements of their current strategy (network architecture) over a strategy-change which involves a radical re-design of their infrastructure. Furthermore, computing the best strategy in NCGs is known to be NP-hard[1]. Hence, it seems realistic to assume that agents perform smooth strategy-changes and that they do not play optimally. We take this idea to the extreme by considering very simple agents and by introducing and analyzing a natural solution concept, called *Greedy Equilibrium*, for which agents can easily compute whether a stable state is reached and which models an ISP's preference for smooth strategy-changes.

[1]See Section 3.1.2 for more details.

3.1. Preliminaries

3.1.1. Additional Definitions

We will use the concept of an *approximate Nash Equilibrium* in the Network Creation Game. Let (G, α) be any network. For all agents $u \in V(G)$ let c_u and c_u^* denote agent u's cost induced by her current strategy in the network (G, α) and by her best possible alternative pure strategy, respectively. We say that (G, α) is in β-approximate Nash Equilibrium in the NCG if for all agents $u \in V(G)$ we have $c_u \leq \beta c_u^*$, for some $\beta \geq 1$.

3.1.2. Related Work

One of the main drawbacks of the Network Creation Game by Fabrikant, Luthra, Maneva, Papadimitriou and Shenker [FLM+03] is that computing a best possible strategy of an agent is NP-hard. The authors gave a reduction from the classical Dominating Set problem [GJ79]. Thus, computationally bounded agents in the NCG cannot even recognize easily if they can improve on their current strategy, which implies that such agents cannot tell whether they have already collectively found a stable network.

To circumvent this problem and to get rid of the intricate dependence on the edge-cost parameter α, Alon, Demaine, Hajiaghayi and Leighton [ADHL13] introduced the Swap Game, as defined in Section 2.2.3. Remember that in Swap Games agents can only swap any single incident edge to decrease their cost. Here, a swap is the exchange of an incident edge with a non-existing incident edge. The cost of an agent is defined as in Network Creation Games but without the edge-cost term. The corresponding solution concept is the *Swap Equilibrium* (SE). A network is in SE, if no agent can unilaterally swap one incident edge to strictly decrease her cost. This solution concept has the nice property that agents can check in polynomial time if they can perform an improving strategy-change. This can be done by simply checking all possible edge-swaps and computing the incurred cost. Thus, agents can decide efficiently if they have collectively found a swap-stable network. However, simplifying the model as in [ADHL13] is not without its problems. Allowing only edge-swaps implies that the number of edges remains constant. Hence, this model seems too limited to explain the creation of rapidly growing networks.

The Greedy Equilibrium, which we introduce and analyze in this chapter, can be understood as an extension of the Swap Equilibrium which has similar properties but provides agents more freedom to act and which leads to more realistic equilibrium networks.

Some of our results in this chapter are for tree networks. Such topologies are common outcomes of the Network Creation Game if edges are expensive, which led the authors of [FLM+03] to conjecture that all (non-transient) stable networks of the Sum-NCG are trees if α is greater than some constant. The conjecture, known as

the Tree Conjecture, was later disproved by Albers, Eilts, Even-Dar, Mansour and Roditty [AEED+06] but it was shown to be true for high edge-cost. In particular, Mihalák and Schlegel [MS13] proved that all stable networks are trees if $\alpha > 273n$ in the SUM-NCG or if $\alpha > 129$ in the MAX-NCG. Experimental evidence suggests that this transition to tree networks already happens at much lower edge-cost and it is an interesting open problem to improve on these bounds.[2]

Demaine, Hajiaghayi, Mahini and Zadimoghaddam [DHMZ09] investigated Network Creation Games, where agents cannot buy every possible edge. This is modeled by considering a *host graph* which specifies which edges may be created. Furthermore, Ehsani, Fazli, Mehrabian, Sadeghabad, Safari, Saghafian and Shokat-Fadaee [EFM+11] recently analyzed a bounded-budget version. Both versions seem realistic, but in the following we do not restrict the set of edges which can be bought or the budget of an agent. Clearly, such restrictions reduce the qualitative gap between simple and arbitrary strategy-changes and would lead to weaker results for our analysis. Note, that this indicates that outcomes found by simple agents in the edge- or budget-restricted version may be even more stable than we show in the following sections.

To the best of our knowledge, approximate Nash Equilibria have not been studied before in the context of selfish network creation. Close to our approach here is our previous work [AL10], which analyzes for a different game how tolerant the agents have to be in order to accept a centrally designed solution. We adopt a similar point of view by asking how tolerant agents have to be to accept a solution found by greedy play.

Guylás, Kõrösi, Szabó and Biczók [GKSB12] recently published a paper having a very similar title to our original work [Len12]. They investigate networks created by agents who use the length of "greedy paths" as communication cost and show that the resulting equilibria are substantially different to the ones we consider here. Their term "greedy" refers to the distances whereas our term "greedy" refers to the behavior of the agents.

3.1.3. Our Contribution

We introduce and analyze Greedy Equilibria (GE) as a new solution concept for the Network Creation Game (or Buy Game) which leads to the Greedy Buy Game. This solution concept is based on the idea that agents (ISPs) prefer greedy refinements of their current strategy (network architecture) over a strategy-change which involves a radical re-design of their infrastructure. Furthermore, GE represent solutions found by very simple agents, which are computationally bounded. We show in Section 3.2 that such greedy refinements can be computed efficiently and clarify the relation of GE to the other known solution concepts for NCGs which we have briefly introduced in Section 2.2.2.

[2] We provide a new line of attack on this problem in Chapter 5.

Our main contribution of this chapter follows in Section 3.3 and Section 3.4, where we analyze the stability of solutions found by greedily playing agents. For the SUM-version we show the unexpected result that, despite the fact that greedy strategy-changes may be sub-optimal from an agent's point of view, SUM-GE capture SUM-NE on trees. That is, in any tree network which is in SUM-GE *no* agent can decrease her cost by performing *any* strategy-change. For general networks we prove that any network in SUM-GE is in 3-approximate SUM-NE and we provide a lower bound of $\frac{3}{2}$ for this approximation ratio. Hence, we are able to show the rather surprising result that greedy play almost suffices to create perfectly stable networks.

For the MAX-version we show that these games have a strong non-local flavor which yields diminished stability. Here even GE-trees may be susceptible to non-greedy improving strategy-changes. Interestingly, susceptible trees can be fully characterized and we show that their stability is very close to being perfect. Specifically, we show that any star in MAX-GE is in 2-approximate MAX-NE and that any MAX-GE tree having larger diameter is in $\frac{6}{5}$-approximate MAX-NE. We give a matching lower bound for both cases. For non-tree networks in MAX-GE the picture changes drastically. We show that for MAX-GE networks having a very small α the approximation ratio is related to their diameter and we provide a lower bound of 4. For $\alpha \geq 1$, we show that there are non-tree networks in MAX-GE, which are only in $\Omega(n)$-approximate MAX-NE. The latter result yields that the locality gap of the UNCAPACITATED METRIC MIN-MAX FACILITY LOCATION problem[3] is in $\Omega(n)$.

Regarding the complexity of deciding Nash-stability, we show that there are simple polynomial time algorithms for tree networks in both versions. Furthermore, greedy-stability represents an easy-to-check certificate for 3-approximate Nash-stability in the SUM version.

3.2. Greedy Agents and Greedy Equilibria

We consider agents which check three simple ways to improve their current infrastructure. The three operations are

- *greedy augmentation*, which is the creation of *one* new own link,
- *greedy deletion*, which is the removal of *one* own link,
- *greedy swap*, which is a swap of *one* own link.

Computing the best augmentation/deletion/swap for one agent can be done in $\mathcal{O}(n^2(n+m))$ steps by trying all possibilities and re-computing the incurred cost via a modified breadth-first-search computation. Observe, that these smooth strategy-changes induce some kind of organic evolution of the whole network which seems

[3]This problem is the max-analogue of the classical Uncapacitated Metric Facility Location problem [Vaz01]. The only difference is, that the sum-operator in the objetive function is replaced by a max-operator.

highly adequate in modeling the Internet or social networks. This greedy behavior naturally leads us to a new solution concept:

Definition 3.2.1 (Greedy Equilibrium) (G, α) *is in* Greedy Equilibrium *if no agent in* G *can decrease her cost by buying, deleting or swapping* one *own edge.*

Note, that GE can be understood as solutions which are obtained by a distributed local search procedure performed by selfish agents. We will discuss the dynamics of this local search in detail in Chapter 4.

The next theorem relates GE to other solution concepts in the Sum version. See Fig. 3.1 for an illustration. Relationships are similar in the Max version.

Theorem 3.2.2 *For the* Sum *version it is true that* $NE \subset GE \subset ASE$ *and that* $SE \subset ASE$. *Furthermore, we have* $NE \backslash SE \neq \emptyset$, $GE \backslash SE \neq \emptyset$, $(GE \backslash SE) \backslash NE \neq \emptyset$, $(GE \backslash NE) \cap SE \neq \emptyset$ *and* $NE \cap GE \cap SE \neq \emptyset$.

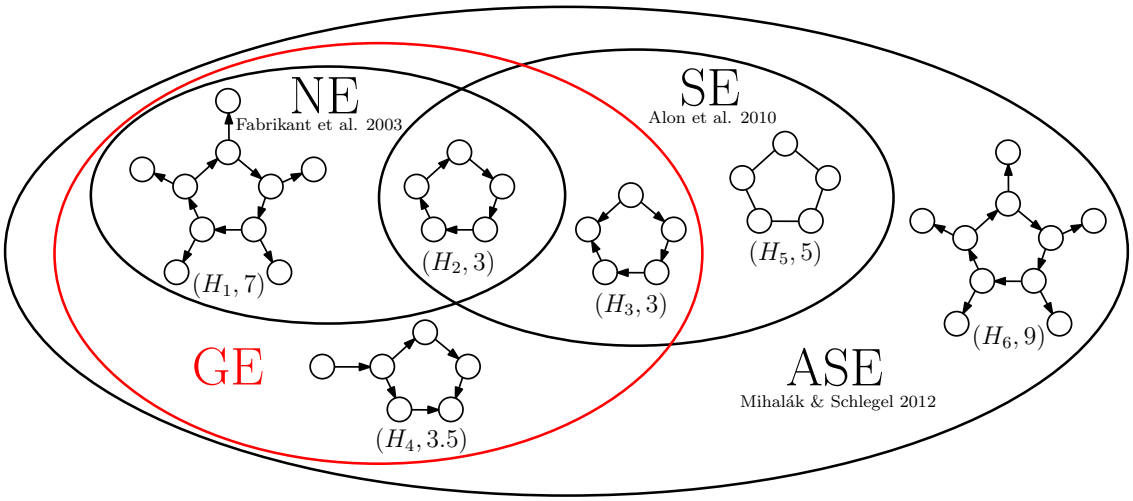


Figure 3.1.: Relations between solution concepts for NCGs in the Sum version. Edge-directions indicate edge-ownership, edges point away from their owner.

Proof It is easy to see that Sum-NE $\subseteq$ Sum-ASE and Sum-SE $\subseteq$ Sum-ASE must hold, since in both cases we restrict the set of available strategies for the agents. Clearly, greediness restricts the possible strategies of an agent as well. Hence, we have Sum-NE $\subseteq$ Sum-GE. Furthermore, by the same argument, if no agent can buy, delete or swap one own edge, then such a network must be in Asymmetric Swap Equilibrium. It follows that Sum-GE $\subseteq$ Sum-ASE.

Consider the networks depicted in Fig. 3.1. It follows from the work of Alon et al. [ADHL13] that any network having diameter 2 is in Sum-SE.[4] Thus, we

[4]See Corollary 5.2.2 in Chapter 5.

have that $(H_2, 3), (H_3, 3), (H_5, 5) \in$ SUM-SE, since the edge-cost parameter and the edge-ownerships have no influence on the stability in the Swap Game. Observe that $(H_1, 7), (H_6, 9) \notin$ SUM-SE, since any leaf-agent can swap her edge towards a neighbor of another leaf-agent and thereby strictly decrease her cost. Furthermore, $(H_4, 3.5) \notin$ SUM-SE, since an agent having distance 3 towards the leaf-agent can decrease her cost by swapping an edge towards the neighbor of the leaf.

Now we show that a cycle having 5 vertices, C_5 for short, is in SUM-GE for any edge assignment if and only if $1 \leq \alpha \leq 4$. Since C_5 is in SUM-SE, we only have to show that no agent wants to buy or delete one edge if and only if $1 \leq \alpha \leq 4$. Since every vertex of C_5 has eccentricity 2, we have that an agent can strictly decrease her cost by buying one edge if and only if $\alpha < 1$. On the other hand, since deleting one edge increases the distance-cost of the moving agent by 4, we have that an agent can strictly decrease her cost by deleting one edge if and only if $\alpha > 4$. Thus, $(H_2, 3), (H_3, 3) \in$ SUM-GE and $(H_5, 5) \notin$ SUM-GE.

Next, we show that $H_1 = H_6$ is in SUM-GE for $6 \leq \alpha \leq 8$: For H_1 to be in SUM-GE we have to make sure that no leaf-agent can buy an edge and that every cycle-agent has bought the best possible edge and cannot be better off by removing that edge or by purchasing one *additional* edge. By symmetry of the construction, we can focus on one leaf-agent l only. A best possible edge for agent l is the edge towards a cycle-vertex which has maximum distance to l. This edge decreases agent l's distance-cost by 6. Now we consider a cycle-agent u and again, by symmetry, it suffices to argue for agent u. Observe that agent u cannot remove her edge towards the neighboring leaf, since this would disconnect the network. If x removes an edge towards a neighboring cycle-vertex, then agent u's distance-cost increases by 8. Furthermore, it is easy to see that no edge-swap can decrease agent u's cost. Hence, it remains to show that agent u cannot buy an additional edge and thereby strictly decrease her cost. A best possible additional edge for u is an edge towards a non-neighboring cycle-vertex, which yields a distance decrease of 2. Hence, if $\alpha > 2$, no such additional edge will be bought by u. Analogously, it is easy to check that H_4 is in SUM-GE for $3 \leq \alpha \leq 4$. If we restrict agents only to swapping own edges, it follows that $(H_6, 9) \in$ SUM-ASE $\setminus$ SUM-GE.

Now, let us investigate $(H_1, 7)$. Note that agents of H_1 who do not own any edge cannot change their strategy to strictly decrease their cost. Hence, we only have to argue that no cycle-vertex u can unilaterally change her strategy to strictly decrease her cost. By symmetry of the construction, it suffices to argue for agent u. Let l_u be u's leaf-neighbor and let w be u's cycle-neighbor to which u owns an edge and let v be u's other cycle-neighbor. Observe that u has to buy the edge towards l_u in any strategy to ensure connectedness. Hence, we can safely ignore this edge. Furthermore, since $\alpha \leq 8$, removing edge uw does not yield a strict cost decrease for u. Since $(H_1, 7)$ is in SUM-Greedy Equilibrium, we have that u cannot swap edge uw with some other edge to decrease her cost. It remains to show that u cannot strictly decrease her cost by removing edge uw and buying at least two edges. First, let us assume that u can remove edge uw and simultaneously buy

two edges ux and uy and thereby strictly decrease her cost. Observe that $x \neq w$ and $y \neq w$ must hold, since otherwise the edge not connecting to w would be a greedy augmentation. Furthermore, it is easy to show that $v \neq x$, $v \neq y$ and $x \neq y$ must hold and that x and y cannot be leaves, since leaves are always dominated by their corresponding cycle-neighbors. Hence, the only possible strategy for agent u, which satisfies the mentioned constraints, is to buy the edges uz_1 and uz_2, where z_1 and z_2 are the cycle-vertices which have maximum distance to u. This strategy yields a distance decrease of 10 compared to buying only edge ul_u. Clearly, every edge in an equilibrium strategy, which is not required for ensuring connectedness of the network must yield at least a distance decrease of α, since otherwise the agent would be better off without buying that edge. Since $10 < 2\alpha$, we have that agent u's new cost is strictly higher than u's cost in $(H_1, 7)$. Observe that removing uw and buying three edges ux, uy, uz, with $x \neq w, y \neq w$ and $z \neq w$, yields a distance decrease of $12 < 3\alpha$. Hence, agent u cannot strictly decrease her cost by buying 3 edges. For more than three edges, where no edge is allowed to connect to w, an analogous argument yields that u cannot strictly decrease her cost. Hence, agent u cannot change her strategy to strictly decrease her cost. Analogously, it is easy to check that $(H_2, 3) \in$ Sum-NE.

The network $(H_3, 3) \notin$ Sum-NE, since the vertex which owns two edges can strictly decrease her cost by removing both edges and buying one edge towards a vertex in distance 2 in H_3. Finally, $(H_4, 3.5) \notin$ Sum-NE, since the agent who owns two edges can strictly decrease her cost by performing a similar strategy-change as the respective agent in $(H_3, 3)$. ∎

3.3. The Quality of Sum Greedy Equilibria

This section is devoted to discussing the quality of Greedy Equilibrium networks in the Sum-version of the NCG. We begin with a simple but very useful property.

Lemma 3.3.1 *If an agent u cannot decrease her cost by buying* one *edge in the* Sum*-NCG, then buying $k > 1$ edges cannot decrease agent u's cost.*

Proof Let u be an agent who cannot strictly decrease her cost in network (G, α) by purchasing one edge. Let q denote the number of edges in (G, α) owned by agent u. Now assume towards a contradiction that agent u can strictly decrease her cost by purchasing $k > 1$ edges $e_1, \ldots, e_k$. Let (G^k, α) be the network (G, α) augmented by these k edges. Hence, we have $c_u(G^k, \alpha) < c_u(G, \alpha)$. We have $c_u(G, \alpha) = q\alpha + \delta_u(G)$ and $c_u(G^k, \alpha) = q\alpha + k\alpha + \delta_u(G^k)$. Let (G^1, α) denote the network (G, α), where agent u has built the best possible additional edge e^*. That is, there is no other additional edge e', such that agent u can strictly decrease her cost by swapping edge e^* with edge e'. Since (G, α) is in Greedy Equilibrium, we have $c_u(G^1, \alpha) = q\alpha + \alpha + \delta_u(G^1) \geq c_u(G, \alpha)$. Hence, we have

$$c_u(G^k, \alpha) < c_u(G, \alpha) \iff k\alpha < \delta_u(G) - \delta_u(G^k) \tag{3.1}$$

and

$$c_u(G^1, \alpha) \geq c_u(G, \alpha) \iff \alpha \geq \delta_u(G) - \delta_u(G^1). \tag{3.2}$$

Let $g^k = \delta_u(G) - \delta_u(G^k)$ and $g^1 = \delta_u(G) - \delta_u(G^1)$, that is, g^k and g^1 denote the distance decrease of agent u by building edges $e_1, \ldots, e_k$ simultaneously or by building edge e^*, respectively. For all edges $l \in \{e_1, \ldots, e_k, e^*\}$, let g^l denote the decrease in distance-cost for agent u if only the edge l is inserted into network (G, α). Observe that $g^k \leq g^{e_1} + g^{e_2} + \cdots + g^{e_k}$ holds.

By inequality (3.1) we have that

$$\alpha < \frac{g^k}{k} \leq \frac{g^{e_1} + g^{e_2} + \cdots + g^{e_k}}{k} \ .$$

It follows that $\alpha < g^{e_j}$, for some $1 \leq j \leq k$, since otherwise we would have

$$\alpha < \frac{g^{e_1} + g^{e_2} + \cdots + g^{e_k}}{k} \leq \frac{k\alpha}{k} = \alpha \ .$$

Furthermore, since e^* is the best possible additional edge for agent u in (G, α), we have $g^{e_j} \leq g^{e^*}$. It follows that $\alpha < g^{e^*}$, which contradicts inequality (3.2). ∎

3.3.1. Tree Networks in Sum Greedy Equilibrium

We show that in the NCG all stable trees found by greedily behaving agents are even stable against *any* strategy-change. Hence, in case of a tree equilibrium *no* loss in stability occurs by greedy play. This is a counter-intuitive result, since for each agent alone being greedy is clearly sub-optimal (the network in Fig. 3.2 with $\alpha = 6$ is an example). Thus, the following theorem shows the emergence of an optimal outcome out of a combination of potentially sub-optimal strategies.

Theorem 3.3.2 *If (T, α) is in* Sum-*GE and T is a tree, then (T, α) is in* Sum-*NE.*

Before we prove Theorem 3.3.2, we first provide some useful observations. The well-known notion of a 1-*median* [KH79b] is used:

Definition 3.3.3 (1-median) *A* 1-median *of a connected graph G is a vertex $x \in V(G)$, where*

$$x \in \arg\min_{u \in V(G)} \sum_{w \in V(G)} d_G(u, w) \ .$$

Lemma 3.3.4 *Let (T, α) be a tree network in* Sum-*GE. If agent u owns edge uw in (T, α), then w must be a 1-median of its tree in the forest $T - \{u\}$.*

Proof Assume towards a contradiction that vertex w is not a 1-median vertex in its respective tree T_w in the forest $T - \{u\}$. Clearly, agent u's unique shortest paths to all vertices in $V(T_w)$ in (T, α) traverse vertex w and we have that agent u's total

distance-cost to vertices in T_w is $|V(T_w)| + \sum_{v \in V(T_w)} d_{T_w}(w, v)$. Let x be a 1-median vertex of T_w. By definition, we have that

$$\sum_{v \in V(T_w)} d_{T_w}(x, v) < \sum_{v \in V(T_w)} d_{T_w}(w, v) \ .$$

Thus, agent u can strictly decrease her total distance-cost to vertices in T_w by performing an edge-swap from w towards x. This contradicts the fact that (T, α) is in Sum-GE. ∎

Let (T, α) be any tree network in Sum-GE and let T^u be the forest induced by removing all edges owned by agent u from T. Let F^u be the forest T^u without the tree containing vertex u. The above lemma directly implies the following:

Corollary 3.3.5 *Let (T, α) be in Sum-GE, and let F^u be defined as above. Agent u's strategy in (T, α) is the optimal strategy among all strategies that buy exactly one edge into each tree of F^u.*

Let $x \in V(T)$ be a 1-median of the tree T. Let $u \notin V(T)$ be a special vertex. We consider the network (G_T^u, α), which is obtained by adding vertex u and inserting edge ux, which is owned by u, in T and by assigning the ownership of all other edges arbitrarily among the respective endpoints of any other edge in G_T^u. Furthermore, let $y_1, \ldots, y_l$ denote the neighbors of vertex x in T and let T_{y_i}, for $1 \leq i \leq l$, denote the maximal subtree of T which is rooted at y_i and which does not contain vertex x. See Fig. 3.2 (left) for an illustration. We consider a special strategy of agent u in

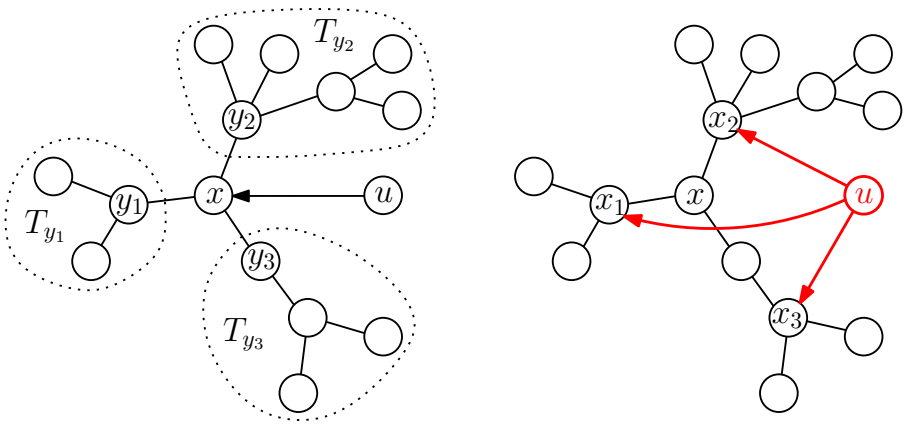


Figure 3.2.: The network (G_T^u, α) before and after agent u changes her strategy to strategy S_u^*.

(G_T^u, α): Let $S_u^* = \{x_1, \ldots, x_k\}$ be the best strategy of agent u which purchases *at least two* edges. The situation with agent u playing strategy S_u^* is depicted in Fig. 3.2 (right).

Lemma 3.3.6 *Let (G_T^u, α), $S_u^* = \{x_1, \ldots, x_k\}$ and the subtrees T_{y_i}, for $1 \leq i \leq l$ be specified as above. There is no subtree T_{y_i}, which contains all vertices $x_1, \ldots, x_k$.*

PROOF We assume towards a contradiction that there is a strategy S_u^* buying $k > 1$ edges and a subtree T_{y_i} of T such that $x_1, \ldots, x_k$ are vertices of T_{y_i}.

We claim that if this is the case, then there is a strategy S_u' which purchases exactly k edges and which strictly outperforms strategy S_u^*, that is, agent u can strictly decrease her cost by switching from strategy S_u^* to strategy S_u'. Clearly, this yields a contradiction to S_u^* being the best strategy for agent u.

Consider the vertices $x_1, \ldots, x_k$ induced by strategy S_u^*. By assumption, all these vertices are contained in subtree T_{y_i}. Observe that vertex x does not belong to any subtree T_{y_j}. We will use the following well-known fact [KH79b] about a 1-median in a tree stated in our terminology: Vertex x is a 1-median of tree T having n vertices if and only if $|V(T_{y_i})| \leq \frac{n}{2}$ for all $1 \leq i \leq l$.

Let $x' \in \{x_1, \ldots, x_k\}$ be the vertex having minimum distance to vertex x and let x'' be the neighbor of x' which is closer to x. (Note that $x'' = x$ is possible and that x'' must be a non-neighbor of u.) Clearly, we have $d_T(x', x) \geq 1$. Let S_u' be the strategy S_u^* with only one modification: Vertex x' is replaced by vertex x''.

We claim that S_u' yields strictly less cost for agent u than strategy S_u^*. Observe that since x is a 1-median we have $|V(T) \setminus V(T_{y_i})| \geq \frac{n}{2}$. Hence, the replacement of x' by x'' yields a cost decrease for agent u by at least $\frac{n}{2}$. On the other hand, this replacement increases agent u's cost by at most $\frac{n}{2} - 1$. This is true, because $k > 1$ and $d_T(x', x'') = 1$ we have that agent u's distances to all but one vertices in T_{y_i} can possibly increase by 1. Since we have only replaced x' by x'' all other distances stay the same. Hence, we have that S_u' yields strictly less cost for agent u than strategy S_u^* and we have a contradiction. ∎

Next, let us consider two special strategies of agent u. Let S_u^1 be agent u's best strategy, which buys at least two edges including one edge towards vertex x. Furthermore, let S_u^2 be agent u's best strategy, which buys at least two edges but no edge towards vertex x.

Lemma 3.3.7 *Let (G_T^u, α), S_u^1, S_u^2 and vertex x be specified as above. Let $x_j \in S_u^2$ be a vertex which has minimum distance to x among all vertices in S_u^2. If strategy S_u^2 yields less cost for agent u than strategy S_u^1, then x_j cannot be a leaf of G_T^u.*

PROOF We assume towards a contradiction that S_u^2 yields strictly less cost for agent u than strategy S_u^1 and vertex $x_j \in S_u^2$, which has minimum distance to x among all vertices in S_u^2, is a leaf of G_T^u. Let x_j' be the unique neighbor of x_j. It follows that $d(x_j', x) = d(x_j, x) - 1$. There are two cases:

If $d(x_j, x) \geq 2$, then let S_u' be the strategy S_u^2, where vertex x_j is replaced by vertex x_j'. We claim that agent u can strictly decrease her cost by switching from strategy S_u^2 to strategy S_u'. Observe that by switching from S_u^2 to S_u', agent u decreases her distance to x and to x_j' by one. On the other hand, only the distance

to vertex x_j increases by one. Observe that $|S'_u| = |S^2_u|$ and $x \notin S'_u$. Hence, S'_u yields strictly less cost for agent u than strategy S^2_u, which is a contradiction to the fact that S^2_u is agent u's best strategy which buys at least two edges and no edge towards x.

On the other hand, consider the case where $d(x_j, x) \leq 1$. Note that $x_j \neq x$, since $x \notin S^2_u$. Hence, we have $d(x_j, x) = 1$. Let S''_u be the strategy S^2_u, where we replace x_j by x. We have $|S^2_u| = |S''_u|$ and $x \in S''_u$. Furthermore, since $x \notin S^2_u$, $d(x_j, x) = 1$ and since x_j is a leaf, we have that strategy S''_u yields at most the cost of S^2_u for agent u. This contradicts the fact that S^2_u strictly outperforms S^1_u. ∎

Now we have all the tools we need to prove Theorem 3.3.2.

Proof (of Theorem 3.3.2) We will prove the contra-positive statement of Theorem 3.3.2. We show that if an agent u can decrease her cost by performing a strategy-change in a tree network (T, α) which is in Sum-GE, then there is an agent z in $V(T)$ who can decrease her cost by performing a *greedy* strategy-change. In that case we have a contradiction to (T, α) being in Sum-GE.

If agent u can decrease her cost by buying, deleting or swapping one own edge, then we have $u = z$ and we are done. Hence, we assume that agent u cannot decrease her cost by a greedy strategy-change but by performing an arbitrary strategy-change. We consider agent u's strategy-change towards the best possible *arbitrary* strategy S^* (if u has more than one such strategy, then we choose the one which buys the least number of edges).

Clearly, agent u cannot remove any owned edge without purchasing edges, since T is a tree and the removal would disconnect T. Furthermore, since (T, α) is in Sum-GE and by Lemma 3.3.1, agent u cannot decrease her cost by purchasing $k > 0$ *additional* edges. Hence, the only way agent u can possibly decrease her cost is by removing j own edges and building k edges simultaneously. Clearly, $k \geq j$ must hold. Furthermore, by Corollary 3.3.5, it follows that $k > j$.

Let F^u be the forest obtained by removing the j edges owned by agent u from T and let T^* be the tree in F^u which contains vertex u. Observe that among the k new edges there cannot be edges having an endpoint in T^*. This is true because (T, α) is in Sum-GE and by Lemma 3.3.1. Any such edge would be a possible greedy augmentation which we assume not to exist. Hence, by the pigeonhole principle, we have that there must be at least one tree T_q in F^u into which agent u buys at least *two* edges with strategy S^*. We focus on T_q and will find agent z within.

Let ux, with $x \in V(T_q)$, be the unique edge of T which connects u to the subtree T_q. Hence, agent u's strategy-change to S^* removes edge ux and buys $k_q > 1$ edges $ux_1, \ldots, ux_{k_q}$, with $x_j \in V(T_q)$ for $1 \leq j \leq k_q$. Let $X = \{x_1, \ldots, x_{k_q}\}$. By Lemma 3.3.1, we have $x_j \neq x$, for $x_j \in X$. Let $y_1, \ldots, y_l$ denote the neighbors of vertex x in T_q and let $T_{y_1}, \ldots, T_{y_l}$ be the maximal subtrees of T_q not containing vertex x, which are rooted at vertex $y_1, \ldots, y_l$, respectively. Let $x_a \in X$ be a vertex of X which has minimum distance to vertex x. Let $T_a \in \{T_{y_1}, \ldots, T_{y_l}\}$

be the subtree containing x_a. By Lemma 3.3.6, we have that there is a subtree $T_b \in \{T_{y_1}, \ldots, T_{y_l}\}$, with $T_b \neq T_a$, which contains at least one vertex of X. Let $B = \{x_{b_1}, \ldots, x_{b_p}\} = X \cap V(T_b)$. Furthermore, since no strategy which buys at least two edges including an edge towards x into T_q outperforms u's greedy strategy within T_q and by Lemma 3.3.7, we have that vertex x_a cannot be a leaf. That is, there is a vertex $z \in V(T_q)$, which is a neighbor of x_a, such that $d(z,x) > d(x_a,x)$. We show that agent z can decrease her cost by buying *one* edge in (T,α).

First of all, notice that by definition of S^*, we have that *each* edge ux_j, with $x_j \in X$, must independently of the other bought edges yield a distance decrease of *more than* α for agent u. Otherwise agent u could remove this edge and obtain a strictly better (or smaller) strategy, which contradicts the fact that S^* is the best possible strategy (buying the least number of edges). Let $D_j \subset V(T_q)$ be the set of vertices to which edge ux_j is the first edge on agent u's unique shortest path. Since x_a has minimum distance to x, it follows that $D_r \subseteq V(T_b)$ for $r \in \{b_1, \ldots, b_p\}$.

The main observation is that agent z faces in some sense the same situation as agent u with strategy S^* but without all edges uy, where $y \in B$: Both have vertex x_a as neighbor and their shortest paths to any vertex in T_b all traverse x_a and x. Remember that each edge uy, for all $y \in B$, yields a distance decrease of more than α for agent u and that $D_r \subseteq V(T_b)$, for $r \in \{b_1, \ldots, b_p\}$. Furthermore, removing all those edges from S^* yields a strict cost increase for agent u. This implies that agent z can decrease her cost by buying all edges zy, for $y \in B$, simultaneously. If $|B| = 1$, then this strategy-change is a greedy move by agent z which decreases z's cost. If $|B| > 1$, then, by the contra-positive statement of Lemma 3.3.1, it follows that there exists *one* edge zy^*, with $y^* \in B$, which agent z can greedily buy to decrease her cost. ∎

3.3.2. Non-Tree Networks in Sum Greedy Equilibrium

There exist non-tree networks in Sum-GE, since, as shown by Albers, Eilts, Even-Dar, Mansour and Roditty [AEED+06], there exist non-tree networks in Sum-NE and we have Sum-NE ⊆ Sum-GE. Having Theorem 3.3.2 at hand, one might hope that this nice property carries over to non-tree Greedy Equilibria in the Sum-version. Unfortunately, this is not true.

Theorem 3.3.8 *There is a network in* Sum-*GE which is not in* β*-approximate* Sum-*NE for* $\beta < \frac{3}{2}$.

Proof We consider a special family $G_1, G_2, \ldots$ of networks. The network G_k is constructed as follows: We have $V(G_k) = \{u, w, x, y_1, \ldots, y_k\} \cup \{z_i^j \mid 1 \leq i,j \leq k\}$. Vertex u owns edges towards $y_1, \ldots, y_k$, vertex w owns edges towards x and u, each vertex z_i^j owns an edge to x and y_i and the vertices $y_1, \ldots, y_k$ form a clique, with arbitrary edge-ownership. Fig. 3.3 (left) illustrates this construction for $k = 3$.

First we show that the network $(G_k, k+1)$ is in Sum-GE and then we show that agent u's best strategy yields a cost decrease by a factor of roughly $\frac{2}{3}$.

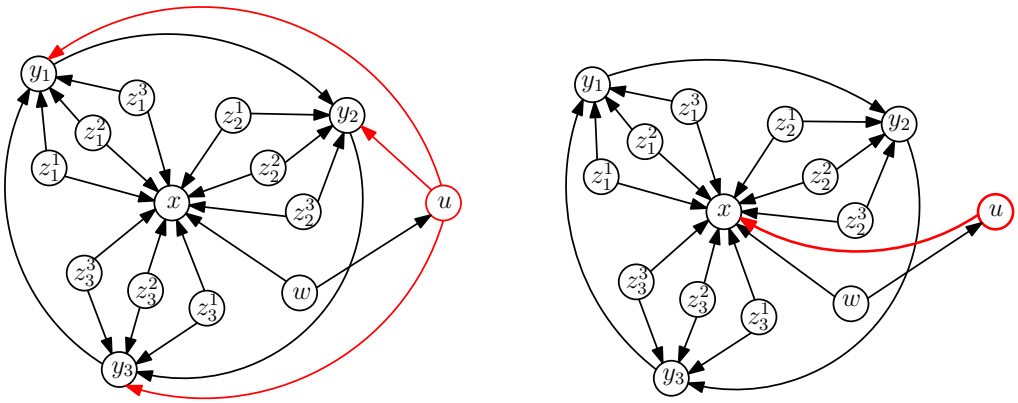


Figure 3.3.: The network $(G_k, k+1)$ for $k = 3$ and agent u's best response. For $k \to \infty$ agent u's improvement approaches a factor of $\frac{3}{2}$.

Note that G_k has diameter 2. Since $\alpha = k + 1 > 1$, it follows that no agent can buy an edge to decrease her cost. Furthermore, swapping any own edge cannot decrease an agent's cost either, since the number of neighbors stays the same. Thus, we only have to argue that no agent can delete an own edge to decrease her cost.

Consider agent u. Deleting edge uy_i increases u's distances to $y_i, z_i^1, \ldots, z_i^k$ by one. Hence, for $\alpha = k + 1$, this operation does not decrease agent u's cost. An agent y_i is in essentially the same situation. If y_i deletes her edge y_iy_j, then y_i's distances to $y_j, z_j^1, \ldots, z_j^k$ increase by one. Thus, agents $y_1, \ldots, y_k$ cannot delete an edge to decrease their cost.

If agent w deletes edge wu, then all distances towards $u, y_1, \ldots, y_k$ increase by one. Furthermore if w deletes edge wx, then all distances towards x and all z_i^j, for $1 \leq i, j \leq k$ increase by at least one. Thus, agent w cannot delete an edge to decrease her cost.

Finally, consider an agent z_i^j. Deleting edge z_i^jx increases z_i^j's distances to x and all z_p^q, for $p \neq i$ and $1 \leq q \leq k$. Deleting edge $z_i^jy_i$ increases z_i^j's distances to $u, y_1, \ldots, y_k$ by one. Hence, no agent can delete an edge to decrease her cost and we have that $(G_k, k+1)$ is in Sum-GE.

Now consider a strategy-change of agent u from strategy $S_u = \{y_1, \ldots, y_k\}$ to strategy $S_u^* = \{x\}$, see Fig. 3.3 (right). Let $(G_k^*, k+1)$ be the network induced by S_u^*. We claim that S_u^* is agent u's best possible strategy. It is easy to see that no other strategy S_u', with $|S_u'| \leq 1$ outperforms S_u^*. Furthermore, note that with strategy S_u^* agent u has exactly the k vertices $y_1, \ldots, y_k$ at distance 3. Any edge uy_i yields a cost decrease of exactly $k + 1$, but since $\alpha = k + 1$, such an edge does not decrease agent u's cost. Clearly, edges towards a vertex z_i^j are even worse than edges towards y_i. By Lemma 3.3.1, we have that even more additional edges cannot decrease u's cost. Furthermore, it is easy to see that strategy S_u is agent u's best possible strategy, which does not buy an edge towards x.

We will show that S_u^* yields strictly less cost than S_u for agent u, which will settle

the claim that S_u^* is optimal. We have

$$\lim_{k\to\infty} \frac{c_u(G_k, k+1)}{c_u(G_k^*, k+1)} = \lim_{k\to\infty} \frac{k\alpha + k + 1 + 2(k^2+1)}{\alpha + 2 + 2k^2 + 3k} = \lim_{k\to\infty} \frac{3k^2+2k+3}{2k^2+4k+3} = \frac{3}{2} \ .$$

Thus, for any $\beta < \frac{3}{2}$ there is a k' such that $c_u(G_{k'}, k'+1) > \beta c_u(G_{k'}^*, k'+1)$, which implies that the Sum-GE $(G_{k'}, k'+1)$ is not a β-approximate Sum-NE for $\beta < \frac{3}{2}$.■

Now let us turn to the good news. We show that Sum-GEs cannot be arbitrarily unstable. On the contrary, they are very close to Sum-NEs in terms of stability.

Theorem 3.3.9 *Every network in* Sum-*GE is in* 3*-approximate* Sum-*NE.*

Proof We prove Theorem 3.3.9 by providing a "locality gap preserving" reduction to the Uncapacitated Metric Facility Location problem (UMFL) [Vaz01].

Let u be an agent in (G, α) and let Z be the set of vertices in $V(G)$ which own an edge towards u. Consider the network (G', α), where all edges owned by agent u are removed. Observe that the set Z is the same in (G, α) and (G', α). Let $\mathcal{S} = \{U \mid U \subseteq (V(G') \setminus \{u\}) \wedge U \cap Z = \emptyset\}$ denote the set of agent u's pure strategies in (G', α) which do not induce multi-edges or a self-loop. We transform (G', α) into an instance $I(G')$ for UMFL as follows:
Let $V(G') \setminus \{u\} = F = C$, where F is the set of facilities and C is the set of clients. For all facilities $f \in Z \cap F$ we define the opening cost to be 0, all other facilities have opening cost α. Thus, Z is exactly the set of cost 0 facilities in $I(G')$. For every $i, j \in F \cup C$ we define $d_{ij} = d_{G'}(i, j) + 1$. If there is no path between i and j in G', then we define $d_{ij} = \infty$. Clearly, since the distance in G' is metric we have that all distances d_{ij} in $I(G')$ are metric as well. See Fig. 3.4 for an example.

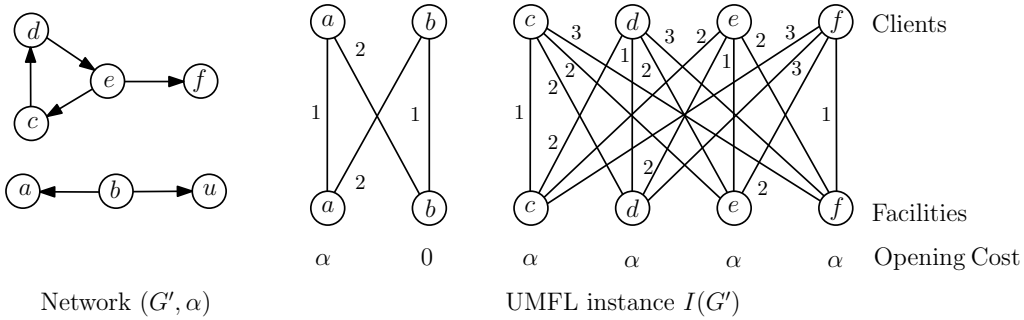


Figure 3.4.: Network (G', α) and its corresponding UMFL instance $I(G')$. Edges between clients and between facilities are omitted. All other omitted edges have length ∞.

Now, observe that any strategy $S \in \mathcal{S}$ of agent u in (G', α) corresponds to the solution of the UMFL instance $I(G')$, where exactly the facilities in $F_S = S \cup Z$ are opened and where all clients are assigned to their nearest open facility. Moreover,

every solution $F' = X \cup Z$, where $X \subseteq F \setminus Z$, for instance $I(G')$ corresponds to agent u's strategy $X \in \mathcal{S}$ in (G', α). Let $\mathcal{S}_{\text{UMFL}} = \{W \subseteq F \mid Z \subseteq W\}$ denote the set of all solutions to instance $I(G')$, which open at least all cost 0 facilities. Hence, we have a bijection $\pi : \mathcal{S} \to \mathcal{S}_{\text{UMFL}}$, with $\pi(S) = S \cup Z$ and $\pi^{-1}(X) = X \setminus Z$. Let $\pi(S) = F_S$ and let (G_S, α) denote the network (G', α), where agent u has bought all edges towards vertices in S. Let $cost(F_S)$ denote the cost of the solution F_S to instance $I(G')$. We have that agent u's cost in (G_S, α) is equal to the cost of the corresponding UMFL solution F_S, since

$$\begin{aligned} c_u(G_S, \alpha) &= \alpha|S| + \sum_{w \in V(G_S) \setminus \{u\}} \Big(1 + \min_{x \in S \cup Z} d_{G'}(x, w)\Big) \\ &= \alpha|S| + 0|Z| + \sum_{w \in V(G_S) \setminus \{u\}} \min_{x \in S \cup Z} d_{xw} \\ &= \alpha|F_S \setminus Z| + 0|Z| + \sum_{w \in C} \min_{x \in F_S} d_{xw} \quad = cost(F_S) \ . \end{aligned}$$

We claim the following: If agent u plays strategy $S \in \mathcal{S}$ and cannot decrease her cost by buying, deleting or swapping *one* edge in (G_S, α), then we have that the cost of the corresponding solution $F_S \in \mathcal{S}_{\text{UMFL}}$ to instance $I(G')$ cannot be strictly decreased by opening, closing or swapping *one* facility.

Proving the above claim suffices to prove Theorem 3.3.9. This can be seen as follows: For UMFL, Arya, Garg, Khandekar, Meyerson, Munagala and Pandit [AGK+04] have already shown that the locality gap of UMFL is 3, that is, that any UMFL solution in which clients are assigned to their nearest open facility and which cannot be improved by opening, closing or swapping *one* facility is a 3-approximation of the optimum solution.

By construction of $I(G')$, we have that every facility $z \in Z$ is the unique facility which is nearest to some client $w \in C$. Thus, we have that in any locally optimal and any globally optimal UMFL solution to $I(G')$ all cost 0 facilities must be open, since otherwise such a solution can be improved by opening a cost 0 facility. Hence, every locally or globally optimal solution to $I(G')$ has a corresponding strategy of agent u which yields the same cost. Using the claim and the result by Arya et al. [AGK+04], it follows that if agent u cannot decrease her cost by buying, deleting or swapping an edge in (G_S, α) then we have $c_u(G_S, \alpha) \leq 3c_u(G_{S^*}, \alpha)$, where S^* is agent u's optimal (non-greedy) strategy in (G', α) and (G_{S^*}, α) the network induced by S^*.

Now we prove the claim. Let $\pi(S) = F_S$. We have already shown that $c_u(G_S, \alpha) = cost(F_S)$. Furthermore, we have $Z \subseteq F_S$. We prove the contra-positive statement of the claim. Assume that solution F_S can be improved by opening, closing or swapping *one* facility. Let F'_S be this locally improved solution and let $cost(F'_S) < cost(F_S)$. Note that $Z \subseteq F'_S$ must hold. This is true, since by construction of $I(G')$ closing a cost 0 facility increases the cost of any solution to $I(G')$. Hence, no facility $z \in Z$ can be included in a closing or swapping operation. It follows that the strategy $S' := \pi^{-1}(F'_S)$ exists. Observe that $S = F_S \setminus Z$ and $S' = F'_S \setminus Z$ must differ by

one element. Furthermore, by cost-equality, we have that $c_u(G_{S'}, \alpha) = cost(F'_S) < cost(F_S) = c_u(G_S, \alpha)$. Hence, agent u can buy, delete or swap one edge in (G_S, α) to decrease her cost. ∎

3.4. The Quality of Max Greedy Equilibria

In this section, we discuss the stability of networks in MAX-GE. We will start by showing that the operations of buying, deleting and swapping edges each may have a strong non-local flavor. See Fig. 3.5 for an illustration.

Lemma 3.4.1 *For $k \geq 2$ there is a network (G, α), where an agent in the* MAX-*NCG can decrease her cost by buying/deleting/swapping k edges but not by buying/deleting/swapping $j < k$ edges.*

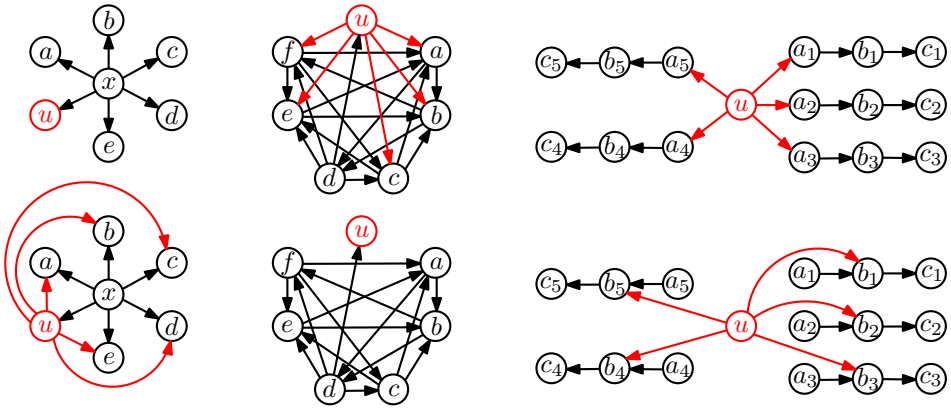


Figure 3.5.: The networks and strategy-changes for $k = 5$.

PROOF We consider each operation separately:

Buying k edges versus buying $j < k$ edges: Let G be a star having $n = k + 2$ vertices and let agent u be a leaf vertex. See Fig. 3.5 (left). Furthermore, let $\alpha < \frac{1}{n-2}$. Let x be the center of the star. If agent u owns the edge ux, then we have $c_u(G, \alpha) = \alpha + 2$, otherwise, we have $c_u(G, \alpha) = 2$. Now, observe that u has exactly k vertices at maximum distance 2. Hence, buying $j < k$ edges does not decrease agent u's maximum distance to any vertex. On the other hand, if u buys $k = n - 2$ edges to all distance 2 vertices, then agent u's distance-cost decreases by 1 while agent u's edge-cost increases by $k\alpha < 1$. Thus, buying k edges yields a strict cost decrease for agent u.

Deleting k edges versus deleting $j < k$ edges: Let G be a clique having $n = k + 2$ vertices and let u be an agent who owns all but one of her $k + 1$ incident edges. See Fig. 3.5 (middle). Let $\frac{1}{n-2} < \alpha \leq \frac{1}{n-3}$. Observe that $c_u(G, \alpha) = k\alpha + 1$. If agent u deletes $j < k$ edges, then u's distance-cost increases by 1 while u's edge-cost

decreases by $j\alpha \leq 1$. Thus, deleting $j < k$ edges does not decrease agent u's cost. On the other hand, if u deletes k edges, then u distance-cost increases by 1 while u's edge-cost decreases by $k\alpha > 1$. Hence, deleting k edges decreases agent u's cost.

Swapping k edges versus swapping $j < k$ edges: Let $G = (V, E)$ be a star-like graph which is defined as follows: Vertex u is the center of the star and we have k triples of vertices a_i, b_i, c_i, for $1 \leq i \leq k$. Let $E = \{ua_i, a_ib_i, b_ic_i \mid 1 \leq i \leq k\}$, where u owns the edges ua_i, a_i owns the edge a_ib_i and b_i owns the edge b_ic_i, for all $1 \leq i \leq k$. See Fig. 3.5 (right). Observe that agent u cannot decrease her cost by swapping $j < k$ edges simultaneously, since each edge must connect to the same subtree of u. In contrast to this, agent u can decrease her cost by performing the multi-swap, where every edge ua_i is replaced by the edge ub_i. Note that this multi-swap decreases agent u's distance-cost by 1 while having the same edge-cost.■

Having seen Lemma 3.4.1, it should not come as a surprise that greedy local optimization may get stuck at sub-optimal states of the game.

3.4.1. Tree Networks in Max Greedy Equilibrium

The examples on the left and right side of Fig. 3.5 already show that there are tree networks which are in Max-GE but not in Max-NE. In the following we show that this undesired behavior is restricted to only two families of tree networks in Max-GE. That is, we provide a characterization of all tree networks in Max-GE which are not in Max-NE. Furthermore, we show tight bounds on the stability for both mentioned families which are very close to the optimum.

We start by introducing the main actors: *Cheap Stars* and *Badly Connected Trees*.

Definition 3.4.2 (Cheap Star) *A network (T, α) in* Max-*GE is called a* Cheap Star*, if T is a star having at least $n \geq 4$ vertices and $\alpha < \frac{1}{n-2}$. Furthermore, the ownership of all edges in T is arbitrary.*

As we will see, Cheap Stars are exactly those tree networks (T, α) in Max-GE which remain in Max-GE even if we let the edge-cost parameter α tend to 0.

Definition 3.4.3 (Badly Connected Tree) *A tree network (T, α) in* Max-*GE is a* Badly Connected Tree *if there is an agent $u \in V(T)$ who can decrease her cost by swapping $k > 1$ own edges simultaneously.*

Intuitively, Cheap Stars owe their instability to a multi-buy operation, whereas Badly Connected Trees owe their instability to a multi-swap operation. Observe that Cheap Stars have diameter 2 and that Badly Connected Trees have diameter at least 3. Hence, these families are disjunct. The following theorem shows that Cheap Stars and Badly Connected Trees are the *only* tree networks in Max-GE which are not in Max-NE.

Theorem 3.4.4 *Let (T, α) be a network in* Max-*GE, where T is a tree. The network (T, α) is in* Max-*NE if and only if it is* not *a Cheap Star or a Badly Connected Tree.*

The proof of Theorem 3.4.4 is based on the following two observations.

Lemma 3.4.5 *Let (T, α) be a tree network in* Max-*GE having diameter at most* 2*. If (T, α) is not in* Max-*NE, then (T, α) is a Cheap Star.*

Lemma 3.4.6 *Let (T, α) be a tree network in* Max-*GE having diameter at least* 3*. If (T, α) is not in* Max-*NE, then (T, α) is a Badly Connected Tree.*

We start with proving Lemma 3.4.5.

Proof (of Lemma 3.4.5) Trivially, any tree network having diameter at most 1 must be in Max-NE. Hence, we focus on diameter 2 tree networks which are in Max-GE but not in Max-NE. Note that every tree network in Max-GE which has diameter 2 is stable against multi-swap operations. This is easy to see, since leaves can own at most one edge and the unique non-leaf vertex (the center of the star) has already optimal distance-cost of 1. Since edge deletions lead to a disconnected network, it follows that the instability against arbitrary strategy-changes must be due to a multi-buy operation of a leaf agent. If T has at most 3 vertices, then any leaf can buy at most one additional edge, which represents a greedy operation. Hence, T must have at least 4 vertices. Since T is a star, we have that any leaf l has exactly $n-2$ vertices in distance 2. Thus, to strictly decrease her distance-cost, agent l must buy all edges towards these $n-2$ non-neighbors. It follows that such a multi-buy operation yields a strict cost decrease for agent l, if $\alpha < \frac{1}{n-2}$. This matches exactly the definition of a Cheap Star and implies that Cheap Stars are the only possible diameter 2 tree networks in Max-GE which are not in Max-NE.■

For proving Lemma 3.4.6, we first need some additional observations. We will use well-known notion of a 1-*center* vertex [KH79a]:

Definition 3.4.7 (1-center) *A* 1-center *vertex of a connected graph G is a vertex $x \in V(G)$, where*

$$x \in \arg\min_{u \in V(G)} \max_{w \in V(G)} d_G(u, w) \ .$$

Lemma 3.4.8 *If (T, α) is a tree network in* Max-*GE having diameter at least* 3*, then $\alpha \geq 1$.*

Proof Assume towards a contradiction that there is a tree network (T, α), which is in Max-GE and has diameter at least 3 and where $\alpha < 1$. There are two cases:

If every leaf of T is a neighbor of a 1-center of T, then, since T has diameter at least 3, there must be two 1-center vertices of T and T has diameter exactly 3. It

is easy to see that any tree can have at most two 1-center vertices[5]. Thus, we have that T must be a "double-star". Let x and y be the two 1-center vertices of T and let l be a leaf which is a neighbor of x. Since y is a 1-center, there must be a leaf z which is a neighbor of y and where $d_T(l,z) = 3$. If agent l buys the edge $\{l,y\}$, then l's edge-cost increases by α but her distance-cost decreases by 1. Since $\alpha < 1$, this yields a strict cost decrease for agent l and we have a contradiction to (T,α) being in MAX-GE.

If not all leaves of T have a neighboring 1-center vertex, then let l be one such leaf which has the maximum distance to any 1-center in T. Let x be this 1-center vertex and let $d_T(l,x) = k \geq 2$. Let D_l be the set of vertices which have maximum distance to vertex l in T. Since l has maximum distance to x and x is a 1-center of T, it follows that x lies on all shortest paths from l to any vertex in D_l. Thus, if agent l buys the edge $\{l,x\}$, she reduces her distance-cost by at least 1 while increasing her edge-cost by $\alpha < 1$. This yields a strict cost decrease for agent l and again we have a contradiction to (T,α) being in MAX-GE. ∎

Lemma 3.4.9 *Let (G,α) be any network in* MAX-*GE which is not in* MAX-*NE. Let u be any agent who can strictly decrease her cost by performing a non-greedy strategy-change towards strategy S_u^*. If $\alpha \geq 1$, then agent u's distance-cost induced by strategy S_u^* is at least 2.*

PROOF Let S_u be agent u's current strategy in (G,α). Clearly, we have $\delta_u(G) > 1$, since otherwise agent u could improve on her current strategy only by deleting edges, which yields an increase in distance-cost by at least 1. We assume towards a contradiction that agent u can perform a strategy-change towards strategy S_u^*, which yields a strict cost decrease and where $\delta_u(G^*) = 1$. Here (G^*,α) is the new network induced by agent u's strategy-change. It is easy to see that $|S_u^*| > |S_u|$ must hold, since we have $\delta_u(G) > 1$ which implies that agent u cannot possibly bring her distance-cost down to 1 by performing a multi-swap. Consider agent u's shortest path in (G,α) towards a vertex having maximum distance to u. Clearly, this path has length $\delta_u(G)$. Since in (G^*,α) we have $\delta_u(G^*) = 1$, it follows that u with her new strategy S_u^* must buy an edge to all vertices on this path. Hence, agent u must buy at least $\delta_u(G) - 1$ many additional edges to achieve distance-cost of 1. But, since $\alpha \geq 1$, this yields that

$$e_u(G^*,\alpha) \geq e_u(G,\alpha) + (\delta_u(G) - 1)\alpha \geq e_u(G,\alpha) + \delta_u(G) - 1 \ .$$

Hence, we have that

$$\begin{aligned} c_u(G^*,\alpha) &= e_u(G^*,\alpha) + \delta_u(G^*) \geq e_u(G^*,\alpha) + \delta_u(G) - 1 + 1 \\ &\geq e_u(G,\alpha) + \delta_u(G) = c_u(G,\alpha) \ , \end{aligned}$$

[5]The proof is similar to the proof of Lemma 4.2.14. If there are at least three 1-center vertices, then there must be a vertex which lies on all paths between all pairs of 1-center vertices. This vertex must have strictly smaller eccentricity than all the (other) 1-center vertices, which is a contradiction.

which is a contradiction to the fact that S_u^* strictly decreases agent u's cost. ∎

Now we are ready for the proof of Lemma 3.4.6.

Proof (of Lemma 3.4.6) Let (T, α) be any tree network in Max-GE, where T has diameter at least 3. Note that by Lemma 3.4.8, it follows that $\alpha \geq 1$.

We claim that if there is an agent u in $V(T)$ with strategy S_u who can strictly decrease her cost by changing to a strategy S_u^*, where $|S_u| < |S_u^*|$, then there must be an agent p who can strictly decrease her cost by buying *one* edge. This yields a contradiction to (T, α) being in Max-GE.

Observe that in a tree network, no agent can change to a strategy which involves buying less edges than before, since such a change would disconnect the network. Proving the above claim suffices to prove the Lemma, since Badly Connected Trees are exactly those tree networks in Max-GE, where one agent v with strategy S_v can strictly decrease her cost by performing a multi-swap, that is, agent v can change to a strategy S_v^*, where $|S_v| = |S_v^*|$.

Now we prove the claim. Let u be an agent with strategy S_u who can strictly decrease her cost by changing to strategy S_u^*, with $|S_u| < |S_u^*|$. Let (G^*, α) be network induced by agent u's new strategy S_u^*. Let $x_1, \ldots, x_l$ denote the neighbors of u in T and let T_{x_i} denote the maximal subtree rooted at x_i which does not contain u, for all $1 \leq i \leq l$. Let $k = |S_u^*| - |S_u|$ denote the number of additional edges purchased by agent u with her new strategy. Since we assume that S_u^* yields a strict cost decrease for agent u, it follows that S_u^* must decrease agent u's distance-cost by *more than* $k\alpha$. Let D_u denote the set of vertices of T which have maximum distance $\delta_u(T)$ to u. By Lemma 3.4.9, it follows that $2 \leq \delta_u(G^*) < \delta_u(T) - k\alpha$, which yields $\delta_u(T) \geq \lfloor k\alpha \rfloor + 3$. There are two cases:

1. $D_u \not\subset V(T_v)$, for any $v \in \{x_1, \ldots, x_l\}$. In this case there are two vertices p, q, where $p \in V(T_{x_i})$ and $q \in V(T_{x_j})$, for some $i \neq j$, and we have $\delta_u(T) = d_T(u, p) = d_T(u, q)$. Since T is a tree, it follows that $d_T(p, q) = 2\delta_u(T)$ and that $q \in D_p$, where D_p is the set of vertices of T which have maximum distance to p. Observe that p has distance at most $2\delta_u(T) - 2$ to any other vertex in T_{x_i}. This implies that vertex u lies on agent p's shortest paths to any vertex of D_p. If agent p buys the edge pu, then agent p's distance to all vertices which are not in $V(T_{x_i})$ decreases by $\delta_u(T) - 1 \geq \lfloor k\alpha \rfloor + 2 > \alpha$. Furthermore, edge pu yields that agent p's distance to any vertex in $V(T_{x_i})$ is at most $1 + \delta_u(T)$. Since $\delta_u(T) > k\alpha + 2$ and $k \geq 1$ it follows that $1 + \delta_u(T) < 2\delta_u(T) - \alpha$. Thus we have that edge pu increases agent p's edge-cost by α but at the same time it decreases her distance-cost by more than α, which is a contradiction to (T, α) being in Max-GE.

2. $D_u \subset V(T_v)$, for some $v \in \{x_1, \ldots, x_l\}$. Consider agent $p \in D_u$, for which $d_T(u, p) = \delta_u(T) \geq \lfloor k\alpha \rfloor + 3$ holds. Since $D_u \subset V(T_v)$ we have that $d_T(u, w) \leq \delta_u(T) - 1$, for all $w \in V(T) \setminus V(T_v)$. Hence, on the one hand we have that agent p's maximum distance in T to any vertex in $V(T_v)$ is at most $2\delta_u(T) - 2$. On the

other hand, agent p's maximum distance in T to any vertex in $V(T) \setminus V(T_v)$ is at most $2\delta_u(T) - 1$. If agent p buys the edge pv, then we have that her maximum distance to any vertex in $V(T_v)$ decreases by $\delta_u(T) - 2 \geq \lfloor k\alpha \rfloor + 1 > \alpha$. For any other vertex $q \in V(T) \setminus V(T_v)$, we have that edge pv yields a distance of at most $1 + \delta_u(T)$ between p and q. Since $\delta_u(T) > k\alpha + 2$ and $k \geq 1$ we have $1 + \delta_u(T) < 2\delta_u(T) - 1 - \alpha$. Hence, edge pv increases agent p's edge-cost by α but it decreases agent p's maximum distance by more than α, which implies that p can greedily buy the edge pv and thereby strictly decrease her cost. This is a contradiction to (T, α) being in Max-GE. ■

Finally, we can set out for proving Theorem 3.4.4.

Proof (of Theorem 3.4.4) If a Max-GE tree network (T, α) is a Cheap Star, then by definition of a Cheap Star, there is a leaf-agent who can strictly decrease her cost by buying edges to all non-neighboring vertices. Clearly, this implies that a Cheap Star cannot be in Max-NE. Furthermore, by definition of a Badly Connected Tree, we have that in every such tree network, there is an agent who can strictly decrease her cost by swapping $k > 1$ edges simultaneously, which implies that such networks are not in Max-NE. Hence, it remains to show that Cheap Stars and Badly Connected Trees are the only tree networks which can be in Max-GE and at the same time not in Max-NE.

On the one hand, by Lemma 3.4.5, we have that for Max-GE tree networks having at most diameter 2 Cheap Stars are the only tree networks which are not in Max-NE. On the other hand, by Lemma 3.4.6, it follows that among all Max-GE tree networks having diameter at least 3 only Badly Connected Trees are not in Max-NE. Since this case distinction covers every possible diameter, the Theorem follows. ■

We can use the characterization provided by Theorem 3.4.4 to "circumvent" the hardness of deciding whether a tree network is in Max-NE.

Theorem 3.4.10 *For every n-vertex tree network (T, α) it can be checked in $\mathcal{O}(n^4)$ many steps whether (T, α) is in* Max-*NE.*

Proof We can check whether a tree network (T, α) is in Max-NE as follows: First, we compute whether (T, α) is in Max-GE. If this test fails, then, since Max-GEs are a super-class of Max-NEs, we have that (T, α) is not in Max-NE. On the other hand, if (T, α) is in Max-GE, then we have to check whether (T, α) is a Cheap Star or a Badly Connected Tree. If (T, α) is not a Cheap Star and not a Badly Connected Tree, then, by Theorem 3.4.4, we have that (T, α) must be in Max-NE. Otherwise, (T, α) is not in Max-NE.

Computing whether (T, α) is in Max-GE can be done in $\mathcal{O}(n^4)$ steps by checking for every agent if she can strictly decrease her cost by either swapping or buying one own edge. We can neglect edge deletions since such an operation disconnects

the network. An agent may own $\Theta(n)$ may edges, which implies that at most $\mathcal{O}(n^2)$ many edge-swaps are possible. Computing the incurred cost of a strategy can be done in linear time by performing a modified breadth-first-search of the tree network. Since there are $\mathcal{O}(n)$ many possible edge purchases per agent, it follows that we can check in $\mathcal{O}(n^3)$ steps, if an agent can decrease her cost by performing a greedy strategy-change.

Checking if (T, α) is a Cheap Star is possible in $\mathcal{O}(n)$ steps, since we only have to compute the diameter of T and checking if n and α have the right size. Computing whether (T, α) is a Badly Connected Tree is more involved since we have to check if there is an agent who can perform a multi-swap to strictly decrease her cost. This can be done by computing for every agent u the vertices having the maximum distance to u, checking if u owns all edges towards the respective subtrees of T and by computing the 1-center vertices of those subtrees. Finally, by checking if all edges towards subtrees, which contain maximum distance vertices, do not connect to a 1-center vertex of that subtree it can be decided whether agent u can perform a multi-swap which decreases her cost. Computing the vertices having maximum distance to u can be done by a breadth-first-search. Furthermore, there are linear time algorithms for computing the 1-center of an vertex-unweighted tree - see for example the work of Kariv and Hakimi [KH79a]. Hence, we have that checking if agent u can perform a multi-swap to decrease her cost can be done in $\mathcal{O}(n)$ steps.

In total this yields $\mathcal{O}(n^4)$ steps for deciding whether (T, α) is in Max-NE. ∎

We are interested in the stability of tree networks in Max-GE. By Theorem 3.4.4, we only have to analyze the stability of Cheap Stars and Badly Connected Trees to get bounds on the stability on any tree network in Max-GE.

Lemma 3.4.11 *Every Cheap Star is in* 2*-approximate* Max*-NE. Furthermore, this bound is tight.*

Proof We consider any Cheap Star (T, α). Since Cheap Stars have at least 4 vertices, we have that $V(T)$ consists of x, the center of the star, and at least 3 leaves l_1, l_2 and l_3. Let the edge-ownership be arbitrary. Analogously to the proof of Lemma 3.4.1, we have that no leaf agent of T can buy one edge to decrease her cost. Let S_{l_1} denote agent l_1's strategy in (T, α). Let $S^*_{l_1}$ be l_1's strategy which buys all edges towards all non-neighbors in T and let (G^*, α) denote the induced network. We claim that $S^*_{l_1}$ is agent l_1's best strategy. Since every Cheap Star is in Max-GE, we have that agent l_1 cannot delete or swap one edge to decrease her cost. Since she owns at most one edge in (T, α), this rules out all deletion and swapping operations. Analogously to the proof of Lemma 3.4.1, buying exactly one edge does not decrease player l_1's cost either. Note that since $\alpha < \frac{1}{n-2} < 1$, we have that no strategy which yields distance-cost of 2 can have strictly less cost than S_{v_1} for agent l_1. Hence, the claim follows.

If agent l_1 does not own the edge l_1x, then we have

$$\lim_{\alpha\to 0} \frac{c_{l_1}(T,\alpha)}{c_{l_1}(G^*,\alpha)} = \lim_{\alpha\to 0} \frac{2}{(n-2)\alpha+1} = 2.$$

If the edge l_1x is owned by agent l_1 we get

$$\lim_{\alpha\to 0} \frac{c_{l_1}(T,\alpha)}{c_{l_1}(G^*,\alpha)} = \lim_{\alpha\to 0} \frac{\alpha+2}{(n-1)\alpha+1} = 2.$$

Thus, in both cases we have that the approximation ratio approaches 2 as α tends to 0. Clearly, this also represents a tight lower bound of 2 on this ratio. ∎

Lemma 3.4.12 *Every Badly Connected Tree is in $\frac{6}{5}$-approximate* Max-*NE. Furthermore, this bound is tight.*

Proof Remember that Badly Connected Trees are exactly those tree networks in Max-GE, where an agent u can strictly decrease her cost by performing a multi-swap. Clearly, any multi-swap does not change agent u's edge-cost. Thus, to maximize the ratio between agent u's cost in the Badly Connected Tree (T,α) and u's cost after the best possible multi-swap, we have to consider a Badly Connected Tree, where agent u can decrease her distance-cost as much as possible. By definition of a Badly Connected Tree, we have that agent u has at least two vertices p and q in maximum distance $\delta_u(T)$ and we know that p and q lie in different subtrees of u. Observe that, since T is a tree, agent u owns exactly one edge towards each subtree which contains maximum distance vertices. To ensure connectedness of the network, agent u must swap those edges only within their respective subtree. It follows that the best possible multi-swap connects to the middle vertex of the shortest paths to all maximum distance vertices. Let (T^*,α) be the tree network induced by this best possible multi-swap of agent u. It follows that

$$\delta_u(T^*) \geq 1 + \left\lceil \frac{\delta_u(T)-1}{2} \right\rceil \geq \left\lceil \frac{\delta_u(T)}{2} \right\rceil \quad .$$

Let $\delta_u(T) = k$ and let agent u own $j \geq 2$ edges in T. We have

$$\frac{c_u(T,\alpha)}{c_u(T^*,\alpha)} \leq \frac{j\alpha+k}{j\alpha+\left\lceil \frac{k}{2} \right\rceil} \leq \frac{2\alpha+k}{2\alpha+\left\lceil \frac{k}{2} \right\rceil} \quad .$$

Note that this ratio is maximized for a Badly Connected Tree (T',α), where an agent $u \in V(T')$ can decrease her distance-cost from k to $\left\lceil \frac{k}{2} \right\rceil$ and where α is as small as possible. However, we cannot simply choose $\alpha = 1$ since we have to ensure that (T',α) remains in Max-GE.

We explicitly construct (T',α), which will serve at the same time as lower and upper bound construction. Clearly, T' must consist of a path P of length $2k$, where

agent u is the middle vertex of this path. Without loss of generality, we can choose an odd k. Furthermore, to avoid greedy edge-swaps, we assume that for every edge xy in P we have that x owns xy if x is closer to u than y. Thus, on path P we have that u is the only agent who owns two edges. Now, observe that (P, α) is already stable against greedy deletions and greedy swaps. No agent $x \in V(P)$ can swap any single edge to decrease her cost, since the only owned edge is by construction an edge which does not lie on all shortest paths from x to the vertices having maximum distance to x. However, agents may improve their cost by buying one additional edge if α is small enough. To rule out this possibility, we consider a leaf agent l and choose α in such a way that l cannot decrease her cost by buying one edge. Note that leaf agents have the highest distance-cost in P, which implies that they are exactly those agents which are most susceptible to single edge purchases. Thus, if no leaf agent of P can decrease her cost by buying one edge, then no other agent can.

Let l_1 and l_2 be the to leaf agents of P. Observe that l_1's best possible additional edge connects to some vertex z which lies on the path between u and l_2. Hence, this edge decreases agent l_1's distance-cost by at least $k-1$. We choose α such that it will neutralize this decrease in distance-cost. It follows that to minimize α, we have to ensure that z is as close as possible to u. We force z towards u by adding two branches to P as follows: Let p_1 and p_2 denote the vertices which lie in the middle of the path from u to l_1 and l_2, respectively. Thus, we have $d_P(u, p_1) = d_P(u, p_2) = \left\lceil \frac{k}{2} \right\rceil$. To finally obtain T', we connect both vertices p_1, p_2 to a path of length $\left\lceil \frac{k}{2} \right\rceil - 1$, respectively. Again, the ownership on these paths resembles the edge ownership on P, that is, the respective vertex closer to u owns the edge. Let l_1' and l_2' be the leaves of the newly attached paths. Note that these new paths do not change agent u's distance decrease. See Fig. 3.6 for an illustration.

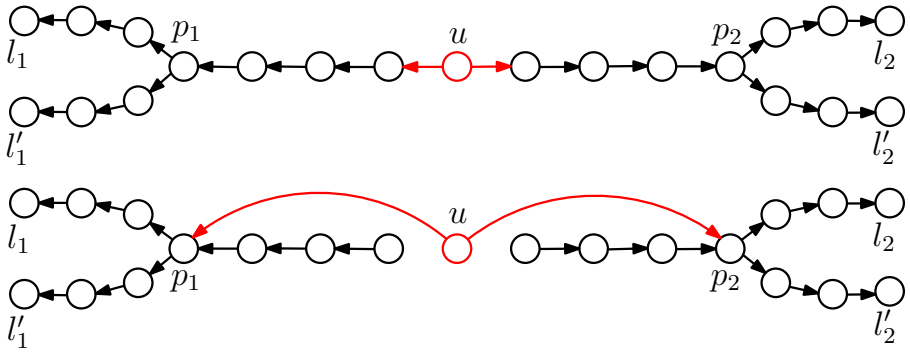


Figure 3.6.: The network (T', α) for $k = 7$ before and after agent u's best multi-swap.

In (T', α), we have that $d_{T'}(l_1, l_1') = k-1$. It follows that every possible additional edge of agent l_1 only yields a distance decrease of at most $k+1$. Thus, setting $\alpha \geq k+1$, implies that no agent in (T', α) can decrease her cost by buying one

edge.

Now we are ready to finally settle the approximation ratio of Badly Connected Trees. For agent u in (T', α) we have

$$\frac{6}{5} = \lim_{k\to\infty} \frac{2(k-1)+k}{2(k-1)+\left\lceil \frac{k}{2} \right\rceil} \geq \lim_{k\to\infty} \frac{c(u)}{c^*(u)} \geq \lim_{k\to\infty} \frac{2(k+1)+k}{2(k+1)+\left\lceil \frac{k}{2} \right\rceil} = \frac{6}{5} \ ,$$

where the limit on the left side represents the upper bound with $\alpha = k-1$ and the limit on the right represents the lower bound with $\alpha = k+1$. Both bounds match if we let k tend to infinity. ∎

Combining Theorem 3.4.4 with Lemma 3.4.11 and Lemma 3.4.12 we arrive at the following:

Theorem 3.4.13 *Let (T, α) be a tree network in* Max-*GE. If T has diameter at most 2, then (T, α) is in 2-approximate* Max-*NE. If T has diameter at least 3, then (T, α) is in $\frac{6}{5}$-approximate* Max-*NE. Moreover, both bounds are tight.*

3.4.2. Non-Tree Networks in Max Greedy Equilibrium

Fig. 3.5 (middle) shows that there are non-tree networks in Max-GE, which are not in Max-NE. We want to quantify the loss in stability of Max-GEs versus Max-NEs. For tree networks we have that Cheap Stars play a crucial role. These networks owe their instability to a multi-buy operation and to the fact that they are in Max-GE for arbitrarily small α. We generalize this property of Cheap Stars to non-tree networks.

Definition 3.4.14 (Cheap Network) *A network (G, α) in* Max-*GE, is called a* Cheap Network*, if (G, α) remains in* Max-*GE when α tends to 0.*

Cheap Stars yield a lower bound on the stability approximation ratio which equals their diameter. We can generalize this observation:

Theorem 3.4.15 *If there is Cheap Network (G, α) having diameter D, then there is an α^* such that the network (G, α^*) is in* Max-*GE but not in β-approximate* Max-*NE for any $\beta < D$.*

Proof Consider a Cheap Network (G, α), where G has diameter D and let u be any vertex of G having eccentricity D. Let j denote the number of edges, which are owned by agent u in (G, α). Thus, we have that agent u has cost $j\alpha + D$. Now we consider the strategy-change of agent u towards the strategy which buys an edge to all vertices of G which do not own an edge to u. Clearly, after the strategy-change agent u incurs cost of at most $(n-1)\alpha + 1$.

Now, observe that since (G, α) is a Cheap Network, we have that (G, α) remains in Max-GE when α tends to 0. Hence, $\lim_{\alpha\to 0} \frac{j\alpha+D}{(n-1)\alpha+1} = D$, which implies that for all $\beta < D$ there is an α^* such that $j\alpha^* + D > \beta(n-1)\alpha^* + 1$. Hence, (G, α^*) is in Max-GE but not in β-approximate Max-NE for any $\beta < D$. ∎

Lemma 3.4.16 *There is a Cheap Network having diameter* 4.

PROOF We construct the Cheap Network $(\tilde{G}, \alpha)$ as follows: The graph $\tilde{G}$ has 24 vertices $u_0, \ldots, u_7, v_0, \ldots, v_7, w_0, \ldots, w_7$ and the vertices $u_0, \ldots, u_7$ form a cycle, where u_i owns the edge towards u_{i+1}, for $0 \leq i \leq 6$, and u_7 owns the edge to u_0. Furthermore, for $0 \leq j \leq 7$ we have that agent v_j owns an edge to u_j and w_j and agent w_j owns an edge towards u_k, where $k = (j + 4) \mod 8$. See Fig. 3.7 for an illustration.

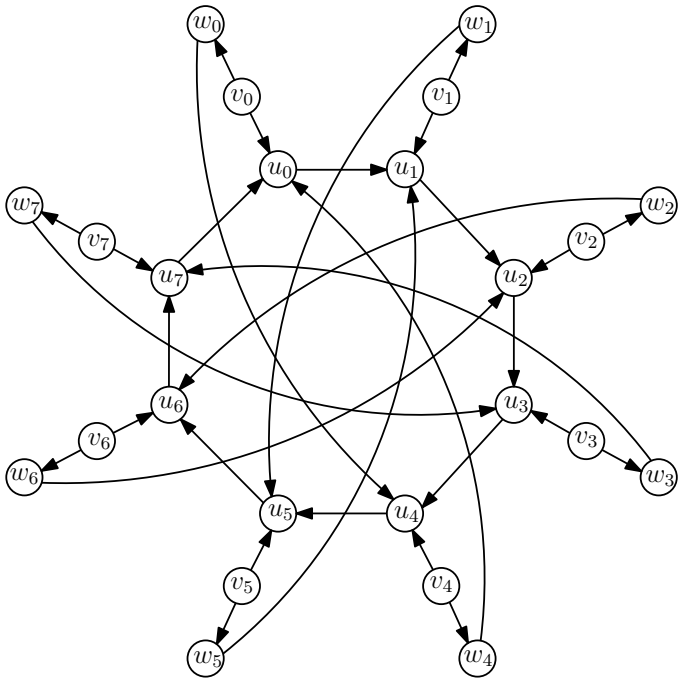


Figure 3.7.: The Cheap Network $(\tilde{G}, 1)$ having diameter 4.

For showing that $(\tilde{G}, \alpha)$ is a Cheap Network, we have to show that it is in MAX-GE for some α and that it remains in MAX-GE if α tends to 0. We begin by proving that $(\tilde{G}, \alpha)$ is in MAX-GE for $\alpha = 1$.

Since $(\tilde{G}, 1)$ is highly symmetric, it suffices to show that agents u_0, v_0 and w_0 cannot strictly decrease their cost by performing a greedy strategy-change. It is easy to see that none of them can decrease her cost by deleting one own edge, since any such deletion increases the respective agent's distance-cost by at least 1. Since $\alpha = 1$, this does not yield a strict cost decrease.

Next, we show that none of the three agents can swap or buy an own edge and thereby strictly decrease her cost. Consider agent u_0, who owns exactly one edge. Now, observe that u_0's edge u_0u_1 is the first edge on u_0's shortest paths to the vertices w_1 and v_5 to which u_0 has maximum distance. Furthermore, observe that there is no vertex in $\tilde{G}$ which is a neighbor to both w_1 and v_5. Thus, no swap can simultaneously decrease agent u_0's distance to w_1 and v_5. Agent u_0 also has vertex

w_2 in maximum distance 3. Since $d_{\tilde{G}}(w_1, w_2) = 4$, it follows that u_0 cannot buy any edge, which strictly decreases u_0's distances to both w_1 and w_2 simultaneously. Hence, u_0 cannot greedily purchase an edge to strictly decrease her cost.

Agent v_0 has, among others, vertices v_2, v_3, v_5 and v_6 in maximum distance 4. We have $d_{\tilde{G}}(v_2, v_3) = d_{\tilde{G}}(v_5, v_6) = 3$ and $d_{\tilde{G}}(v_2, v_5) = d_{\tilde{G}}(v_3, v_6) = 4$. Thus, the best possible swap or edge purchase of v_0, which strictly decreases the distances from v_0 to v_2 and v_3 simultaneously, must connect to a neighbor x of v_2 or v_3. It follows that either $d_{\tilde{G}}(x, v_5) \geq 3$ or $d_{\tilde{G}}(x, v_6) \geq 3$. Thus, such a swap or edge purchase does not reduce v_0's distances to all of the four vertices v_2, v_3, v_5, v_6. Any improving swap or edge purchase must strictly decrease v_0's distance-cost, which implies that such an edge must connect to a neighbor of v_2 or v_3, since this is the only way to decrease the distance to both of them. It follows that v_0 cannot swap or buy any own edge to decrease her cost.

Agent w_0 has the vertices $v_1, v_2.v_6$ and v_7 in maximum distance 4. We have $d_{\tilde{G}}(v_1, v_2) = d_{\tilde{G}}(v_6, v_7) = 3$ and $d_{\tilde{G}}(v_1, v_6) = d_{\tilde{G}}(v_2, v_7) = 4$. Hence, w_0 faces essentially the same situation as v_0 and an analogous argument shows that w_0 cannot swap or buy an edge to decrease her cost. This establishes that $(\tilde{G}, 1)$ is in Max-GE.

We have argued above that any edge deletion increases the distance-cost of the moving agent by 1. Furthermore, we have shown that no swap or edge purchase can strictly decrease any agent's distance-cost. This implies that $(\tilde{G}, \alpha)$ is in Max-GE for any $\alpha \leq 1$. Hence $(\tilde{G}, \alpha)$ is a Cheap Network having diameter 4. ∎

Remark 3.4.17 *The Cheap Network $(\tilde{G}, \alpha)$ is not only stable against greedy strategy-changes, it is even stable against any strategy-change. That is, $(\tilde{G}, \alpha)$ is in* Max-*NE for any $\alpha \leq 1$. To the best of our knowledge, this is the first known non-tree* Max-*NE network having diameter* 4.

Corollary 3.4.18 *For $\alpha < 1$ there is a network (G, α) in* Max-*GE, which is not in β-approximate* Max-*NE for any $\beta < 4$.*

Now we consider the case, where $\alpha \geq 1$. Quite surprisingly, it turns out that this case yields a very high lower bound on the approximation ratio.

Theorem 3.4.19 *For $\alpha \geq 1$ there is a* Max-*GE network (G, α) having n vertices, which is not in β-approximate* Max-*NE for any $\beta < \frac{n-1}{5}$.*

We give a family of networks in Max-GE each having an agent u who can decrease her cost by a factor of $\frac{5}{n-1}$ by a non-greedy strategy-change. The network (G_1, α) can be obtained as follows: $V(G_1) = \{u, v, l_1, l_2, a_1, a_2, b_1, b_2, x_1, y_1\}$ and agent u owns edges to a_1, a_2 and x_1. For $i \in \{1, 2\}$, agent b_i owns an edge to v and to a_i and agent l_i owns an edge to b_i. Finally, agent y_1 owns an edge to x_1 and to v. Fig. 3.8 (left) provides an illustration. To get the k-th member of the family, for $k \geq 2$, we simply add the vertices x_j, y_j, for $2 \leq j \leq k$, and let agent y_j own edges towards x_j and v. See Fig. 3.8 (right).

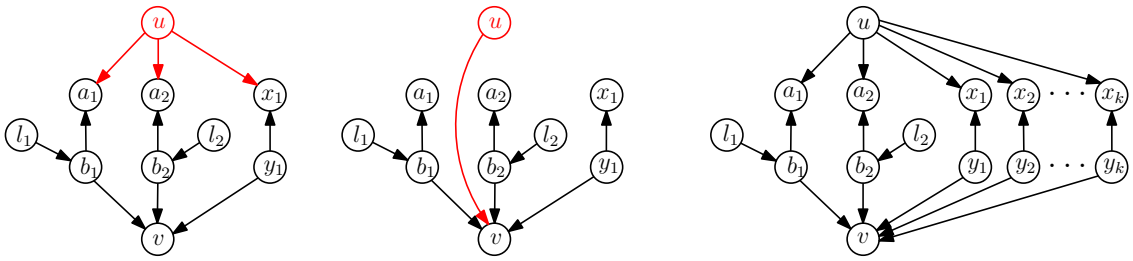


Figure 3.8.: (G_1, α) before (left) and after (middle) agent u's non-greedy strategy-change and the network (G_k, α) (right).

Lemma 3.4.20 *Each of the networks (G_i, α), as described above, is in* Max-*GE for $1 \leq \alpha \leq 2$.*

Proof The statement is proven as follows: If any agent of (G_1, α) deletes one own edge, then either this operation disconnects the network or her distance-cost increases by 2. Since deleting an own edge decreases the edge-cost by $\alpha \leq 2$, we have that such a move cannot yield a strict cost decrease for any agent.

Next, we show that no agent can swap an own edge to strictly decrease her cost. Clearly, agents a_1, a_2, v and $x_1, \ldots, x_k$ cannot swap any edge since they do not own one. By symmetry of the construction, we only have to show that agents u, b_1, l_1 and y_1 cannot decrease their cost by swapping one own edge. Agent u has vertices l_1, l_2 and v in maximum distance 3. But, since $d_{G_k}(l_1, l_2) = 4$, it is impossible for u to swap any own edge such that the distance to *all* three of them is strictly decreased. Agent b_1 has vertices l_2, a_2 and x_1 in maximum distance 3. But since $d_{G_k}(l_2, x_1) = 4$, no swap can decrease b_1's distance to all of them. Analogously, the same holds true for agent y_1, who has l_1, l_2, a_1 and a_2 in maximum distance 3. Agent l_1 cannot improve by swapping her edge, since b_1 is a 1-center vertex of the graph $G_k - l_1$ and for a leaf vertex it is clearly optimal to connect to a 1-center of the remaining network.

Finally, let us focus on greedy edge purchases in (G_k, α). Since $\alpha \geq 1$, it follows that greedily buying one edge can strictly decrease an agent's cost only if this operation decreases the distance-cost of that agent by more than α, that is, by at least 2. Clearly, for all agents of (G_k, α) which have eccentricity 3, this is impossible. Now we consider all other agents, which all have eccentricity 4 and we show that none of them can buy an edge to decrease her distance-cost by more than 1. By symmetry, it suffices to argue for agents l_1, a_1 and x_1. Agent l_1 has vertices l_2 and x_1 in maximum distance 4. To decrease both distances simultaneously, agent l_1 must buy an edge towards a vertex, which lies on both shortest paths from l_1 to l_2 and from l_1 to x_1. Indeed, vertex v is such a vertex and it is easy to see that it is the only non-neighbor of l_1, which lies on both shortest paths. But buying an edge towards v decreases l_1's distance-cost only by 1. Agent a_1 only has vertex l_2 in maximum distance 4. There are two shortest paths from a_1 to l_2 which both use

vertex b_2. However, buying an edge towards b_2 only yields a distance decrease of 1 for agent a_1. The same holds true for an edge towards a_2 or v, respectively. Thus, no edge to any non-neighboring vertex on a_1's shortest paths to l_2 can decrease a_1's distance-cost by more than 1. Agent x_1 has vertices l_1 and l_2 in maximum distance 4. But, analogously to agent l_1's situation, there is no vertex which simultaneously lies on a shortest path from x_1 to l_1 and on a shortest path from x_1 to l_2 and which has distance 1 to l_1 and l_2. Thus, agent x_1 can decrease her distance-cost by buying one edge by at most 1. ■

Proof (of Theorem 3.4.19) We focus on agent u in the network (G_k, α) and show that this agent can change her strategy in a non-greedy way and thereby decrease her cost by a factor of $\frac{n-1}{5}$, where n is the number of vertices of G_k. Let S_u be agent u's current strategy in (G_k, α) and let S_u^* be u's strategy which only buys one edge towards vertex v and let (G_k^*, α) be the corresponding network. See Fig 3.8 (left and middle). For $\alpha = 2$, we have

$$\frac{c_u(G_k, \alpha)}{c_u(G_k^*, \alpha)} = \frac{\alpha(2+k)+3}{\alpha+3} = \frac{7}{5} + \frac{2k}{5} = \frac{n-1}{5} \quad ,$$

where the last equality follows since $k = \frac{n-8}{2}$, by construction. ■

Corollary 3.4.21 *Uncapacitated Metric Min-Max Facility Location*[6] *has a locality gap of* $\frac{n-1}{5}$*, where* n *is the number of clients.*

Proof The corollary follows by using the "locality gap preserving" reduction provided in the proof of Theorem 3.3.9 and the lower bound of Theorem 3.4.19.

The lower bound construction of Theorem 3.4.19 can be transformed into an instance of uncapacitated metric min-max facility location. Remember that we have cost-equality and that greedy strategy-changes of agent u in the NCG transfer one to one to greedy modifications of the facility location solution. Thus, we have the property that the corresponding solution to the facility location problem is locally optimal but resembles only a $\frac{n-1}{5}$-approximation to the globally optimal solution. ■

[6] This problem is definied exactly like the Uncapacitated Metric Facility Location problem [Vaz01] but with a max-operator instead of the sum-operator in the objective function.

4. The Dynamics of Selfish Network Creation

Network Creation Games and several variants, as defined in Section 2.2, have been studied intensively, but, to the best of our knowledge, almost all these works exclusively focus on properties of the equilibrium states of the game[1]. With this focus, the game is usually considered to be a one-shot simultaneous-move game.

However, the Internet and social networks are not created in "one shot". It evolved from an initial network, the ARPANET, into its current shape by repeated infrastructural changes performed by selfish agents who entered or left the stage at some time in the process. For this reason, we focus on a more dynamic point of view: We analyze the properties of the network creation *processes* induced by the sequential-move versions of the known model of selfish network creation.

It is well-known that Network Creation Games have low Price of Anarchy, which implies that the social cost of the worst stable states arising from selfish behavior is close to the cost of the social optimum. Therefore these games are appealing for the decentralized and selfish creation of networks which optimize the service quality for all agents at low infrastructural cost, e.g. overlay networks created by selfish peers. But, to the best of our knowledge, it is not known how a group of agents can collectively *find* such a desirable stable state. Analyzing the game dynamics of Network Creation Games is equivalent to analyzing a very natural search strategy: (uncoordinated) distributed local search, where in every step some agent myopically modifies the network infrastructure to better suit her needs. Clearly, if at some step in the process no agent wants to modify her part of the network, then a stable network has emerged.

4.1. Preliminaries

4.1.1. Additional Definitions

We consider several versions of a network creation process performed by n selfish agents. In all versions we consider networks, where every node corresponds to an agent and undirected links connect network nodes. The creation process is based on an underlying Network Creation Game (NCG) and can be understood as a dynamic process where agents sequentially perform strategy-changes in the NCG.

[1] See Section 5.1.2 for a detailed summary.

As explained in Section 2.2, the strategies of the agents determine which links are present in the network and any strategy-profile, which is a vector of the strategies of all n agents, determines the induced network. But this also works the other way round: Given some network (G, α) with edge-ownership information then this determines the current strategies of all agents in the network. Thus, starting from a network (G_0, α), any sequence of strategy-changes by agents can thus be seen as a sequence of networks $(G_0, \alpha), (G_1, \alpha), (G_2, \alpha), \ldots$, where the network (G_{i+1}, α) arises from the network (G_i, α) by the strategy-change of exactly one agent.

The creation process starts in an initial state (G_0, α), which we call the *initial network*. A step from state (G_i, α) to state (G_{i+1}, α) consists of a *move* by one agent. A move of agent u in state (G_i, α) is the replacement of agent u's pure strategy in (G_i, α) by another *admissible* pure strategy of agent u. The induced network after this strategy-change by agent u then corresponds to the state (G_{i+1}, α). We consider only improving moves, that is, strategy-changes which strictly decrease the moving agent's cost.

The cost of an agent in G_i depends on the structure of (G_i, α) as defined in Section 2.2. If agent u in state (G_i, α) has an admissible new strategy which yields a strict cost decrease for her, then we call agent u *unhappy in network* (G_i, α) and we let U_i denote the set of all unhappy agents in state (G_i, α). Only one agent can actually move in a state of the process and this agent $u \in U_i$, whose move transforms (G_i, α) into (G_{i+1}, α), is called *the moving agent in network* (G_i, α). In any state of the process the *move policy* determines which agent is the moving agent. The process stops in some state (G_j, α) if no agent wants to perform a move, that is, if $U_j = \emptyset$, and we call the resulting networks (swap-)stable.

Depending on what strategies are admissible for an agent in the current state, we get the different game types Swap Game (SG), Asymmetric Swap Game (ASG), Greedy Buy Game (GBG) and Buy Game (BG) (or Network Creation Game) as defined in Section 2.2. Note that the (swap-)stable networks which may emerge at the end of the network creation process correspond to pure Nash Equilibria if the underlying game is the Buy Game, the Greedy Buy Game or the Asymmetric Swap Game. They correspond to Swap Equilibria, if the underlying game is the Swap Game. We will omit the reference to the edge-cost parameter α whenever we focus on the SG or ASG.

The *move policy* specifies for any state of the process, which of the unhappy agents is allowed to perform a move. From a mechanism design perspective, the move policy is a way to enforce coordination and to guide the process towards a stable state. We will focus on the *max cost policy*, where an agent having the highest cost is allowed to move and ties among such agents are broken arbitrarily. Sometimes we will assume that an adversary chooses the worst possible moving agent. Note that the move policy only specifies who is allowed to move, not which specific move has to be performed. We do not consider such strong policies since we do not want to restrict the agents' freedom to act.

Any combination of an underlying game (SG, ASG, GBG or BG), the two distance

functions (SUM and MAX) and some move policy together with an initial network completely specifies a network creation process. If not stated otherwise, edge-costs cannot be shared by agents.

A cyclic sequence of networks $(C_1, \alpha), \ldots, (C_j, \alpha)$, where network $(C_{i+1 \bmod j}, \alpha)$ arises from network $(C_{i \bmod j}, \alpha)$ by an improving move of one agent is called a *better response cycle*. If every move in such a cycle is a *best response move*, which is a strategy-change towards an admissible strategy which yields the largest cost decrease for the moving agent, then we call such a cycle a *best response cycle*. Clearly, a best response cycle is a better response cycle, but the existence of a better response cycle does not imply the existence of a best response cycle.

4.1.2. Classifying Games According to their Dynamics

Analyzing the convergence processes of games is a very rich and diverse research area. We will briefly introduce the two well-known classes of finite strategic games: *games having the finite improvement property* (FIPG) [MS96] and *weakly acyclic games* (WAG) [You93] as well as several subclasses of them.

FIPG have the most desirable form of dynamic behavior: Starting from any initial state, every sequence of improving moves must eventually converge to an equilibrium state of the game, that is, such a sequence must have finite length. Thus, in such games distributed local search is guaranteed to succeed. It was shown by Monderer and Shapley [MS96] that a finite game is a FIPG if and only if there exists a *generalized ordinal potential function* Φ, which maps strategy-profiles to real numbers and has the property that if the moving agent's cost decreases, then the potential function value decreases as well. Stated in our terminology, this means that $\Phi : \mathcal{G}_n \to \mathbb{R}$, where $\mathcal{G}_n$ is the set of all networks on n nodes, and we have

$$c_u(G_i, \alpha) - c_u(G_{i+1}, \alpha) > 0 \Rightarrow \Phi((G_i, \alpha)) - \Phi((G_{i+1}, \alpha)) > 0 \ ,$$

if agent u is the moving agent in the network (G_i, α). Clearly, no FIPG can admit a better response cycle. An especially nice subclass of FIPG are games that are guaranteed to converge to a stable state in a number of steps which is polynomial in the number of players. We call this subclass poly-FIPG.

Another well-known subclass of FIPG is the class of *ordinal potential games* (OPG) [MS96]. OPGs admit an *ordinal potential function* $\Psi : \mathcal{G}_n \to \mathbb{R}$, with the stronger constraint that

$$c_u(G_i, \alpha) - c_u(G_{i+1}, \alpha) > 0 \iff \Psi((G_i, \alpha)) - \Psi((G_{i+1}, \alpha)) > 0 \ ,$$

if agent u is the moving agent in the network (G_i, α). Clearly every ordinal potential function is a generalized ordinal potential function.

Weakly acyclic games (WAG) are a super-class of FIPG. Here it is not necessarily true that *any* sequence of improving moves must converge to an equilibrium but we have that from any initial state there exists *some* sequence of improving moves

which enforces convergence. Thus, with some additional coordination distributed local search may indeed lead to stable states for such games. A subclass of WAG are games where from any initial state there exists a sequence of best response moves, which leads to an equilibrium. We call those games *weakly acyclic under best response*, BR-WAG for short. Observe that if a game is not weakly acyclic, then there is *no* way of enforcing convergence if agents stick to playing improving moves.

The above mentioned classes of finite strategic games are related as follows:

$$\begin{array}{rl} \text{poly-FIPG} & \subsetneq \\ & \text{FIPG} \subset \text{BR-WAG} \subset \text{WAG} \\ \text{OPG} & \subsetneq \end{array}$$

The story does not end here. Very recently, Apt and Simon [AS12] have classified WAG in much more detail by introducing a "scheduler", which is a moderating super-player who guides the agents towards an equilibrium.

4.1.3. Related Work

Most of the previous work on Network Creation Games focuses on properties of stable networks or on the complexity of computing an agent's best response. To the best of our knowledge, the dynamic behavior of most of these variants, including best response dynamics in the well-studied original model [FLM+03], has not yet been analyzed.

Previous work, e.g. [FLM+03, AEED+06, DHMZ12, MS13], has shown that the Price of Anarchy for the Sum-BG and the Max-BG is constant for a wide range of α and in $2^{\mathcal{O}(\sqrt{\log n})}$ in general. For the Sum-(A)SG the best upper bound is in $2^{\mathcal{O}(\sqrt{\log n})}$ as well [ADHL13, MS12], whereas the Max-SG has a lower bound of $\Omega(\sqrt{n})$ [ADHL13]. Interestingly, if played on trees, then the Sum-SG and the Max-SG have constant Price of Anarchy [ADHL13], whereas the Sum-ASG and the bounded budget version on trees has Price of Anarchy in $\Theta(\log n)$ [EFM+11, MS12]. Moreover, it is easy to show that the Max-ASG on trees has Price of Anarchy in $\Theta(n)$. Thus, we have the desirable property that selfish behavior leads to a relatively small deterioration in social welfare for most of the proposed versions.[2]

Very recently, Cord-Landwehr, Hüllmann, Kling and Setzer [CLHKS12] studied a variant of the Max-SG where agents have communication interests and showed that this variant admits a best response cycle on a tree network as initial network. Hence the restricted-interest variant of the Max-SG is not a FIPG – even on trees.

Brandes, Hoefer and Nick [BHN08] were the first to observe that the Sum-BG is not a FIPG and they prove this by providing a *better* response cycle. Very recently, Bilò, Gualà, Leucci and Proietti [BGLP12] gave a *better* response cycle for the

[2] See Section 5.1.2 for a more detailled discussion.

MAX-BG which implies the same statement for this version. Note that both proofs contain agents who perform a sub-optimal move at some step in the better response cycle. Hence, these two results do not address the convergence behavior if agents play optimally.

It was shown in [FLM+03] that computing a best response in the SUM-BG is NP-hard. In [MS13] the NP-hardness of this problem for the MAX-version was established. Independently from us Ehsani, Fazli, Mehrabian, Sadeghabad, Safari, Saghafian and ShokatFadaee [EFM+11] came up with a NP-hardness proof for computing a best response in the bounded budget version of the SUM-BG and the MAX-BG. Their results imply NP-hardness for the corresponding problems in all versions of Swap Games since agents will always use up their budget. Interestingly, they also reduce from the p-MEDIAN problem [KH79b].

4.1.4. Our Contribution

In this chapter, we study Network Creation Games, as proposed by Fabrikant, Luthra, Maneva, Papadimitriou and Shenker [FLM+03], and several natural variants of this model from a new perspective. Instead of analyzing properties of equilibrium states, we apply a more constructive point of view by asking if and how fast such desirable states can be found by selfish agents. For this, we turn the original model, which was originally formulated as one-shot simultaneous-move game, into several more algorithmic models, where moves are performed sequentially.

We study the dynamics of the SUM-SG and the MAX-SG and show that if the initial network is a tree on n nodes, then the network creation process is guaranteed to converge in $\mathcal{O}(n^3)$ steps for both versions. By employing the max cost policy and enforcing best responses in the SUM-SG on trees, this process can be sped up significantly to $n + \lfloor \frac{n}{2} \rfloor - 5$ steps, which is asymptotically optimal and close to optimal in the exact number of steps. The same move policy with best responses yields a significant speed-up in the MAX-version as well: we prove an upper bound of $\Theta(n \log n)$ steps, which is almost asymptotically optimal. Moreover, we show that these results carry over to the Asymmetric Swap Game on trees in both the SUM- and the MAX-version.

These positive results for initial networks which are trees are contrasted by several strong negative results on general networks. We show that the SUM-SG, the MAX-SG, the SUM-ASG and the MAX-ASG on general networks are *not* guaranteed to converge if agents repeatedly perform best possible improving moves and, even worse, for the latter three versions we show that *no* move policy can enforce convergence. We show that these games are not in FIPG, which implies that there cannot exist a generalized ordinal potential function which "guides" the way towards an equilibrium state. For the SUM-ASG we show the even stronger negative result that it can happen that *no* sequence of best response moves may enforce convergence, that is, the SUM-ASG is not even weakly acyclic under best response. If not all possible edges can be created, that is, if we have a non-complete *host*

graph [DHMZ09, BGLP12], then we show that the Sum-ASG and the Max-ASG on non-tree networks is not weakly acyclic. Moreover, we map the boundary between convergence and non-convergence in ASGs and show the surprising result that cyclic behavior can already occur in n-vertex networks which have n edges. That is, even one non-tree edge suffices to completely change the dynamic behavior of these games. In our constructions we have that every agent owns exactly one edge, which is equivalent to the uniform-budget case introduced by Ehsani, Fazli, Mehrabian, Sadeghabad, Safari, Saghafian and ShokatFadaee [EFM+11]. In their paper the authors raise the open problem of determining the convergence speed for the bounded-budget version. Thus, our results answer this open problem – even for the simplest version of these games – in the negative, since we show that no convergence guarantee exists.

We provide best response cycles for all versions of the Buy Game, which implies that these games have no convergence guarantee – even if agents have the computational resources to repeatedly compute best response strategies. To the best of our knowledge, the existence of *best* response cycles for all these versions was not known before.

See Table 4.1 for an overview of our convergence results for Network Creation Games without cost sharing.

Furthermore, we investigate the version where bilateral consent is needed for edge-creation and where the edge-cost is shared equally among its endpoints. We show that this version exhibits a similar undesirable dynamic behavior as the unilateral version. Quite surprisingly, we can show an even stronger negative result in the Sum-version which implies the counter-intuitive statement that cost-sharing may lead to worse dynamic behavior. Our findings nicely contrast a result of Corbo and Parkes [CP05] who have shown guaranteed convergence if agents repeatedly play best response strategies against *perturbations* of the other agents' strategies. We show that these perturbations are necessary for achieving convergence.

Along the way, we observe that the hardness of computing a best response in the Sum-(A)SG behaves similarly as the convergence property. We give a linear time algorithm for computing a best response on tree networks - even if agents are allowed to swap multiple edges at a time. This is contrasted with the result that this task is NP-hard even on simple general networks if agents can perform multi-swaps.

Finally, we present a careful empirical study of the convergence time in the ASG and in the GBG. Interestingly, our simulations show that our negative theoretical results seem to be confined to a small set of pathological instances. Even more interesting may be that our simulations show a remarkably fast convergence towards stable networks in $\mathcal{O}(n)$ steps, where n is the number of agents. This indicates that despite our negative results distributed local search may be a suitable method for selfish agents for collectively finding equilibrium networks.

Table 4.1.: Summary of convergence results for the SUM and MAX-versions. "Tree" stands for tree networks as initial network. "Arbitrary" is the case where the initial network is an arbitrary connected network. "Host" is the case where a host network is given and only edges from the host network may be used. BB abbreviates "bounded-budget" [EFM+11], BRC abbreviates "best response cycle".

	SUM-SG	SUM-ASG	BB SUM-(G)BG	SUM-(G)BG
tree	poly-FIPG (Thm. 4.2.7)	poly-FIPG (Cor. 4.4.1)		BRC (Thm. 4.5.1)
arbitrary	BRC (Thm. 4.2.20)	$\notin$ BR-WAG (Thm. 4.4.3)	BRC (Thm. 4.4.7)	BRC (Thm. 4.5.1)
host		$\notin$ WAG (Cor. 4.4.6)		$\notin$ WAG (Cor. 4.5.2)

	MAX-SG	MAX-ASG	BB MAX-(G)BG	MAX-(G)BG
tree	poly-FIPG (Thm. 4.3.1)	poly-FIPG (Cor. 4.4.1)		BRC (Thm. 4.5.1)
arbitrary	BRC, no move-policy (Thm. 4.3.16)	BRC, no move-policy (Thm. 4.3.16)	BRC (Thm. 4.4.7)	BRC (Thm. 4.5.1)
host		$\notin$ WAG (Cor. 4.4.6)		$\notin$ WAG (Cor. 4.5.2)

4.2. Dynamics in Sum-Swap Games

We consider the SUM-SG. Remember that the cost of an agent u in a connected network G in this version is $c_u(G) = \sum_{w \in V(G)} d_G(u, w)$ and that both endpoints of an edge may swap the edge. Recall that we use $|G|$ to denote the number of vertices in network G.

4.2.1. Dynamics on Trees

In this subsection we analyze the special case where the initial graph G_0 is a tree. We show that the SUM-SG on trees belongs to the well-studied class of *ordinal potential games* (OPG) [MS96]. This guarantees the desirable property that Swap Equilibria can be found by distributed local search.

Theorem 4.2.1 *The* SUM*-SG on trees is an ordinal potential game.*

Before we prove the Theorem, we analyze the impact of an edge-swap on the individual cost of the swapping agent and on the social cost.

Let $T = (V, E)$ be a tree having n vertices. Assume that agent u performs the edge-swap uv to uw in the tree T. (Note that this implies that $uw \notin E$). Let T' be the tree obtained after this edge-swap. Let

$$\Psi(G) = \sum_{u \in V(G)} c_u(G) = \sum_{u \in V(G)} \sum_{w \in V(G)} d_G(u, w)$$

denote the *social cost* of network G. Hence, $\Psi(T)$ and $\Psi(T')$ denotes the social cost of T and T', respectively. Let T_u and T_v be the tree T rooted at u and v, respectively. Let A be the subtree rooted at u in T_v and let B be the subtree rooted at v in T_u. See Fig. 4.1 for an illustration.

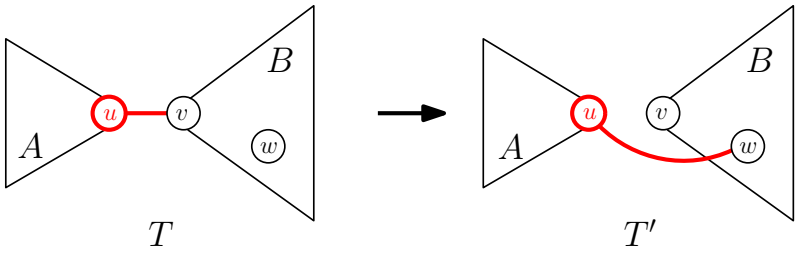


Figure 4.1.: Agent u swaps edge uv to edge uw.

Lemma 4.2.2 *The change in agent u's cost induced by the edge-swap uv to uw is*

$$\Delta(u) = c_v(B) - c_w(B) \ .$$

Proof We have

$$\begin{aligned} c_u(T) &= \sum_{x \in V(A)} d_T(u, x) + \sum_{y \in V(B)} d_T(u, y) = \sum_{x \in V(A)} d_A(u, x) + \sum_{y \in V(B)} (1 + d_B(v, y)) \\ &= \sum_{x \in V(A)} d_A(u, x) + |B| + \sum_{y \in V(B)} d_B(v, y) = \sum_{x \in V(A)} d_A(u, x) + |B| + c_v(B) \ . \end{aligned}$$

Analogously, we obtain $c_u(T') = \sum_{x \in V(A)} d_A(u, x) + |B| + c_w(B)$. Thus, we have

$$\Delta(v) = c_u(T) - c_u(T') = c_v(B) - c_w(B) \ . \qquad \blacksquare$$

The following lemma implies the desired property that a local improvement of an agent yields a global improvement in terms of social cost.

Lemma 4.2.3 *The change in social cost induced by the edge-swap uv to uw is*

$$\Delta(\Psi) = \Psi(T) - \Psi(T') = 2|A|\Delta(v) \ .$$

PROOF First, we analyze $\Psi(T)$ in terms of the subtrees A and B:

$$\begin{aligned}\Psi(T) &= \sum_{x\in V(T)} c_x(T) = \sum_{x\in V(A)} c_x(T) + \sum_{x\in V(B)} c_x(T) \\ &= \sum_{x\in V(A)} \Big(\sum_{y\in V(A)} d_T(x,y) + \sum_{y\in V(B)} d_T(x,y) \Big) \\ &\qquad + \sum_{x\in V(B)} \Big(\sum_{y\in V(B)} d_T(x,y) + \sum_{y\in V(A)} d_T(x,y) \Big) \ .\end{aligned}$$

For all $a \in A$ and $b \in B$, we have that the neighbors u and v lie on the shortest path between a and b. Hence, we have

$$\begin{aligned}\Psi(T) =& \sum_{x\in V(A)} \sum_{y\in V(A)} d_A(x,y) + \sum_{x\in V(A)} \sum_{y\in V(B)} (d_A(x,u) + d_T(u,y)) \\ &+ \sum_{x\in V(B)} \sum_{y\in V(B)} d_B(x,y) + \sum_{x\in V(B)} \sum_{y\in V(A)} (d_B(x,v) + d_T(v,y)) \\ =& \sum_{x\in V(A)} \sum_{y\in V(A)} d_A(x,y) + \sum_{x\in V(A)} \Big(|B| d_A(x,u) + \sum_{y\in V(B)} (1 + d_B(v,y)) \Big) \\ &+ \sum_{x\in V(B)} \sum_{y\in V(B)} d_B(x,y) + \sum_{x\in V(B)} \Big(|A| d_B(x,v) + \sum_{y\in V(A)} (1 + d_A(u,y)) \Big) \\ =& \Psi(A) + |B| c_u(A) + |A||B| + |A| c_v(B) \\ &+ \Psi(B) + |A| c_v(B) + |A||B| + |B| c_u(A) \\ =& \Psi(A) + \Psi(B) + 2\left(|A||B|\right) + 2|B| c_u(A) + 2|A| c_v(B) \ .\end{aligned}$$

In an analogous way, we get

$$\Psi(T') = \Psi(A) + \Psi(B) + 2(|A||B|) + 2|B| c_u(A) + 2|A| c_w(B) \ .$$

Thus, the amount of change in social cost of the edge-swap uv to uw is

$$\Delta(\Psi) = \Psi(T) - \Psi(T') = 2|A|(c_v(B) - c_w(B)) \ .$$

By Lemma 4.2.2, we have that $\Delta(u) = c_v(B) - c_w(B)$. Thus, $\Delta(\Psi) = 2|A|\Delta(u)$. ■

Now we are ready to prove Theorem 4.2.1.

PROOF (OF THEOREM 4.2.1) By Lemma 4.2.3, we have that the social cost strictly decreases if and only if the cost of the swapping agent strictly decreases. This implies that the social cost $\Psi(G)$ is an ordinal potential function for the SUM-SG on trees. ■

Theorem 4.2.1 guarantees that a Swap Equilibrium in the SUM-Swap Game can be found by distributed local search, even if the agents do not play optimally. We

only need one very natural ingredient for convergence: Whenever an agents moves, this move must decrease this agent's cost. We call every dynamic where an agent strictly improves by making a move (or passing if no improving move is possible) an *improving response dynamic* (IRD)[3]. Such a dynamic stops if no agent can strictly improve, which implies that any IRD stops if a swap-stable network is obtained. We will now analyze IRDs in more detail.

Improving Response Dynamics on Trees

For trees it was shown by Alon, Demaine, Hajiaghayi and Leighton [ADHL13] that the star is the only swap-stable tree. Using this observation and Theorem 4.2.1, we arrive at the following corollary.

Corollary 4.2.4 *For every tree T, every IRD converges to a star.*

Having guaranteed convergence, the natural question to ask is how many steps are needed to reach the unique Swap Equilibrium by myopic play. The following theorems provide a lower and an upper bound on that number.

Theorem 4.2.5 *Let P_n be a path having n vertices. Any IRD on P_n needs at least $\max\{0, n-3\}$ steps to converge.*

Proof Let $n \geq 4$, since otherwise P_n is already a star. Since the leaf-agents can perform an improving move (every swap to an inner vertex strictly decreases their cost), we have that P_n cannot be stable. By Corollary 4.2.4, any IRD converges to a star on n vertices. Clearly, such a star contains a vertex having degree $n-1$. In any step-wise transformation of P into a star, some vertex will become the center of the star. Assume that an inner vertex v of P is the designated center. Since v has degree 2 there are $n-3$ non-neighbors of v. Since in every step of the dynamic only one edge can be swapped, it follows that at least $n-3$ steps are needed, to connect all of these non-neighbors to v. If a leaf of P is the designated center, then one additional step is needed. ∎

Lemma 4.2.6 *P_n is the tree on n vertices which has maximum social cost.*

Proof Let T have at least four vertices and assume towards a contradiction that T has more than two leaves and has maximum social cost. Consider a leaf l of T, which has minimum cost $c_l(T)$ among all leaves of T. Let k be the neighbor of l in T and observe that $c_l(T) = c_k(T) + (n-2)$, since k is the first vertex on l's shortest paths to all other $n-2$ vertices of T. Let u be a leaf of T, which has maximum cost $c_u(T)$ among all leaves of T. Thus, we have $c_u(T) \geq c_l(T)$.

Now consider the tree $T' = T - l$. Since l is a neighbor of k in T, we have $c_k(T') = c_k(T) - 1 = c_l(T) - (n-1)$. Furthermore, we have $c_u(T') = c_u(T) - d_T(u, l)$.

[3]Such dynamics are also known as better response dynamics.

The tree T has at least three leaves, which implies that the longest path of T can have length at most $n-2$. Thus, $d_T(u,l) < n-1$, together with $c_u(T) \geq c_l(T)$, this implies $c_u(T') > c_k(T')$. Consider the edge-swap lk to lu by agent l and let T'' be the obtained tree.

We have

$$c_l(T'') = c_u(T'') + (n-2) = c_u(T') + (n-1) > c_k(T') + (n-1) = c_l(T) \ .$$

Thus, the edge-swap lk to lu strictly increases agent l's cost. By Lemma 4.2.3, it follows that the social cost of T'' is strictly larger than the social cost of T, which is a contradiction. Hence, every tree with maximum social cost must have exactly two leaves. ∎

Theorem 4.2.7 *Any IRD on trees having n vertices converges in $\mathcal{O}(n^3)$ steps. That is, the* Sum-*SG on trees is in poly-FIPG.*

Proof The idea is to start with the tree having the highest potential and to bound the number of steps any IRD needs by analyzing the number of steps needed if this potential is decreased by the smallest possible amount per step. By Lemma 4.2.6, we have that P_n has the maximum social cost $\Psi(P_n)$. Observe that

$$\Psi(P_n) = \sum_{i=1}^{n-1} 2i(n-i) = \frac{n^3-n}{3} \ .$$

Let X_n be a star having n vertices. We have $\Psi(X_n) = 2n^2 - 4n + 2$. To transform P_n into X_n any IRD has to decrease the social cost by

$$\Psi(P_n) - \Psi(X_n) = \frac{n^3}{3} - 2n^2 + \frac{11n}{3} - 2 \ .$$

Since we have an IRD, every moving agent decreases her cost by at least 1. By Lemma 4.2.3, we have that the minimum decrease in social cost by any move is 2. Hence, at most $\frac{n^3}{6} - n^2 + \frac{11n}{6} - 1 \in \mathcal{O}(n^3)$ steps are needed to transform P_n into X_n. ∎

Best Response Dynamics on Trees

It is reasonable to assume that agents greedily try to decrease their cost most, whenever swapping an edge. In this section we analyze dynamics, where every move of an agent is a best response move.

Since a best response is always an improving response, we have that every dynamic where every move is a best response must converge to a star for every tree T. We are left with the question of how fast best response dynamics converge. In the following, we analyze a specific best response dynamic, called *Max Cost Best Response Dynamic* (mcBRD), whose convergence speed almost matches the lower bound provided by Theorem 4.2.5. Hence, for best response dynamics we can significantly improve the upper bound of Theorem 4.2.7.

Definition 4.2.8 *The* Max Cost Best Response Dynamic *on a network G is a dynamic, where, starting with the initial network G, the moving agent is chosen by the max cost policy and moving agents always play best responses. We use the short-hand mcBRD(G).*

In this section we show the following upper bound on the speed of convergence for the Max Cost Best Response Dynamic. Surprisingly, mcBRD behaves differently depending on whether the number of agents in the tree is odd or even.

Theorem 4.2.9 *Let T be a tree having n vertices. The following holds:*

- *If n is even, then mcBRD(T) converges after at most $\max\{0, n-3\}$ steps and every agent moves at most once.*
- *If n is odd, then at most $\max\{0, n+\lfloor n/2 \rfloor - 5\}$ steps are needed and every agent moves at most twice.*

In order to prove Theorem 4.2.9, we first show some useful properties of the convergence process induced by the mcBRD-rule.

We begin with characterizing an agent's best response on a tree. Here, the notion of a 1-*median vertex*, as defined in Definition 3.3.4, is crucial.

Lemma 4.2.10 *Let u be an arbitrary vertex of a tree T and let $F = T - u = \bigcup_{j=1}^{l} T_l$, where the trees T_j are connected components in the forest F. Let $v_1, \dots, v_l$ be the neighbors of u in T, where v_j is a vertex of T_j for all $1 \leq j \leq l$. Let w_j be a 1-median vertex of the tree T_j. The best response of u in T is the edge-swap uv_j to uw_j, where*

$$j \in \arg\max_{1 \leq j \leq l} \left\{ c_{v_j}(T_j) - c_{w_j}(T_j) \right\} \ .$$

Proof Let T' be the tree obtained after agent u's swap. Observe that if agent u removes the edge uv_i for some $i \in \{1, \dots, l\}$, then it must be replaced with an edge ux_i, where $x_i \in V(T_i)$, since otherwise T' would be disconnected. Thus, if the edge uv_i is removed, then agent u has to choose which of the vertices of T_i to connect to. By Lemma 4.2.2, agent u's change in cost is $\Delta(u) = c_{v_i}(T_i) - c_{x_i}(T_i)$, if the edge uv_i is removed and ux_i is built. Since agent u's best response yields the largest decrease in cost, it follows that x_i must be chosen such that $c_{x_i}(T_i) \leq c_{y_i}(T_i)$ holds for all vertices $y_i \in V(T_i)$. Thus, x_i must be a 1-median vertex of T_i. Agent u can swap only one edge. Hence, u's best response is to connect to a 1-median vertex x_j of a tree T_j, which maximizes $c_{v_j}(T_j) - c_{x_j}(T_j)$. ∎

The next Lemma provides a very useful property of neighbors in a tree.

Lemma 4.2.11 *Let u and w be neighbors in a tree T. Let T_u and T_w denote the tree T rooted at vertex u and w, respectively. Let U be the set of vertices in the subtree rooted at u in T_w. Analogously, let W be the set of vertices in the subtree rooted at w in T_u. Then we have*

$$c_u(T) \leq c_w(T) \iff |U| \geq |W| \quad \text{and} \quad c_u(T) < c_w(T) \iff |U| > |W| \ .$$

PROOF Let u, w, U, W, T_u and T_w be defined as in the Lemma. Since T is a tree, we have

$$\begin{aligned} & c_u(T) \leq c_w(T) \\ \iff & \sum_{x \in U} d_T(u,x) + \sum_{x \in W} (1 + d_T(w,x)) \leq \sum_{x \in W} d_T(w,x) + \sum_{x \in U} (1 + d_T(u,x)) \\ \iff & \sum_{x \in W} (1 + d_T(w,x)) - \sum_{x \in W} d_T(w,x) \leq \sum_{x \in U} (1 + d_T(u,x)) - \sum_{x \in U} d_T(u,x) \\ \iff & |W| \leq |U| \ . \end{aligned}$$

If the inequality is strict, then the proof is similar. ∎

We can use Lemma 4.2.11, to show an important property of the mcBRD-process.

Lemma 4.2.12 *Let T be a tree. Every agent who moves in any step of mcBRD(T) must be a leaf-agent.*

PROOF In every step of mcBRD the agent with the highest cost is allowed to move. Assume towards a contradiction that an inner vertex u has the highest cost c^* in T. Let $x_1, \ldots, x_l$ be the neighbors of u. By Lemma 4.2.11, we have that at most one of the neighbors of u can have the same cost c^*.

If u has no neighbor having cost c^*, then all neighbors must have strictly lower cost than agent u. But Lemma 4.2.11 yields that at most one neighbor of any vertex can have lower cost. Since u has at least two neighbors, there must be a neighbor of u having higher cost and we have a contradiction.

If u has a neighbor w having cost c^*, then, by Lemma 4.2.11, all other neighbors of w must have lower cost. If there is more than one such neighbor, then again, we have a contradiction. Thus, assume that there is exactly one such neighbor z. Let T_u, T_w and T_z denote the tree T rooted at vertex u, w and z, respectively. Let U and W_1 denote the subtree rooted at u and w, respectively, in tree T_z. Let W_2 denote the tree rooted at w in tree T_u and let Z denote the subtree rooted at z in tree T_w. By Lemma 4.2.11, we have that $|Z| > |W_1| \geq |U|$. Furthermore, we have $|W_2| > |Z|$. Hence, we have $|W_2| > |U|$ and thus, again by Lemma 4.2.11, it follows that $c_u(T) > c_w(T)$, which is a contradiction. ∎

The following Lemma provides the key to analyzing mcBRD. It shows that at some point in the dynamic a certain behavior is "triggered" which forces the dynamic to converge quickly.

Lemma 4.2.13 (First Trigger Lemma) *Let T be a tree. If the agent who moves in step i of mcBRD(T) has a unique best response vertex w, then all agents who move in a later step of mcBRD(T) will connect to vertex w.*

PROOF Let T be any tree. Let T^s denote the tree which is obtained after step s of mcBRD(T) and let u^s denote the agent who moves in step s. Consider step i of

mcBRD(T) and assume that agent u^i has maximum cost in T^{i-1}. Let the edge-swap towards w be the unique best response of agent u^i in this step. We show for any step $j \geq i+1$ of mcBRD(T) that if T^{j-1} is not a star, then agent u^j will connect to vertex w, if agent u^{j-1} did.

Consider the tree T^{j-2} in which agent u^{j-1} has maximum cost. It follows by Lemma 4.2.12 that u^{j-1} must be a leaf of T^{j-2}. Since T^{j-1} is not a star, we have that T^{j-2} is not a star. Assume that the unique best response of agent u^{j-1} is to connect to vertex w. By Lemma 4.2.10, it follows that w must be the unique 1-median vertex of the tree $T'' = T^{j-2} - u^{j-1}$. Let $x_1, \ldots, x_k$ be the neighbors of w in T''. Let T_w be the tree T'' rooted at vertex w and let $X_1, \ldots, X_k$ be the subtrees of T_w rooted at $x_1, \ldots, x_k$, respectively. Using the fact that w is the unique 1-median vertex of T'' and Lemma 4.2.11, we obtain that $1 + \sum_{p \neq q} |X_p| > |X_q|$ for any $q \in \{1, \ldots, k\}$. After her move, agent u^{j-1} will end up as the $k+1$'th neighbor of w in the tree T^{j-1}. Since, by assumption, this tree is not swap-stable, there is a leaf u^j of T^{j-1} who swaps an edge in step j. Clearly, we have $u^j \in V(X_r)$ for some $r \in \{1, \ldots, k\}$. Now consider $T''' = T^{j-1} - u^j$ and let $X'_1, \ldots, X'_{k+1}$ be defined analogously as above for the tree T'''. We have that $|X_i| = |X'_i|$, for all $i \in \{1, \ldots, k\} \setminus \{r\}$, and $|X_r| = |X'_r| + 1$. The new tree X_{k+1} contains only vertex u^{j-1} and thus compensates the loss of tree $|X'_r|$. Hence, we have $1 + \sum_{p \neq q} |X_p| > |X_q|$, for $q \in \{1, \ldots, k+1\}$. By Lemma 4.2.11, this implies that w is the unique 1-median vertex of T'''. Thus, agent u^j will connect to w in step j. ∎

Lemma 4.2.14 *In any tree T on n vertices, there are at most two 1-median vertices. If this is the case, then they are neighbors and n must be even.*

Proof Assume that T contains exactly two vertices x_1 and x_2, which both have minimum cost c^* but there is no edge x_1x_2 in T. Since T is connected, there is a path P from x_1 to x_2 of length at least 2. Let z_1 and z_2 be the neighbors of x_1 and x_2, respectively, on path P. Let $T_{x_1}, T_{x_2}, T_{z_1}$ and T_{z_2} be the tree T rooted at vertex x_1, x_2, z_1 and z_2, respectively. Let X_1 be the set of vertices in the subtree of T_{z_1}, which is rooted at vertex x_1. Analogously, X_2 denotes the set of vertices in the subtree rooted at x_2 in T_{z_2}. We define Z_1 and Z_2 in the same way, for the trees T_{x_1} and T_{x_2}, respectively. We apply Lemma 4.2.11, which yields

$$c_{x_1}(T) \leq c_{z_1}(T) \iff |X_1| \geq |Z_1| \quad \text{and} \quad c_{x_2}(T) \leq c_{z_2}(T) \iff |X_2| \geq |Z_2| \ .$$

Since T is a tree and x_1 and x_2 are non-neighbors, we have that $|Z_1| > |Y|$ and $|Z_2| > |X|$. Using Lemma 4.2.11, this implies $c_{z_1}(T) < c_y(T)$ and $c_{z_2}(T) < c_x(T)$, which is a contradiction. The only feasible solution is that $z_1 = x_2$ and $z_2 = x_1$ and thus, x_1 and x_2 have to be neighbors. Furthermore, we have that $c_{x_1}(T) = c_{x_2}(T) = c^*$. By Lemma 4.2.11, it follows that $|X_1| = |X_2|$ which implies that n must be even.

If there are more than two agents having minimum cost, then the above argumentation implies that all of them must be pairwise neighbors. Since T is a tree, this is impossible. ∎

Now we are ready, to prove the first part of Theorem 4.2.9.

Proof (of Theorem 4.2.9, Part 1) We show that if the number of vertices in a tree T is even, then mcBRD needs at most $\max\{0, n-3\}$ to converge and every agent moves at most once.

If T has two vertices, then it is a star and no agent will move in mcBRD(T). Thus, let T be a tree having at least $n \geq 4$ vertices, where n is even. By Lemma 4.2.12, we have that in every step of mcBRD(T) a leaf l of the current tree is allowed to move. By Lemma 4.2.10, we know that agent l will connect to a 1-median vertex of $T' - l$, where T' is the tree before agent l moves. Observe that the tree $T' - l$ has an odd number of vertices. By Lemma 4.2.14, we have that any tree having an odd number of vertices must have a unique 1-median vertex. It follows that the leaf who moves in the first step of mcBRD(T) has a unique best response. Let this best response be the edge-swap towards vertex w. Lemma 4.2.13 implies that all agents who move in a later step of mcBRD(T) will connect to vertex w as well. Furthermore, again by Lemma 4.2.13, after the first step of mcBRD(T) it holds that every vertex who is already connected to vertex w will never move again. Hence, every agent moves at most once.

By Lemma 4.2.11, we have that w must be an inner vertex of T. Thus, w has at most $n-3$ non-neighbors which implies that the dynamic mcBRD(T) will need at most $n-3$ steps to converge to a star having w as its 1-median vertex. ∎

The next theorem shows a lower bound on the speed of convergence for mcBRD on trees having an odd number of vertices. Surprisingly, the behavior of the dynamic on such instances is much more complex. The lower bound for odd n is roughly 50% greater than the upper bound for even n. Furthermore, the following theorem together with Theorem 4.2.5 implies that the analysis of mcBRD is tight.

Theorem 4.2.15 *There is a family of trees having an odd number of vertices greater than* 5*, where mcBRD can take* $n + \lfloor n/2 \rfloor - 5$ *steps to converge. Furthermore, every agent moves at most twice.*

Figure 4.2 shows an example of a tree which belongs to the above mentioned family of trees and it sketches the convergence process induced by mcBRD.

Proof (of Theorem 4.2.15) A member of the family is constructed as follows: We start with a path having 5 vertices. Let the leaves of this path be l and r, let the center be w and let v be the vertex between w and r and u be the vertex between l and w. Fix an even number $k \geq 2$ and connect $k/2$ vertices to l and r, respectively. Let $x_1, \ldots, x_{k/2}$ be the vertices having l as neighbor and $y_1, \ldots, y_{k/2}$ are the vertices connected to r. An example is shown top left in Fig. 4.2.

Let T be a tree constructed in the described way. During mcBRD(T) some agents will have two best responses and the number of steps towards convergence depends on which best response is chosen. Note that this implies that a local decision has

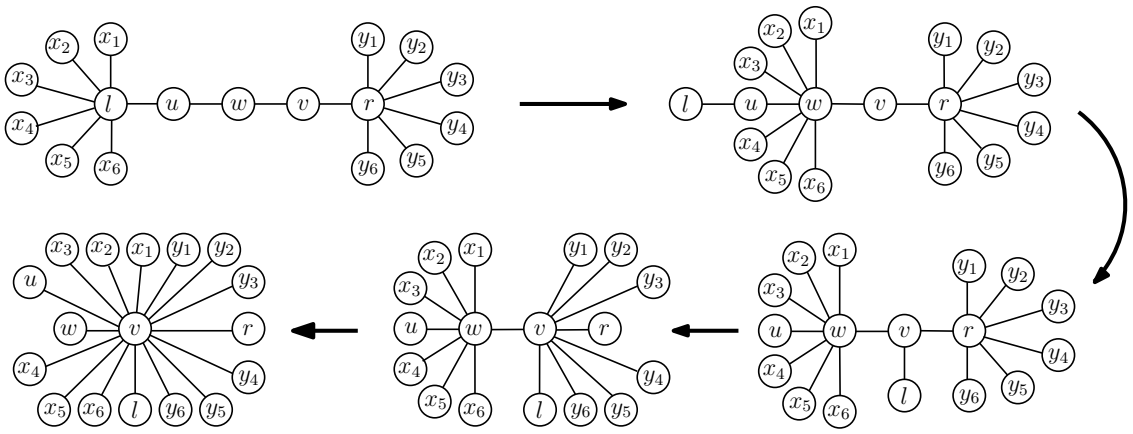


Figure 4.2.: Example of a tree network T having 17 vertices, where mcBRD(T) takes $n + \lfloor n/2 \rfloor - 5 = 20$ steps to converge. The agents $x_1, \ldots, x_6, u$ move twice.

global impact. We show that these choices can be made such that mcBRD(T) takes $n + \lfloor n/2 \rfloor - 5$ steps until a star emerges.

Since T is symmetric, all leaves are equal in the first step of mcBRD(T), that is, they all have the same cost. By Lemma 4.2.12, one of those leaves moves in the first step of mcBRD(T). Let T^i denote the tree, which is obtained after the i-th step of mcBRD(T). Let T^i_w and T^i_v denote the tree T^i rooted at w and v, respectively. Let W^i be the set of vertices in the tree rooted at w in T^i_v and let V^i be the set of vertices in the tree rooted at v in T^i_w. Before the first step of mcBRD(T), we have that $|W^0| = |V^0| + 1$.

The convergence proceeds in three stages:

Stage 1: Without loss of generality, assume that agent x_1 moves first. Consider the tree $T^0 - x_1$. This tree has two 1-median-vertices, namely w and v. Hence, agent x_1 has two best responses. Assume that x_1 chooses to connect to vertex w. Thus, we have that $W^0 = W^1$ and $V^0 = V^1$. In the next step, we have that vertices $x_2, \ldots, x_{k/2}$ have maximum cost. Each of those agents has two best responses, namely to connect to w or v. This is true by Lemma 4.2.11, since $|W^1| - 1 = |V^1|$. Let x_2 move towards vertex w, which implies $W^1 = W^2$ and $V^1 = V^2$. This process iterates until all x_i-vertices are connected to w. Thus, we have that $W^0 = W^{k/2}$ and $V^0 = V^{k/2}$. The top right network in Fig. 4.2 illustrates the result of the steps mentioned so far.

In the following step, agent l is allowed to move and, again, there are the two best responses w and v. Let l choose the connection towards v. This implies $W^{k/2+1} = W^0 \setminus \{l\}$ and $V^{k/2+1} = V^0 \cup \{l\}$. Observe that $|W^{k/2+1}| < |V^{k/2+1}|$ holds. This leads to the bottom right network in Fig. 4.2.

Stage 2: Now, all y_i-vertices have maximum cost. Again, they have the two best responses w and v and we let them choose the vertex which is closer, that

is v. After another $k/2$ steps a network similar to the one in the middle of the bottom row in Fig. 4.2 is obtained. Furthermore, we have that $W^{k+1} = W^{k/2+1}$ and $V^{k+1} = V^{k/2+1}$. Let N_v^V denote the number of neighbors of v in V^{k+1}. Let N_w^W be defined analogously. Observe that $N_v^V > N_w^W$ holds.

Stage 3: In the next steps, all neighbors of w in W have maximum cost, but this time there is only one best response, which is the connection to vertex v. The dynamic stops when all vertices are connected to v, which happens after N_w^W many steps.

Now we analyze the number steps of mcBRD(T). In Stage 1, there are $k/2$ steps, where an agent x_i moves and one step where l swaps an edge. In Stage 2, all y_i-agents move, which implies $k/2$ steps. In Stage 3, there are N_w^W steps. Since $|W^{k+1}| = |W^0| - 1$, we have that $N_w^W = |W_0| - 2 = \lceil n/2 \rceil - 2$. Observe that in Stage 3 all x_i-agents move a second time. There are no other agents which move twice.

Hence, in total there are $k/2 + 1 + k/2 + \lceil n/2 \rceil - 2 = k + \lfloor n/2 \rfloor$ steps. By construction, we have that $k = n-5$ and thus mcBRD(T) takes $n + \lfloor n/2 \rfloor - 5$ steps to converge and every agent moves at most twice. ∎

For proving the second part of Theorem 4.2.9 we need two additional properties of the mcBRD-process.

Lemma 4.2.16 (Second Trigger Lemma) *Let T be an unstable tree having n vertices. If after any step i in mcBRD(T) a vertex w of T^i has degree $\lceil n/2 \rceil$, then this vertex will be the unique best response to connect to for all agents moving in a later step of mcBRD(T).*

Proof Let T be any unstable tree having n vertices. Observe that whenever an agent moves in mcBRD(T), the degrees of exactly two vertices change by 1. Let step i be the first step of mcBRD(T) after which a vertex w having degree $\lceil n/2 \rceil$ occurs. Using Lemma 4.2.13, it suffices to show that if vertex w has degree $\lceil n/2 \rceil$ after step i of mcBRD(T), then the agent u^{i+1} who moves in the $i+1$'th step will have w as its unique best response vertex. To prove this, we show that u^{i+1} cannot be a neighbor of w in T^i. By Lemma 4.2.11, this implies that w is the unique best response vertex of u^{i+1}.

Assume towards a contradiction that agent x is a neighbor of w in T^i and that x moves in the $i+1$'th step of mcBRD(T). Thus, agent x must be a leaf and can swap an edge to decrease her cost. This implies that w is not a vertex having minimum cost in $T' = T^i - x$. Observe that vertex w has degree $\lfloor n/2 \rfloor$ in T'. Let v be a vertex having minimum cost in T'. Let T'_w denote the tree T' rooted at w. We have that v lies in a subtree rooted at some neighbor y of w in T'_w. Let T'_y denote the tree T' rooted at y. Let W be the set of vertices in the subtree rooted at w in T'_y and let Y denote the set of vertices in the subtree rooted at y in T'_w. Since w has degree $\lfloor n/2 \rfloor$, we have that $|W| \geq \lfloor n/2 \rfloor$. Thus, $|Y| \leq |T'| - |W| = \lfloor n/2 \rfloor$.

By Lemma 4.2.11, it follows that w cannot have higher cost than v and we have a contradiction. ∎

Lemma 4.2.17 *Let T be an unstable tree having an odd number of vertices. Only vertices which are best response vertices of the agent who moves in the first step of mcBRD(T) will be best responses in any step of mcBRD(T).*

Proof Let T be an unstable tree having n vertices, where n is odd. Let u be the agent who moves in the first step of mcBRD(T). Let T^i denote the resulting tree after step i of mcBRD(T).

If agent u has a unique best response w, then, by Lemma 4.2.13, vertex w will be the unique best response of any agent who moves in a later step of mcBRD(T).

The only case left is the one where agent u has two best responses v and w. We show that the set of best responses of any agent y who moves in step $j \geq 2$ is a subset of the set of best responses of the agent x who moved in step $j-1$. Settling this implies the Lemma.

If in step $j-1$ agent x has a unique best response, then, by Lemma 4.2.13, the claim is true. Thus, suppose that in step $j-1$ agent x has two best responses p and q. Let T^{j-2} be the tree T after step $j-2$. Let T_p^{j-2} and T_q^{j-2} be the tree T^{j-2} rooted at p and q, respectively. Let P^{j-2} be the subtree rooted at p in T_q^{j-2} and let Q^{j-2} be the subtree rooted at q in T_p^{j-2}. Suppose that x is a leaf of P^{j-2}. Since both p and q are best responses for x we have, by Lemma 4.2.11, that $|P^{j-2} - x| = |Q^{j-2}|$. In step $j-1$ agent x will connect either to p or to q. Suppose x connects to p. Now consider the moving agent y in step j. Define T^{j-1}, P^{j-1} and Q^{j-1} analogous to the respective trees after step $j-2$. There are two cases:

If y is a leaf of P^{j-1}, then $|P^{j-1} - y| = |Q^{j-1}|$ and thus, by Lemma 4.2.10 and Lemma 4.2.11, agent y has p and q as its best responses.

If y is a leaf of Q^{j-1}, then we claim that vertex p is the unique vertex having minimum cost in $T^{j-1} - y$ and thus vertex p must be agent y's unique best response. To prove the claim, it suffices to show that every neighbor of p in $T^{j-1} - y$ has higher cost than p itself. Since $|P^{j-1}| > |Q^{j-1} - y|$, this holds by Lemma 4.2.11 for vertex q. Furthermore it is trivially true for x, which is a leaf connected to p. Let z be any other neighbor of p in $T^{j-1} - y$. In tree $T^{j-2} - x$, by assumption, p and q are the vertices having minimum cost. This implies that agent z must have higher cost than agent p in $T^{j-2} - x$. Since agent x, who is missing in $T^{j-2} - x$ connects in step $j-1$ to vertex p, this difference increases further, which settles the claim.

The case where x connects in step $j-1$ to vertex q and both subcases where x is a leaf of Q^{j-2} are analogous. ∎

Finally, we have set the stage to prove the second part of Theorem 4.2.9.

Proof (of Theorem 4.2.9, Part 2) We show that if a tree T has an odd number of vertices, then mcBRD(T) takes at most $\max\{0, n + \lfloor n/2 \rfloor - 5\}$ steps to converge and every agent moves at most twice.

If $n = 5$, then the worst case instance is a path and thus the convergence takes at most 2 steps. Hence, we assume for the following that $n \geq 7$. Observe that there are two events that force the dynamic to converge: Let E_1 be the event, where for the first time in the convergence process a vertex w becomes the unique best response of a moving agent. Let E_2 be the event, where for the first time a vertex w has degree $\lceil n/2 \rceil$.

If event E_1 occurs in step j, then, by Lemma 4.2.13, all non-neighbors of the vertex w will connect to w in the subsequent steps of mcBRD(T). Thus, mcBRD(T) will converge in at most $j + |V \setminus \Gamma(w)|$ steps, where $\Gamma(w)$ is the closed neighborhood of w. If event E_2 occurs in step j, then, by Lemma 4.2.16, all non-neighbors of w will connect to w in the subsequent steps. Thus, in this case $j + \lfloor n/2 \rfloor - 1$ steps are needed for mcBRD(T) to converge.

Let T be any tree and u be the first agent to move and assume that u has two best responses p and q, since otherwise the dynamic will converge in at most $n-3$ steps. By Lemma 4.2.17, we have that in any step of mcBRD(T) an agent will connect either to p or to q. Let $t_1(T)$ denote the number of steps until event E_1 is the first event to occur in mcBRD(T). Analogously, let $t_2(T)$ denote the number of steps until E_2 is the first occurring event. Let $r_1(T)$ denote the number of steps needed for convergence after event E_1. Hence, the maximum number of steps needed until mcBRD(T) converges is $t(T) = \max\{t_1(T) + r_1(T), t_2(T) + \lfloor n/2 \rfloor - 1\}$.

We claim that $t_1(T) + r_1(T) \leq n + \lfloor n/2 \rfloor - 5$. Observe that $r_1(T) \leq n - 3$, since the vertex that becomes the center of the star must be an inner vertex of T and, thus, can have at most $n-3$ non-neighbors. Furthermore, if $t_1(T) \leq \lfloor n/2 \rfloor - 2$, then the claim is true. Now let $t_1(T) > \lfloor n/2 \rfloor - 2$. Note that both p and q must be inner vertices of T. Thus, they have at least degree 2. Since event E_2 did not occur in the first $t_1(T)$ steps of mcBRD(T) we have that not all agents who moved within the first $t_1(T)$ steps can be connected to p. Thus, at least $x = t_1(T) - (\lfloor n/2 \rfloor - 2)$ agents have connected to q. This yields $t_1(T) + r_1(T) \leq t_1(T) + n - 3 - x \leq n + \lfloor n/2 \rfloor - 5$. On the other hand, since all agents move either to p or q and both p and q have degree at least 2, it follows that $t_2(T) \leq 2(\lfloor n/2 \rfloor - 2)$. Hence, $t_2(T) + \lfloor n/2 \rfloor - 1 \leq n + \lfloor n/2 \rfloor - 5$.

Observe that any agent x who is a neighbor of either p or q will not move again until event E_1 or E_2 happens. This holds because every leaf, which is not a neighbor of p or q must have higher cost than x and will therefore move before x. Thus, every agent moves at most twice. ∎

Computing a Best Response on Trees

Observe that Lemma 4.2.10 directly yields an algorithm for computing a best response move of an agent u: Compute the costs of all other agents in $T - u$ within their respective connected component to find a 1-median vertex for every component. Then choose the 1-median vertex, which yields the greatest cost decrease for agent u. Clearly, the cost of an agent can be obtained using a BFS-computation. However the above naive approach of computing a 1-median vertex yields an algo-

rithm with running time quadratic in n, since $\Omega(n)$ BFS-computations can occur. The following lemma shows that a 1-median vertex can be computed in linear time, which is clearly optimal. The algorithm crucially uses the structural property provided by Lemma 4.2.11. Note that we are not the first to observe this[4]. Our algorithm can be seen as an alternative solution for this problem.

Lemma 4.2.18 *Let T be a tree having n vertices. A 1-median vertex of T and its cost can be computed in $\mathcal{O}(n)$ time.*

Proof We give a linear time algorithm, which computes a 1-median vertex of T and its cost. Let L be the set of leaves of T. Clearly, L can be computed in $\mathcal{O}(n)$ steps by inspecting every vertex[5].

Given T and L, the algorithm proceeds in two stages:

1. The algorithm computes for every vertex v of T two values n_v and c_v. This is done in reverse BFS-order: We define n_v to be the number of vertices in the already processed subtree T_v containing v and c_v to v's cost in T_v. For every leaf $l \in L$ we set $n_l := 1$ and $c_l := 0$. Let i be an inner vertex and assume that we have already processed all but one neighbor of i. Let $a_1, \dots, a_s$ denote these neighbors. We set $n_i := 1 + n_{a_1} + \dots + n_{a_s}$ and $c_i := n_i - 1 + c_{a_1} + \dots + c_{a_s}$. By breaking ties arbitrarily, this computation terminates at a root-vertex r, for which all neighbors are already processed. Let $b_1, \dots, b_q$ denote these neighbors. We set $n_r := n$ and $c_r := n - 1 + c_{b_1} + \dots + c_{b_q}$.

2. Starting from vertex r, we perform a local search for a 1-median vertex with the help of Lemma 4.2.11. For all neighbors $b_i \in \{b_1, \dots, b_q\}$ of r, we check if $n_{b_i} > n_r - n_{b_i}$. Since T is a tree, this can hold for at most one neighbor x. In this case, x will be considered as new root-vertex. Let $c_1, \dots, c_s, r$ be the neighbors of x. By setting $n_x := n$ and $c_x := n-1+c_1+\dots+c_s+c_r-c_x$ we arrive at the same situation as before and we now check for all neighbors $c_j \neq r$ if $n_{c_j} > n_x - n_{c_j}$ holds and proceed as above. Once no neighbor of the current root-vertex satisfies the above condition, the algorithm terminates and the current root-vertex is the desired 1-median vertex.

The correctness of the above algorithm follows by Lemma 4.2.11. Step 1 clearly takes time $\mathcal{O}(n)$. Step 2 takes linear time as well, since the condition is checked exactly once for every edge towards a neighbor and there are only $n - 1$ edges in the tree T. ∎

Theorem 4.2.19 *If $p \geq 1$ edges can be swapped at a time, then the best response of an agent u in network G can be computed in linear time if G is a tree.*

[4] To the best of our knowledge, the first linear time algorithm for computing a 1-median vertex on a (weighted) tree is due to Goldman [Gol71]. See also a more general exposition of this simple algorithm in Kariv and Hakimi [KH79b].

[5] We assume that the tree T is given as adjacency list.

PROOF Let u be a degree d vertex in the tree network G. Clearly, agent u can swap at most $\min\{p, d\}$ edges and the task is to determine the $k \leq \min\{p, d\}$ edge-swaps that decrease agent u's cost most. Let $v_1, \ldots, v_d$ denote the neighbors of u in G.

Let $F = T_1 \cup T_2 \cup \cdots \cup T_d$ be the forest obtained by deleting vertex u from G. Let $c_i = |T_i| + \sum_{w \in T_i} d_G(v_i, w)$ denote agent u's connection cost to vertices in T_i, where $1 \leq i \leq d$. By Lemma 4.2.10 we have that every swap in agent u's best response is of the form (v_i, w_i), which abbreviates the swap from uv_i to uw_i, where w_i is a 1-median vertex of T_i. Let $z_i = c_{w_i}(T_i) - c_{v_i}(T_i)$ denote agent u's change in cost after the swap (v_i, w_i). Clearly, if $z_i \geq 0$ then the swap (v_i, w_i) will not be part of u's best response, since it does not yield a cost reduction. If $z_i < 0$, then we call the corresponding swap (v_i, w_i) *attractive*. If there are l attractive swaps for agent u, then we have that u's best response will consist of the $\min\{k, l\}$ attractive swaps having the smallest z_i values.

Thus, computing agent u's best response reduces to finding a 1-median vertex in each tree T_i and to computing the corresponding value of $c_{w_i}(T_i)$. By Lemma 4.2.18 we have that both tasks can be done in time linear in the number of vertices in each T_i. Observe that all negative z_i-values are in the range $[-n^2, 0]$. Hence, we can use radixsort [LRSC01] to find the $\min\{k, l\}$ attractive swaps having the smallest z_i values in linear time. ∎

4.2.2. Playing on General Networks

Now we consider the case where the initial network G_0 is an arbitrary connected network.

Best Response Dynamics on General Networks

We show that there is no convergence guarantee for the SUM-SG on general initial networks.

Theorem 4.2.20 *The* SUM*-SG on general networks admits a best response cycle.*

PROOF Consider the network G_1 depicted left in Figure 4.3. Agent a can decrease her cost and one of her best responses is to swap edge ab with edge ac. This leads to network G_2 depicted in Figure 4.3. In G_2, agent b has the swap bc to ba as its best response, which leads to network G_3 depicted in the illustration. Finally, in G_3 agent c can perform the swap ca to cb as its best response, which leads back to network G_1. ∎

Voorneveld [Voo00] introduced the class of *best-response potential games*, which is a super-class of ordinal potential games. Furthermore he proves that if the strategy space is countable, then a strategic game is a best-response potential game if and only if there is no best response cycle. This implies the following Corollary.

Corollary 4.2.21 *There cannot exist an ordinal potential function for the* SUM*-SG on general networks.*

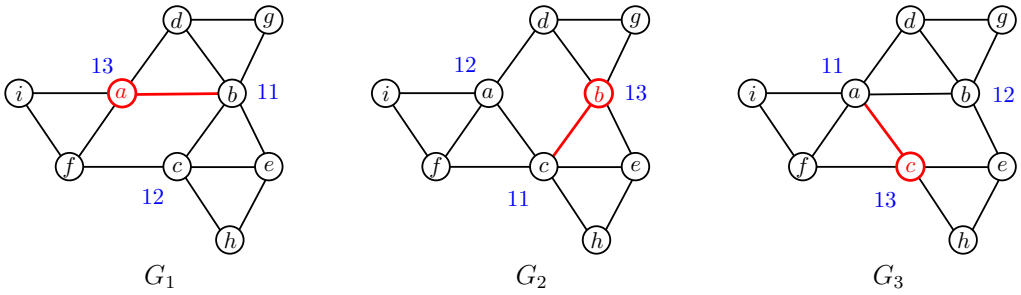


Figure 4.3.: A best response cycle for the SUM-SG. The moving agent is colored red, the agents' costs are drawn in blue.

Computing a Best Response in General Networks

Given an undirected connected network G, then the best response for agent v can be computed in $\mathcal{O}(n^2)$ time, since $|S_G(v)| < n^2$ and we can try all pure strategies to find the best one. In contrast to the result on trees, we show that computing a best response is hard if we allow an agent to swap multiple edges at a time.

Theorem 4.2.22 *If agents are allowed to swap multiple edges at a time, then computing the best response in a network G is NP-hard, even if G is planar and has maximum degree* 3.

PROOF We reduce from the p-MEDIAN problem [KH79b], which is defined as follows: Given a connected undirected network $G = (V, E)$ with non-negative weights $w(v)$ for every vertex $v \in V$ and non-negative lengths $l(e)$ for every edge $e \in E$ and given an integer $p > 1$. The task is to find a subset $X \subseteq V$ with $|X| = p$ such that $\sum_{v \in V} \min_{u \in X} w(v) d_G(v, u)$ is minimized. Here $d_G(v, u)$ denotes the length of the shortest path from v to u in G.

The p-MEDIAN problem is known [KH79b] to be NP-hard for $p > 1$ even if all vertex weights and edge lengths are one, G is planar and has maximum degree 3.

The reduction works as follows: Let G be an instance of the p-MEDIAN problem, where G is planar, has unit vertex weights and edge lengths and has maximum degree 3. We introduce a new agent v^* and connect v^* with p new edges to G which induces the graph $G' = (V', E')$. Now, let agent v^* play a best response and let $S_{v^*} \subseteq V \setminus \{v^*\}$ be the corresponding strategy of v^*, that is, S_{v^*} is the set of vertices incident to v^* after the best response move. Let $G^* = (V', E^*)$ be the network obtained after the best response move of v^*. We claim that S_{v^*} is the solution to the p-MEDIAN problem in G, which implies NP-hardness of computing the best response if p edges can be swapped at a time.

Clearly, we have $|S_{v^*}| = p$, since no best response of v^* will allow multiple edges connecting to the same vertex. By definition of a best response, we have that building edges to vertices in S_{v^*} minimizes the cost $c_{v^*}(G^*)$ of agent v^* in the

network G^*. Thus, we have

$$c_{v^*}(G^*) = \sum_{u \in V'} d_{G^*}(v^*, u) = \sum_{u \in V} \left(1 + \min_{x \in S_{v^*}} d_G(x, u)\right) = |V| + \sum_{u \in V} \min_{x \in S_{v^*}} d_G(x, u) \ ,$$

which yields that the cost of agent v^* is minimized if and only if the set S_{v^*} minimizes $\sum_{u \in V} \min_{x \in S_{v^*}} d_G(x, u)$. ∎

Remark 4.2.23 *Note that the above result trivially carries over to the* Sum-*ASG where agents are allowed to perform multi-swaps. Using a similar proof and reducing from the p-*Center *problem [KH79a] it was shown in [EFM+11] that computing a best response for the bounded budget version of the* Max-*BG in general networks is NP-hard as well. Since agents always use up their budget this implies that computing a best response in the* Max-*(A)SG is NP-hard on general networks if agents are allowed to swap multiple edges at a time.*

4.3. Dynamics in Max Swap Games

In this section we focus on the game dynamics of the Max-SG. Recall that the cost of an agent u in a connected network G in the Max-SG is defined as $c_u(G) = \max_{w \in V(G)} d_G(u, w)$. Interestingly, we obtain results which are very similar to the results shown in Section 4.2 but we need entirely different techniques to derive them.

We emphasize the contrast between both versions with an example which shows that the social cost cannot be used as ordinal potential function in the Max-SG on trees. Consider an improving swap of agent u from v to w in the tree network in Fig. 4.4 (left). This swap decreases agent u's cost from 4 to 3 but it increases the

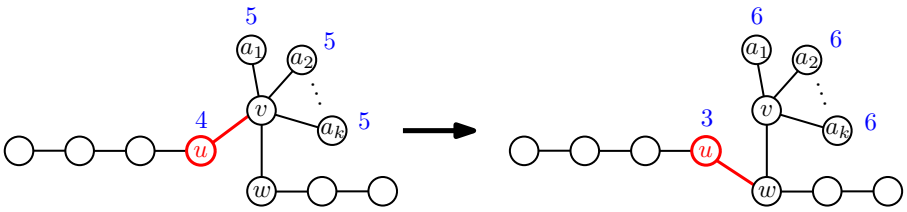


Figure 4.4.: Example of an improving swap in the Max-SG which increases the social cost.

the cost of all agents $a_1, \ldots, a_k$ by 1. Thus, if k is large enough, then the improving swap of agent u increases the social cost of the network.

4.3.1. Dynamics on Trees

We will analyze the network creation process in the Max-SG when the initial network is a tree. We prove that this process has the following desirable property:

Theorem 4.3.1 *The* MAX-*SG on trees is guaranteed to converge in* $\mathcal{O}(n^3)$ *steps to a stable network. That is, the* MAX-*SG on trees is a poly-FIPG.*

Before proving Theorem 4.3.1, we analyze the impact of a single edge-swap. Let $T = (V, E)$ be a tree on n vertices and let agent u be unhappy in network T. Assume that agent u can decrease her cost by performing the edge-swap uv to uw, for some $v, w \in V$. This swap transforms T into the new network $T' = (V, (E \setminus \{uv\}) \cup \{uw\})$. Let A denote the maximal tree of $T'' = (V, E \setminus \{uv\})$ which contains u and let B be the tree of T'' which contains v and w. It is easy to see that we have $d_T(x, y) = d_{T'}(x, y)$, if $x, y \in V(A)$ or if $x, y \in V(B)$.

Lemma 4.3.2 *For all* $x \in V(A)$ *there is no* $y \in V(A)$ *such that* $c_x(T) = d_T(x, y)$.

PROOF By assumption, agent u can decrease her cost by swapping the edge uv to edge uw, where $v, w \in V(B)$. We have that $d_T(x, u) < c_u(T)$, for all $x \in V(A)$, since otherwise this swap would not change agent u's cost. It follows that for arbitrary $x, y \in V(A)$ we have $d_T(x, y) \leq d_T(x, u) + d_T(u, y) < d_T(x, u) + c_u(T)$. Let $z \in V(B)$ be a vertex having maximum distance to u in T, that is, $c_u(T) = d_T(u, z)$. The above implies that $d_T(x, y) < d_T(x, z) = c_x(T)$, for all $x, y \in V(A)$. ∎

Lemma 4.3.2 directly implies the following statement:

Corollary 4.3.3 *For all* $x \in V(A)$*, we have* $c_x(T) > c_x(T')$.

Hence, we have that agent u's improving move decreases the cost for all agents in $V(A)$. For agents in $V(B)$ this may not be true: The cost of an agent $y \in V(B)$ can increase by agent u's move, as shown in Fig. 4.4. Interestingly, the next result guarantees that such an increase cannot be arbitrarily high.

Lemma 4.3.4 *Let* $x \in V(A)$ *and* $y \in V(B)$ *such that* $d_{T'}(x, y) = c_y(T')$*. It holds that* $c_x(T) > c_y(T')$.

PROOF In tree T we have $c_x(T) = d_T(x, u) + d_T(v, z) + 1$. Furthermore, in tree T' we have $c_y(T') = d_{T'}(x, u) + d_{T'}(w, y) + 1$. Since $c_u(T) > c_u(T')$, it follows that $d_T(w, y) < d_T(v, z)$, where $z \in V(B)$ is a vertex having maximum distance to u in T. Hence, we have $c_x(T) - c_y(T') = d_T(v, z) - d_T(w, y) > 0$. ∎

Towards a generalized ordinal potential function we will need the following:

Definition 4.3.5 (Sorted Cost Vector) *Let* G *be any network on* n *vertices. The* sorted cost vector *of* G *is* $\overrightarrow{c_G} = (\gamma_G^1, \ldots, \gamma_G^n)$*, where* γ_G^i *is the cost of the agent, who has the* i*-th highest cost in the network* G.

Observe that an agent having cost γ_G^n in a n-vertex network G is a 1-*center vertex* of G, as already defined in Definition 3.4.7.

Lemma 4.3.6 *Let T be any tree on n vertices. The sorted cost vector of T induces a generalized ordinal potential function Φ for the* Max-*SG on T.*

Proof Let u be any agent in T, who performs an edge-swap which strictly decreases her cost and let T' denote the network after agent u's swap. We show that $c_u(T) - c_u(T') > 0$ implies $\overrightarrow{c_T} >_{\text{lex}} \overrightarrow{c_{T'}}$, where $>_{\text{lex}}$ is the lexicographic order on $\mathbb{N}^n$. The existence of a generalized ordinal potential function Φ then follows by mapping the lexicographic order on $\mathbb{N}^n$ to an isomorphic order on $\mathbb{R}$.

Let the subtrees A and B be defined as above and let $c_u(T) - c_u(T') > 0$. By Lemma 4.3.2 and Lemma 4.3.4, we know that there is an agent $x \in V(A)$ such that $c_x(T) > c_y(T')$, for all $y \in V(B)$. By Lemma 4.3.2 and Corollary 4.3.3, we have that $c_x(T) > c_x(T')$, which yields that $\overrightarrow{c_T} >_{\text{lex}} \overrightarrow{c_{T'}}$. ∎

In the following, a special type of paths in the network will be important.

Definition 4.3.7 (Longest Path) *Let G be any connected network. Let u be any agent in G having cost $c_u(G) = k$. Any simple path in G, which starts at u and has length k is called a* longest path of agent u.

As we will see, 1-center vertices and longest paths are closely related.

Lemma 4.3.8 *Let T be any connected tree and let v^* be a 1-center-vertex of T. Vertex v^* must lie on all longest paths of all agents in $V(T)$.*

Proof Let P_{xy} denote the path from vertex x to vertex y in T. We assume towards a contradiction that there are two vertices $v, w \in V(T)$, where $c_v(T) = d_T(v, w)$, and that $v^* \notin V(P_{vw})$. Let $z \in V(T)$ be the only shared vertex of the three paths $P_{vv^*}, P_{wv^*}, P_{vw}$. We have $d_T(v, z) < d_T(v, v^*) \leq c_{v^*}(T)$ and $d_T(w, z) < d_T(w, v^*) \leq c_{v^*}(T)$. We show that $c_z(T) < c_{v^*}(T)$, which is a contradiction to v^* being a 1-center vertex in T.

Assume that there is a vertex $u \in V(T)$ with $d_T(u, z) \geq c_{v^*}(T)$. It follows that $V(P_{vz}) \cap V(P_{zu}) = \{z\}$, since otherwise $d_T(v^*, u) = d_T(v^*, z) + d_T(z, u) > c_{v^*}(T)$. But now, since $d_T(z, w) < c_{v^*}(T) \leq d_T(z, u)$, we have $d_T(v, u) > c_v(T)$, which clearly is a contradiction. Hence, we have $d_T(z, u) < c_{v^*}(T)$, for all $u \in V(T)$, which implies that $c_z(T) < c_{v^*}(T)$. ∎

Lemma 4.3.8, leads to the following observation.

Observation 4.3.9 *Let G be any connected network on n nodes and let $\overrightarrow{c_G} = (\gamma_G^1, \ldots, \gamma_G^n)$ be its sorted cost vector. We have $\gamma_G^1 = \gamma_G^2$ and $\gamma_G^n = \left\lceil \frac{\gamma_G^1}{2} \right\rceil$.*

Now we are ready to provide the key property which will help us upper-bound the convergence time.

Lemma 4.3.10 *Let $T = (V, E)$ be a connected tree on n vertices having diameter $D \geq 4$. After at most $\frac{nD-D^2}{2}$ moves of the* Max-*SG on T one agent must perform a move which decreases the diameter.*

Proof Let $v, w \in V$ such that $d_T(v, w) = D \geq 4$ and let P_{vw} be the path from v to w in T. Clearly, if no agent in $V(P_{vw})$ makes an improving move, then the diameter of the network does not change. On the other hand, if the path P_{vw} is the unique path in T having length D, then any improving move of an agent in $V(P_{vw})$ must decrease the diameter by at least 1. The network creation process starts from a connected tree having diameter $D \geq 4$ and, by Lemma 4.3.6, must converge to a stable tree in a finite number of steps. Moreover, Lemma 4.3.6 guarantees that the diameter of the network cannot increase in any step of the process. It was shown [ADHL13] that any stable tree has diameter at most 3. Thus, after a finite number of steps the diameter of the network must strictly decrease, that is, on all paths of length D some agent must have performed an improving move which reduced the length of the respective path. We fix the path P_{vw} to be the path of length D in the network which survives longest in this process.

It follows that there are $|V \setminus V(P_{vw})| = n - (D + 1)$ agents which can perform improving moves without decreasing the diameter. We know from Observation 4.3.9 and Lemma 4.3.8 that each one of those $n-(D+1)$ agents can decrease her cost to at most $\left\lceil \frac{D}{2} \right\rceil + 1$ and has to decrease her cost by at least 1 for each swap. We show that an swap of such an agent does not increase the cost of any other agent and use the minimum possible cost decrease per step to conclude the desired bound.

Let $u \in V(T) \setminus V(P_{vw})$ be an agent who decreases her cost by swapping the edge ux to uy and let T' be the tree after this edge-swap. Let $a, b \in V(T)$ be arbitrary agents. Clearly, if $\{u, y\} \not\subseteq V(P_{ab})$ in T', then $d_T(a, b) = d_{T'}(a, b)$. Let A be the tree of $T'' = (V, E \setminus \{uy\})$ which contains u and let B be the tree of T'' which contains y. W.l.o.g. let $a \in V(A)$ and $b \in V(B)$. By Corollary 4.3.3, we have $c_z(T) > c_z(T')$ for all $z \in V(A)$ and it follows that $V(A) \cap V(P_{vw}) = \emptyset$. Hence, it remains to analyze the change in cost of all agents in $V(B)$.

If no vertex on the path P_{ab} is a 1-center vertex in T', then, by Lemma 4.3.8, we have that $d_{T'}(a, b) < c_b(T')$. It follows that every longest path of agent b in T' lies entirely in subtree B which implies that $c_b(T') \leq c_b(T)$.

If there is a 1-center vertex of T' on the path P_{ab} in T', then let v^* be the last such vertex on this path. We have assumed that the diameters of T' and T are equal, which implies that P_{vw} is a longest path of agent v in T'. Since, by Lemma 4.3.8, any 1-center vertex of T' must lie on all longest paths, it follows that v^* is on the path P_{vw} and we have $v^* \in V(B)$. W.l.o.g. let $d_{T'}(v, b) \geq d_{T'}(w, b)$. We have $d_{T'}(a, b) = d_{T'}(a, v^*) + d_{T'}(v^*, b) \leq d_{T'}(v, v^*) + d_{T'}(v^*, b)$. Hence, we have $d_{T'}(a, b) \leq c_b(T')$. Since the path P_{bv} is in subtree B, we have $c_b(T') \leq c_b(T)$.

Now we can easily conclude the upper bound on the number of moves which do not decrease the diameter of T. Each of the $n - (D + 1)$ agents with cost at most D may decrease their cost to $\left\lceil \frac{D}{2} \right\rceil + 1$. If we assume a decrease of 1 per step, then

this yields the following bound:

$$(n-(D+1))\left(D-\left(\left\lceil\frac{D}{2}\right\rceil+1\right)\right)<(n-D)\frac{D}{2}=\frac{nD-D^2}{2}.$$ ∎

Proof (of Theorem 4.3.1) By Lemma 4.3.6, we know there exists a generalized ordinal potential function for the Max-SG on trees. Hence, we know that this game is a FIPG and we are left to bound the maximum number of improving moves needed for convergence. It was already shown [ADHL13] that the only stable trees of the Max-SG on trees are stars or double-stars. Hence, the process must stop at the latest when diameter 2 is reached. Observe that in any unstable tree having diameter 3 there can be only one unhappy agent. If this agent moves, then the diamter drops from 3 to 2 and a star emerges.

Let $N_n(T)$ denote the maximum number of moves needed for convergence in the Max-SG on the n-vertex tree T. Let $D(T)$ be the diameter of T. Let $D_{i,n}$ denote the maximum number of steps needed to decrease the diameter of any n-vertex tree having diameter i by at least 1. Hence, we have

$$N_n(T)\leq 1+\sum_{i=4}^{D(T)}D_{i,n}\leq 1+\sum_{i=4}^{n-1}D_{i,n}\ ,$$

since the maximum diameter of a n-vertex tree is $n-1$. The additional step is the possible move which decreases the diameter from 3 to 2. By applying Lemma 4.3.10 and adding the steps which actually decrease the diameter, this yields

$$\begin{aligned}N_n(T)&\leq 1+\sum_{i=4}^{n-1}D_{i,n}<1+\sum_{i=4}^{n-1}\left(\frac{ni-i^2}{2}+1\right)\\&<n+\frac{n}{2}\left(\sum_{i=1}^{n}i\right)-\frac{1}{2}\left(\sum_{i=1}^{n}i^2\right)\in\mathcal{O}(n^3).\end{aligned}$$ ∎

The following result shows that we can speed up the convergence time by employing a very natural move policy - the mcBRD. The speed-up is close to optimal, since it is easy to see that there are instances in which $\Omega(n)$ steps are necessary.

Theorem 4.3.11 *The mcBRD in the* Max-*SG on trees converges in* $\Theta(n\log n)$ *moves.*

Proof We prove Theorem 4.3.11, by proving the lower and the upper bound separately, starting with the former. Since the max cost policy is employed to choose the moving agent, we need two additional observations, which are easy to see.

Observation 4.3.12 *An agent having maximum cost in a tree network T must be a leaf of T.*

Observation 4.3.13 *Let u be an unhappy agent in $T = (V, E)$ and let u be a leaf of T and let v be u's unique neighbor. Let B be the tree of $T' = (V, E \setminus \{uv\})$ which contains v. The edge-swap uv to uw, for some $w \in V(B)$ is a best possible move for agent u if w is a 1-center vertex of B.*

Lemma 4.3.14 *There is a tree T on n vertices where mcBRD(T) in the* MAX-*SG needs $\Omega(n \log n)$ moves for convergence.*

PROOF We consider the path on n-vertices $P_n = v_1 v_2 \dots v_n$ of length $n-1$. We apply the max cost policy and for breaking ties we will always choose the vertex having the smallest index among all vertices having maximum cost. If a maximum cost vertex has more than one best response move, then we choose the edge-swap towards the new neighbor having the smaller index. With these assumptions and with Observation 4.3.12 and Observation 4.3.13, we have that the 1-center-vertex having the smallest index will "shift" towards a higher index, from $v_{\lceil n/2 \rceil}$ to v_{n-2}. Finally, agent v_n is the unique agent having maximum cost and her move transforms the tree to a star. See Fig. 4.5 for an illustration for $n = 9$.

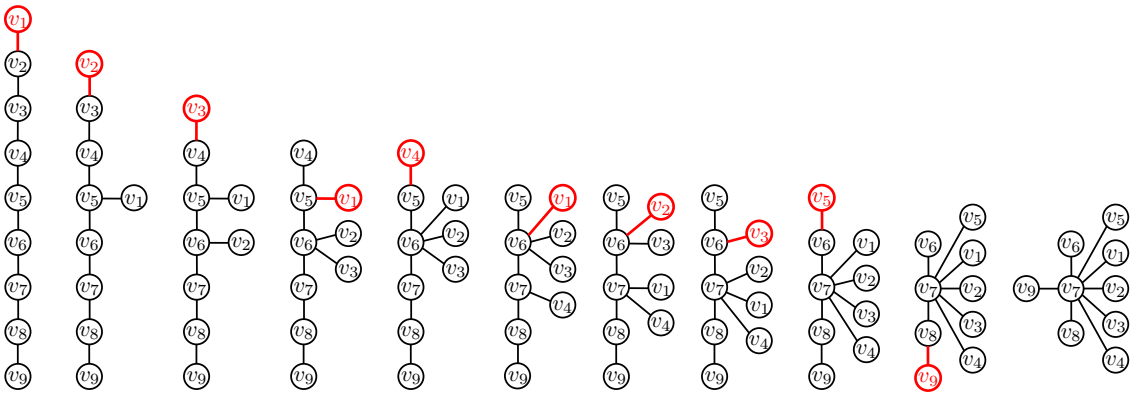


Figure 4.5.: The convergence process for with $n = 9$ in the MAX-SG on P_n.

We start by analyzing the change in costs of agent v_1. Clearly, $c_0 := c_{v_1}(P_n) = n-1$. By Observation 4.3.13, we know that v_1's best swap connects to the minimum index 1-center vertex of the tree without vertex v_1. Hence, after the best move of v_1 this agent has cost

$$c_1 := \left\lceil \frac{c_0 - 1}{2} \right\rceil + 1 > \frac{c_0}{2} \ .$$

When v_1 is chosen to move again, her cost can possibly decrease to

$$c_2 := \left\lceil \frac{c_1 - 1}{2} \right\rceil + 1 > \frac{c_0}{4} \ .$$

After the i-th move of v_1 her cost is at least

$$c_i := \left\lceil \frac{c_{i-1} - 1}{2} \right\rceil + 1 > \frac{c_0}{2^i} \ .$$

Thus, the mcBRD allows agent v_1 to move at least $\log \frac{c_0}{3}$ times until she is connected to vertex v_{n-2}, the center of the final star, where she has cost 3.

The above implies that the number of moves of every agent allowed by the mcBRD only depends on the cost of that agent when she first becomes a maximum cost agent. Moreover, since all moving agents are leaves, no move of an agent increases the cost of any other agent. By construction, the cost of every moving agent is determined by her distance towards vertex v_n. Since agent v_n does not move until in the last step of the process, we have that a move of agent v_i does not change the cost of any other agent $v_j \neq v_n$ who moves after v_i. It follows that we can simply add up the respective lower bounds on the number of moves of all players, depending on the cost when they first become maximum cost agents. It is easy to see that agent v_i becomes a maximum cost agent, when the maximum cost is $n - i$. Let $M(P_n)$ denote the number of moves of the MAX-SG on P_n with the mcBRD and the above tie-breaking rules. This yields

$$M(P_n) > \sum_{c_0=n-1}^{4} \log \frac{c_0}{3} \in \Omega(n \log n) \ .$$

Lemma 4.3.15 *The mcBRD on a n-vertex tree T in* MAX-*SG needs $\mathcal{O}(n \log n)$ moves to converge to a stable tree.*

PROOF Consider any tree T on n vertices. By Observation 4.3.12, we know that only leaf-agents are allowed to move by the max cost policy, which implies that no move of any agent increases the cost of any other agent. Observation 4.3.13 guarantees that the best possible move of a leaf-agent u having maximum cost c decreases agent u's cost to at most $\left\lceil \frac{c}{2} \right\rceil + 1$. Hence, after $\mathcal{O}(\log n)$ moves of agent u her cost must be at most 3. If the tree converges to a star, then agent u may move one more time. If we sum up over all n agents, then we have that after $\mathcal{O}(n \log n)$ moves the tree must be stable.

This concludes the proof of Theorem 4.3.11. ∎

4.3.2. Dynamics on General Networks

In this section we show that allowing cycles in the initial network completely changes the dynamic behavior of the MAX-SG.

Theorem 4.3.16 *The* MAX-*SG on general networks admits best response cycles. Moreover, no move policy can enforce convergence. The first result holds even if agents are allowed to perform multi-swaps.*

PROOF We prove the theorem by showing that there exists an initial network which induces a best response cycle and where in every step of the cycle exactly one agent

is unhappy. The existence of the best response cycle shows that the MAX-SG on this instance does not have the finite improvement property. The fact that in every step of the cycle exactly one agent is unhappy shows that no move policy can avoid that cyclic behavior. In every step, swapping one edge suffices to achieve the best possible cost decrease for the moving agent. Hence, there exists a best response cycle even if agents are allowed to perform multi-swaps. However, note that with multi-swaps it is no longer true that there is only one unhappy agent in every step.

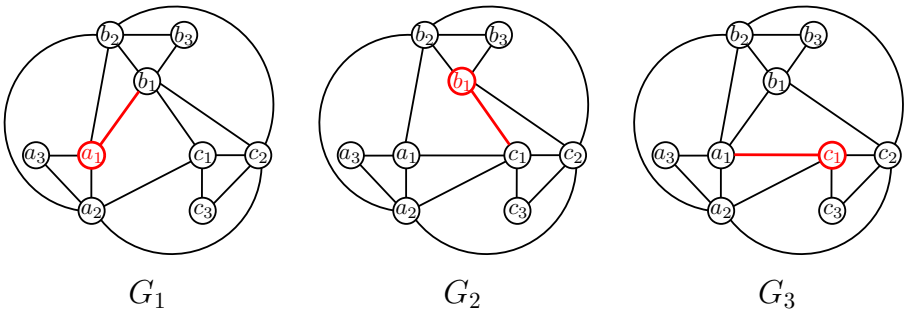


Figure 4.6.: The steps of a best response cycle for the MAX-SG on general networks.

Consider the initial network G_1 which is depicted in Fig. 4.6 (left). Note that only agents a_1, a_3, b_3 and c_3 have cost 3 while all other agents have cost 2. Clearly, agents having cost 2 cannot improve on their current situation. Agents a_3, b_3, c_3 cannot perform an improving move, since all of them have exactly two vertices in distance 3 and there is no vertex which is a neighbor of both of them. This leaves agent a_1 as the only possible candidate for an improving edge-swap. A best possible move for agent a_1 is the swap a_1b_1 to a_1c_1, which yields a distance decrease of 1 which is clearly optimal. This swap transforms G_1 into G_2, which is depicted in Fig. 4.6 (middle). Observe that G_2 is isomorphic to G_1, with agent b_1 facing exactly the same situation as agent a_1 in G_1. Agent b_1 has the swap b_1c_1 to b_1a_1 as best response move and we end up in network G_3, shown in Fig. 4.6 (right). Again, G_3 is isomorphic to G_1, now with agent c_1 being the unique unhappy agent. Agent c_1's best possible swap transforms G_3 back into G_1. ■

4.4. Dynamics in Asymmetric Swap Games

In this section we consider the SUM-ASG and the MAX-ASG. Note that now we assume that each edge has an owner and only this owner is allowed to swap the edge. We show that we can directly transfer the results from Section 4.2 and Section 4.3 to the asymmetric version if the initial network is a tree. On general networks we show even stronger negative results.

Observe that the instance used in the proofs of Theorem 4.2.20 and Theorem 4.3.16 show that best response cycles in the Swap Game are not necessarily best response

cycles in the Asymmetric Swap Game. We will show the rather counter-intuitive result that this holds true for the other direction as well.

4.4.1. Asymmetric Swap Games on Trees

The results in this section follow from the results in Section 4.2.1 and Section 4.3.1 and are therefore stated as corollaries.

Corollary 4.4.1 *The* Sum-*ASG and the* Max-*ASG on n-vertex trees are both a poly-FIPG and both must converge to a stable tree in $\mathcal{O}(n^3)$ steps.*

Proof By Theorem 4.2.1 we have that the Sum-SG on trees is an ordinal potential game, where the social cost, which is the sum of all agent's costs, serves as ordinal potential function. Note that ordinal potential games are a subclass of FIPG [MS96]. Furthermore, by Theorem 4.2.7 and Theorem 4.3.1, we know that the Sum-SG and the Max-SG on n-vertex trees must converge in $\mathcal{O}(n^3)$ steps.

The only difference in the *asymmetric* version of this game is that edges have owners and only the respective owner is allowed to swap an edge. Clearly, since any improving swap decreases the value of the (generalized) potential function, this is independent of the edge-ownership. Furthermore, with edges having owners, we have that in each network less moves are possible and every moving agent has, compared with the Swap Game, at most the same number of admissible strategies in any step. Thus, the convergence process cannot be slower. ∎

Corollary 4.4.2 *Using the mcBRD and assuming a n-vertex tree as initial network, we have that*

- *the* Sum-*ASG converges in $\max\{0, n-3\}$ steps, if n is even and in $\max\{0, n + \lceil n/2 \rceil - 5\}$ steps, if n is odd. Moreover, both bounds are tight and asymptotically optimal.*

- *the* Max-*ASG converges in $\Theta(n \log n)$ steps.*

Proof The results from Section 4.2.1 and Section 4.3.1 on speeding up the convergence process with the mcBRD carry over to ASGs. The reason for this is that in all used lower bound constructions it holds that whenever an edge is swapped more than once, then it is the same incident agent who moves again. Hence, we can assign the edge-ownership to this agent and get the same lower bounds in the asymmetric version. The upper bounds carry over trivially, since agents cannot have more admissible new strategies in any step in the asymmetric version compared to the version without edge-owners.

4.4.2. Asymmetric Swap Games on General Networks

If we move from trees to general initial networks, we get a very strong negative result for the SUM-ASG: There is no hope to enforce convergence if agents stick to playing best responses even if multi-swaps are allowed.

Theorem 4.4.3 *The* SUM*-ASG on general networks is not weakly acyclic under best response. Moreover, this result holds true even if agents can swap multiple edges in one step.*

PROOF We give a network which induces a best response cycle. Additionally, we show that in each step of this cycle exactly one agent can decrease her cost by swapping an edge and that the best possible swap for this agent is unique in every step. Furthermore, we show that the moving agent cannot outperform the best possible single-swap by a multi-swap. This implies that if agents stick to best response moves then *no* best response dynamic can enforce convergence to a stable network and allowing multi-swaps does not alter this result.

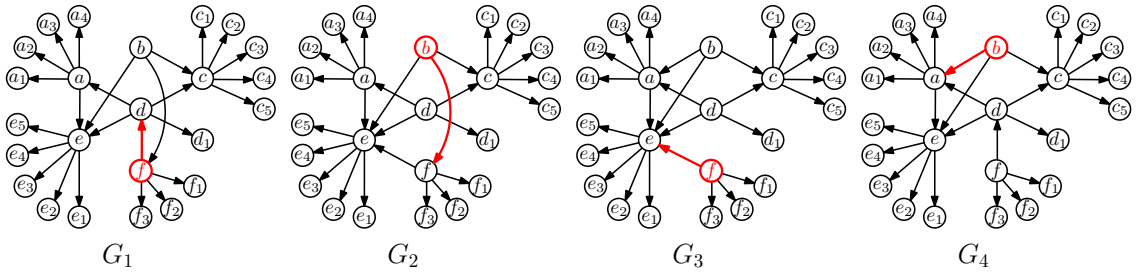


Figure 4.7.: The steps of a best response cycle for the SUM-ASG on general networks. Note that edge directions indicate edge-ownership. Edges point away from their owners. All edges allow two-way communication.

The best response cycle consists of the networks G_1, G_2, G_3 and G_4 given in Fig. 4.7. We begin with showing that in $G_1, \dots, G_4$ all agents, except agent b and agent f, cannot perform an improving strategy-change even if they are allowed to swap multiple edges in one step.

In $G_1, \dots, G_4$ all leaf-agents do not own any edges and the agents c and e cannot swap an edge since otherwise the network becomes disconnected. For the same reason, agent d cannot move the edge towards d_1. Agent d owns three other edges, but they are optimally placed since they are connected to the vertices having the most leaf-neighbors. It follows that agent d cannot decrease her cost by swapping one edge or by performing a multi-swap. Note that this holds true for all networks $G_1, \dots, G_4$, although the networks change slightly. Agent a cannot move her edges towards a_i, for $1 \leq i \leq 4$. On the other hand, it is easy to see that agent a's edge towards vertex e cannot be swapped to obtain a strict cost decrease since the most promising choice, which is vertex c, yields the same cost in G_1 and G_4 and even higher cost in G_2 and G_3. Trivially, no multi-swap is possible for agent a.

Now, we consider agent b and agent f. First of all, observe that in $G_1, \dots, G_4$ agent f owns exactly one edge which is not a bridge. Thus, agent f cannot perform a multi-swap in any step of the best response cycle. Agent b, although owning three edges, is in a similar situation: Her edges to vertex c and e can be considered as fixed, since swapping one or both of them does not yield a cost decrease in $G_1, \dots, G_4$. Hence, agent b and agent f each have one "free" edge to operate with. In G_1 agent b's edge towards f is placed optimally, since swapping towards a or d does not yield a cost decrease. In G_3, agents b's edge towards a is optimal, since swapping towards d or f does not decrease agent b's cost. Analogously, agent f's edge towards e in G_2 and her edge towards d in G_4 are optimally placed.

Finally, we describe the best response cycle: In G_1 agent f can improve and her unique best possible edge-swap in G_1 is the swap from d to e, yielding a cost decrease of 4. In G_2 agent b has the swap from f to a as unique best improvement which yields a cost decrease of 1. In G_3 have agent f being unhappy with her strategy and the unique best swap is the one from e to d yielding an improvement of 1. In G_4 it is agent b's turn again and her unique best swap is from a to f which decreases her cost by 3. After agent b's swap in G_4 we arrive again at network G_1, hence $G_1, \dots, G_4$ is a best response cycle where in each step exactly one agent has a single-swap as unique best possible improvement.

Remark 4.4.4 *Note that the best response cycle presented in the proof of Theorem 4.4.3 is not a best response cycle in the* Sum-SG. *The swap fb to fe of agent f in G_1 yields a strictly larger cost decrease than her swap fd to fe.*

Compared to Theorem 4.4.3, we show a slightly weaker negative result for the Max-version.

Theorem 4.4.5 *The* Max-*ASG on general networks admits best response cycles. Moreover, no move policy can enforce convergence.*

Proof We show that there exists a best response cycle for the Max-ASG, where no move policy can enforce convergence. Our cycle, shown in Fig. 4.8, has six steps $G_1, \dots, G_6$. In G_3 and G_6 there are two unhappy agents whereas in the other steps there is exactly one unhappy agent. It turns out that independently which one of the two agent moves in G_3 or G_6, there is a best response move which leads back to a network in the cycle. This implies that no move policy may enforce convergence.

First of all, note that the networks G_2 and G_5 are isomorphic. The same holds true for the networks G_3 and G_6. We start by showing that in the networks $G_1, \dots, G_4$ only the highlighted agents are unhappy. Then we will analyze the best response moves of the unhappy agents in the respective networks.

Consider any G_i, where $1 \leq i \leq 4$. Clearly, no leaf-agent of G_i can perform any swap. Agent g has cost 3 in G_i. We have $d_{G_i}(g, l_1) = d_{G_i}(g, l_5)$ and there is no

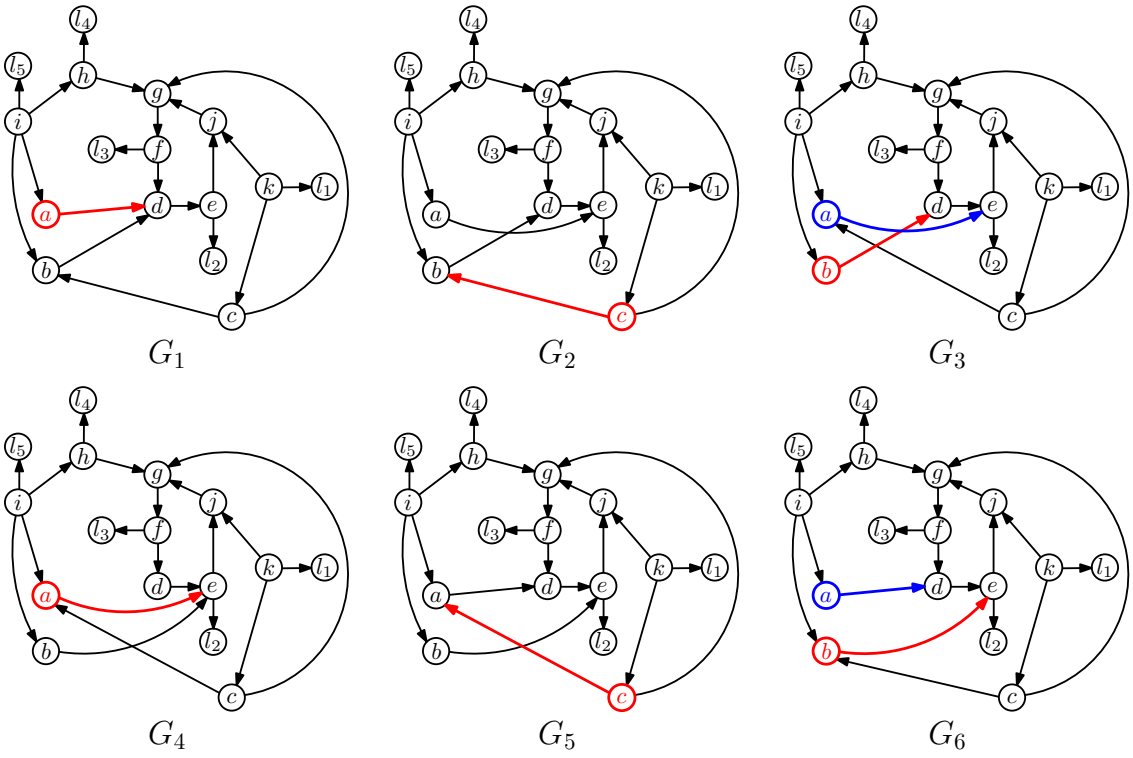


Figure 4.8.: The steps of a best response cycle for the MAX-ASG on general networks.

vertex in G_i which is a neighbor to both l_1 and l_5. Thus, agent g cannot achieve cost 2, which implies that agent g does not want to perform a move in G_i. By the same argument, it follows that any agent having cost 3 must be happy. Hence, we have that agent b in G_1 and G_2 and agent c in G_3 and G_4 do not swap.

Agent d, having cost 4 in G_i, cannot perform an improving move in G_i, since $d_{G_i}(d, l_1) = d_{G_i}(d, l_4) = 4$ and any such improving move must connect to a vertex which has distance at most 2 to vertex l_1. These vertices are c, j, k and l_1 but any of them has distance at least 3 towards l_4 in G_i. Agents f and i, both having cost 4 in G_i, each face an analogous situation, since $d_{G_i}(f, l_1) = d_{G_i}(i, l_1) = d_{G_i}(f, l_5) = d_{G_i}(i, l_3) = 4$ and since vertices c, j, k, l_1 have distance at least 3 to l_5 or l_3 in any G_i.

Agent h, having cost 4 in G_i has both l_1 and l_2 in distance 4. The only vertex which has distance at most 2 to both of them is vertex j. But if h swaps towards vertex j, then this yields distance 4 towards towards vertex l_3.

Now we consider agent e. Any strategy yielding cost at most 3 for agent e must connect to c, j, k or l_1, since otherwise vertex l_1 would be in distance 4. But, since $d_{G_i}(e, l_4) = 4$ and since c, j, k, l_1 all have distance at least 3 towards l_4, no such strategy can exist. For agent j, having cost 4 in G_i. the situation is similar. Any strategy having cost at most 3 for agent j must connect to a, b, h, i or l_5, since

otherwise j's distance towards l_5 would be 4. But if j swaps away from g to any vertex in $\{a, b, h, i, l_5\}$, then her distance to l_3 increases to 4.

Agent k, having cost 4 in G_i, has vertex l_4 and l_5 in distance 4. Thus, to achieve as cost of at most 3, agent k must connect to vertex h or i. Both h and i have distance at least 3 towards l_3. Since $d_{G_i}(k, l_3) = 4$, it follows that agent k cannot perform an improving move.

Now, we are left with agent c in G_1, agent a in G_2 and agent b in G_4, all having cost 4. We have $d_{G_1}(c, l_2) = 4$. Thus, any strategy which yields cost at most 3 must connect to d, e, j or l_2. If c swaps away from g, then her distance to l_4 increases to 4. If c swaps away from b, then her distance to l_5 increases to 4. Hence, agent c cannot swap an edge to decrease her cost in G_1. For agent a in G_2 and agent b in G_4 the situation is similar. We have $d_{G_2}(a, l_1) = d_{G_2}(a, l_3) = d_{G_4}(b, l_1) = d_{G_4}(b, l_3) = 4$. Both agents cannot decrease their cost by swapping an edge in the respective network, since all vertices which have distance at most 2 to l_1 have distance at least 3 to l_3.

Finally, we analyze the best response moves of all unhappy agents in any G_i. In G_1 only agent a, having cost 5, is unhappy. There is only one vertex having distance 5 to a, which is l_1. Thus, agent a could swap towards any vertex having distance at most 3 to l_1 to improve on this distance. Possible target-vertices are b, c, e, g, j, k and l_1. But, after any such swap, agent a must have distance 4 to vertex l_3, which implies that agent a cannot decrease her cost by more than 1. Furthermore, swapping towards c, k or l_1 yields distance 5 towards l_3, which rules out these target-vertices. If agent a performs the swap ad to ae in G_1, then we obtain network G_2. In G_2 only vertex c, having cost 4, is unhappy. Her unique vertex in distance 4 is l_2. Thus, a swap towards a, d, e, j or l_2 would reduce this distance. However, only the swap towards a is an improving move, since in all other cases agent c's distance to l_6 increases to 4. This swap transforms G_2 into G_3. In G_3 we have that agent a and agent b are unhappy. Agent b in G_3 is in a similar situation as agent a in G_1. Her only vertex in distance 5 is the leaf l_1 and by swapping towards a, c, e, g, j, k or l_1 this distance can be reduced. But any such swap yields that agent b's distance to l_3 increases to at least 4, which implies that a cost decrease by 1 is optimal. Furthermore, the swaps towards c, k or l_1 are not improving moves since these yield distance 5 towards l_3. If agent b performs the swap bd to be we obtain network G_4. Agent a in G_3 has l_3 as her only vertex in distance 4. There is exactly one swap for agent a, which decreases this distance to 3 without increasing any other distance to more than 3, and this is the swap ae to ad. Note that this swap of agent a transforms G_3 to a network which is isomorphic to network G_1. Finally, we argue for agent a in G_4. This agent is in a similar situation as agent a in G_3. Her only vertex in distance 4 is l_3 and the swap towards d is the only swap which does not increase any other distance to more than 3. Thus, this move is agent a's unique best response in G_4 and this move leads to network G_5. ■

If played on a non-complete host-graph, then we get the worst possible dynamic behavior.

Corollary 4.4.6 *The* Sum-*ASG and the* Max-*ASG on a non-complete host graph are not weakly acyclic.*

Proof (of Corollary 4.4.6, Sum-version) We use the best response cycle $G_1, \dots, G_4$ shown in Fig. 4.7 and let the host graph H be the complete graph but without the edge af. In this case, agent f's best response move in G_1 is her only possible improving move. For the networks G_2, G_3 and G_4 it is easy to check that the respective moving player has exactly one possible improving move. ∎

Proof (of Corollary 4.4.6, Max-version) We use the best response cycle $G_1, \dots, G_6$ from Fig. 4.8. As host graph H we use the complete graph, but without edges ab, ag, aj, bg, bj. By inspecting the proof of Theorem 4.4.5, it is easy to see that in each step of the cycle the moving agent has exactly one improving move. ∎

4.4.3. The Boundary between Convergence and Non-Convergence

In this section we explore the boundary between guaranteed convergence and cyclic behavior. Quite surprisingly, we can draw a sharp boundary by showing that the undesired cyclic behavior can already occur in n-vertex networks having exactly n edges. Thus, one non-tree edge suffices to radically change the dynamic behavior of Asymmetric Swap Games. Our constructions are such that each agent owns exactly one edge, which corresponds to the uniform unit budget case, recently introduced by Ehsani, Fazli, Mehrabian, Sadeghabad, Safari, Saghafian and ShokatFadaee [EFM+11]. Hence, even if the networks are built by identical agents having a budget the cyclic behavior may arise. This answers the open problem raised in [EFM+11] in the negative.

Theorem 4.4.7 *The* Sum-*ASG and the* Max-*ASG admit best response cycles on a network where every agent owns exactly one edge.*

Proof (of Theorem 4.4.7, Sum-version) The network which induces a best response cycle and the steps of the cycle $G_1, \dots, G_4$ are shown in Fig. 4.9. Let n_k denote the number of vertices having the form k_j, for some index j.
In network G_1, agent a_1 has only one improving move, which is the swap from b_1 to c_1. This swap reduces agent a_1's cost by 1, since $n_c = n_b + n_d + 1$. After this move, in network G_2, agent b_1 is no longer happy with her edge towards d_1, since by swapping towards a_4 she can decrease her cost by 2. This is a best possible move for agent b_1 (note that a swap towards a_3 yields the same cost decrease). But now, in network G_3, by swapping back towards vertex b_1, agent a_1 can additionally decrease her distances to vertices a_4 and a_5 by 1. This yields that agent a_1's swap

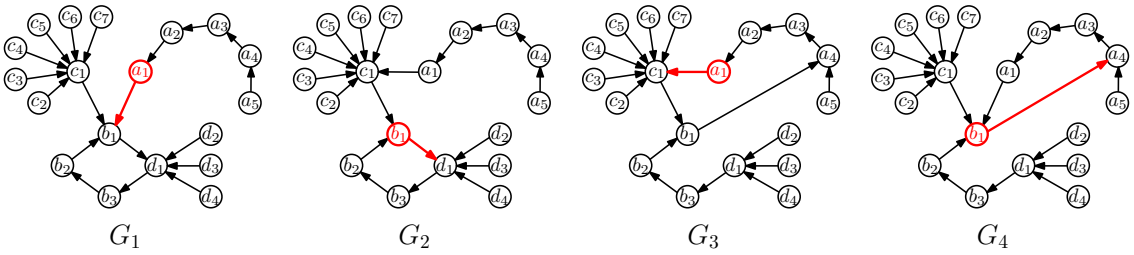


Figure 4.9.: The steps of a best response cycle for the SUM-ASG where each agent owns exactly one edge.

from c_1 to b_1 decreases her cost by 1. This is true, since all distances to c_j vertices increase by 1 but all distances to b_i and d_l vertices and to a_4 and a_5 decrease by 1 and since we have $n_c = n_b + n_d + 1$. Note that this swap is agent a_1's unique improving move. By construction, we have that after agent a_1 has swapped back towards b_1, that is, in network G_4, agent b_1's edge towards a_4 only yields a distance decrease of 7. Hence, by swapping back towards d_1, agent b_1 decreases her cost by 1, since her sum of distances to the d_j vertices decreases by 8. This swap is the unique improving move of agent b_1 in network G_4 and this move leads to network G_1. ■

PROOF (OF THEOREM 4.4.7, MAX-VERSION) Fig. 4.10 shows the four networks $G_1, \ldots, G_4$ which form the steps of a best response cycle for the MAX version, where each agent owns exactly one edge.

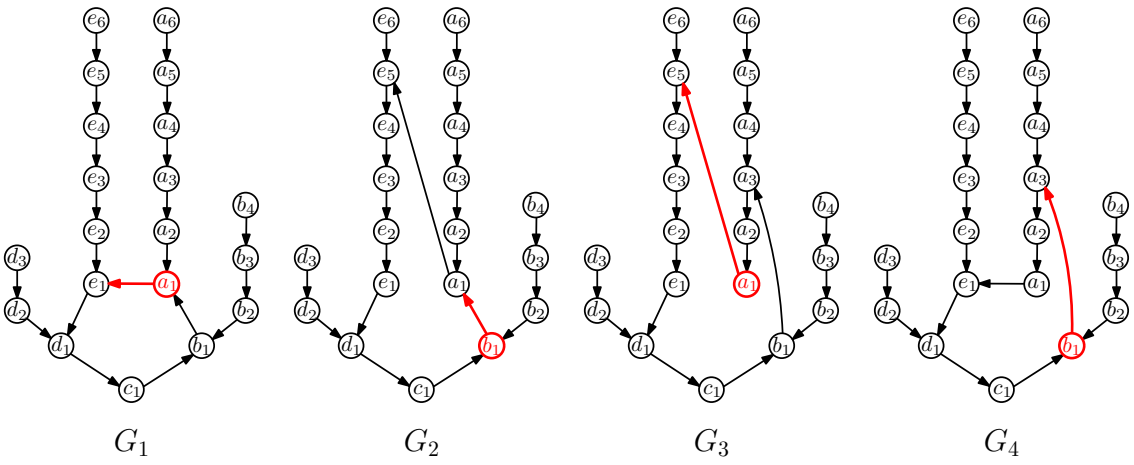


Figure 4.10.: The steps of a best response cycle for the MAX-ASG where each agent owns exactly one edge.

In the first step of the cycle, in network G_1, agent a_1 can decrease her maximum distance from 6 to 5 by swapping from e_1 to one of the vertices $e_2, \ldots, e_5$. Note that all these swaps yield the same distance decrease of 1 and, since a_1 has distance

5 towards a_6 and swapping towards any of the a_i-vertices is obviously sub-optimal, no other swap can yield a larger cost decrease. By agent a_1 performing the swap towards e_5 we obtain the network G_2.

Now, agent b_1 can improve her situation with a swap from a_1 to a_2 or to a_3. Both possible swaps reduce her maximum distance from 6 to 5. This is best possible: The cycle has length 9 before agent b_1's move, which implies that there are two vertices on the cycle which have distance 4 to b_1. Observe that agent b_1 must swap towards a vertex which has at most distance 4 to vertex a_6 to reduce her maximum distance. Clearly, only one of the a_i vertices, with $i \neq 1$, is possible. However, swapping away from a_1 must increase the cycle by at least 1, which implies that after this swap agent b_1 must have at least one cycle-vertex in distance 5. Hence, no swap can decrease agent b_1's maximum distance by more than 1. Let agent b_1 perform the swap towards a_3 and we end up with the network G_3.

Agent a_1 in G_3 finds herself sitting in a large cycle having maximum distance 7 towards d_3 and distance 6 to vertex b_4. By swapping from e_5 to one of the vertices e_1, e_2, e_3, agent a_1 can reduce her maximum distance to 6. This is optimal, since all improving moves must swap towards a vertex having at most distance 5 to vertex d_3 and agent a_1 cannot move to far away from vertex e_6. The vertices e_1, e_2, e_3 are the only vertices which satisfy both conditions. Let a_1 swap towards e_1 and we get the network G_4.

In the last step of the best response cycle, in network G_4, we have agent b_1 with maximum distance 8 towards vertex e_6. Clearly, agent b_1 wants to move closer to this vertex but this implies that she must move away from vertex a_6. The only possible compromise between both distances is a swap either to a_1 or to e_1. Both these swaps yield a decrease of b_1's maximum distance by 1. By swapping towards vertex a_1, we end up with our starting network G_1 and the cycle is complete. ∎

Remark 4.4.8 *We can give best response cycles for both versions of the ASG for the case where every agent owns exactly two edges. We conjecture that such cyclic instances also exist for all cases where every agent owns exactly k edges, for any $k \geq 3$. In particular, it would be interesting if there is a generic construction which works for all $k \geq 1$.*

4.4.4. Empirical Study of the Bounded-Budget Version

We have conducted extensive simulations of the convergence behavior and the obtained results provide a sharp contrast to our mostly negative theoretical results for both versions of the ASG. Our experiments show for the bounded-budget version [EFM+11] a surprisingly fast convergence in at most $5n$ steps under the max cost policy or by choosing the moving agents uniformly at random and by enforcing best responses. Despite millions of trials we have not found any best response cycle in our experiments. This indicates that our negative results may be only very rare pathological examples.

We first describe the experimental setup, then we will discuss the obtained results for the bounded-budget version of the ASG for the Sum and the Max-version of the distance-cost function.

Experimental Setup

One run of our simulation can be described as follows: First we generate a random initial network, where every agent owns exactly k edges. Then, for every step of the process, the move policy decides which agents is allowed to perform a best possible edge-swap. After the respective swap has happened, we again let the move policy decide which agent is allowed to move next. We count the number of steps until a stable network is found. Thus, the number of steps equals the number of performed moves.

Computing a best possible edge-swap of an agent can be done in polynomial time by simply checking all possible edge-swaps and re-computing the cost.

Under the max cost policy we calculate the agents' costs and check in descending order if the respective agent can perform an improving move. If we have found an unhappy agent, then we calculate the best possible edge-swap for this agent, breaking ties uniformly at random, and we let this agent perform the respective move. Then the process starts all over again until there is no unhappy agent left.

Under the random policy we choose one agent uniformly at random and check if this agent can perform an improving move. If not, then we remove this agent from the set of candidates and we choose another agent uniformly at random from the remaining candidates. We proceed iteratively until we have found an unhappy agent or until no candidate is left. In the former case, we let this agent perform a best possible edge-swap and start all over again (with all agents being a possible candidate again). If the latter happens, then we stop.

The initial network is generated as follows: We start with an empty graph G on n vertices. Then, to ensure connectedness, we create a random spanning tree among all n agents as follows: We start with a uniformly chosen random pair of agents and insert the respective edge into G. The owner of this edge is chosen uniformly at random among its endpoints. We mark both vertices. Then, we iteratively choose one unmarked agent uniformly at random from the set of unmarked agents and one marked agent uniformly from the set of all marked agents and we insert the respective edge and mark the former agent. The edge-ownership is chosen uniformly at random with the constraint that no agent is allowed to own more than k edges. If all agents are marked, then we have that G is a spanning tree. Now we proceed inserting edges into G as follows: First we mark all agents who already own k edges. Then we iteratively choose one unmarked agent and one other agent uniformly at random and insert the edge with the first agent being its owner, if the edge is not already present in G. If the edge is present, then we randomly choose another suitable pair of agents. Again we mark agents having already k edges. We stop, if there is no unmarked agent left.

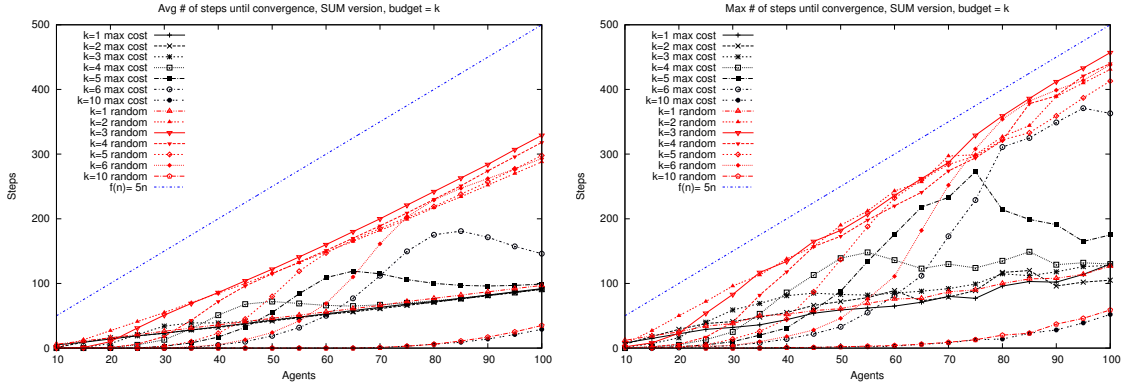


Figure 4.11.: Experimental results for the SUM-ASG with budget k. The average number of steps needed for convergence is plotted on the left, the maximum number of steps needed for convergence is plotted on the right. Each point is the average/maximum over the number of steps needed for convergence of 10000 trials with random initial networks where each agent owns exactly k edges.

The used Python script can be found in Appendix A.1.

Experimental Results and Discussion

Results for the Sum-ASG Our obtained results for the SUM-ASG in the bounded-budget version can be found in Fig. 4.11. We simulated 10000 runs for each configuration and the average and the maximum of the observed convergence time for each configuration is plotted. Here a configuration consists of the number of agents in the initial network and the choice of the move policy. The results for the max cost policy are plotted in black, whereas the results for the random policy are shown in red.

First of all, note that no run took longer than $5n$ steps, where n is the number of agents, and that the max-cost policy yields faster convergence than the random policy. The only exceptions here are the random policy in the cases where $k = 1$ and $k = 10$, which shows roughly the same behavior than the max cost policy. For the case $k = 10$ the number of agents seems to small to produce the difference. Indeed, for all other budgets we see that the plots for the random policy and the max cost policy are close together for small numbers of agents and start to separate as n grows larger. Note that for $k = 1$ only roughly n steps are needed for convergence under both move policies. This is to be expected for the max cost policy since the initial network is almost a tree and we have shown in Corollary 4.4.2 that the SUM-ASG on trees converges in at most $n + \lceil n/2 \rceil - 5$ steps.

For $k > 1$ the results are particularly interesting. Under the max cost policy our simulations reveal a rather curious behavior: The convergence time increases super linear until it peaks and then, for larger n, it converges to n. One possible

explanation for this is the ratio of edges versus non-edges in the network. For small n we have that the initial networks are very dense, which yields that agents have very short distances to each other. This implies that moves yield a relatively low cost-decrease since only a few distances can be reduced, which implies that after a small number of steps, no such move is available and the convergence process stops. For large n we have that the initial networks are very sparse, which implies that agents "at the perimeter" can achieve a large cost-decrease by performing a move. If only such agents move, then we have a sequence of moves which reduce a high number of individual distances and this leads to fast convergence. The slowest convergence time is achieved if the ratio of present edges over all possible edges is between $\frac{1}{7}$ and $\frac{1}{6}$.

Under the random policy, we see a completely different behavior if $k > 1$. For example, note the difference between the $k = 2$ case of the random versus the max cost policy. Here we have the expected behavior that the convergence time is strictly increasing for larger n. Interestingly, except for small n, the convergence time grows only linear in n. The super linear increase for small n can be explained as under the max cost policy. The initial networks are too dense to admit a high number of best possible improving moves. For large n, the results can be explained as follows: in contrast to the max cost policy we have that the random policy often picks agents who already have a relatively central position in the network. Such agents can improve only slightly. Thus, moves of such agents only reduce a small number of individual distances, which explains why a lot of agents remain unhappy with their situation.

Results for the Max-ASG The results for the MAX-ASG under both move policies can be found in Fig. 4.12. As in the SUM-version, we simulated 10000 runs for each configuration. The results for the max cost policy are shown in black whereas the results for the random policy are plotted in red.

The plots show that, with one exception, every run of our simulations converged in less than $5n$ steps. Hence, the MAX-version yields the same fast convergence to a stable network as in the SUM-version. However, there is a crucial difference: Here the different move policies do not yield a different convergence behavior. The curves for the random policy and the max cost policy for the same budget k are very close to each other. For $k \geq 4$ they are almost indistinguishable. For smaller k we see that the random policy slightly outperforms the max cost policy. The reason for the similar behavior under both move policies is that in the MAX-version there are significantly more agents which all have maximum cost than in the SUM-version. After a small number of steps we even have the situation that a large fraction of all agents has maximum cost. Thus, for an agent we have that choosing randomly among the maximum cost agents and choosing randomly among all agents yields no significant difference in the probability of being chosen.

Observe that for $k = 1$ the convergence time of the max cost policy grows slightly

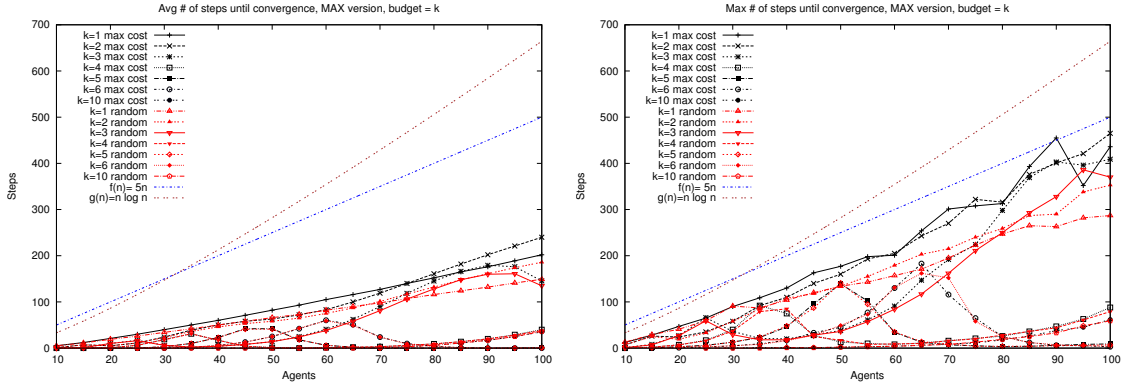


Figure 4.12.: Experimental results for the MAX-ASG with budget k. The average number of steps needed for convergence is plotted on the left, the maximum number of steps needed for convergence is plotted on the right. Each point is the average/maximum over the number of steps needed for convergence of 10000 trials with random initial networks where each agent owns exactly k edges.

super linear and is well below $n \log n$. In this case the network is almost a tree and this explains why the convergence time is very close to our bound of $\Theta(n \log n)$ shown in Corollary 4.4.2. Interestingly, we have for both move policies that the convergence time decreases as the budget k increases. Intuitively this is not surprising, since having more edges leads to a lower distance-cost for an agent and to a lower diameter in the network. However, agents may have many other agents in maximum distance and the respective shortest paths may not overlap. It is not clear how a higher budget can help in such a situation.

The curves for $k \geq 3$ have a rather strange shape. They show a local peak right after the beginning and then they fall and much later rise again. We cannot explain this curious behavior but we conjecture that the reason is the density and the distance-distribution in the initial networks.

Further Remarks We emphasize again that among all these simulations we have never encountered a cyclic instance. Our cyclic constructions basically rely on the observation that it can happen that an improving move of one agent increases the costs of several agents simultaneously. Empirically this happens very often but has no severe consequences since whenever the other agents are allowed to move, they can compensate for their temporary cost increase. As we have shown, it requires a rather intricate sequence of such moves to achieve that the costs of all agents after several steps are the same as before. Such sequences are therefore very unlikely to occur.

4.5. Dynamics in (Greedy) Buy Games

We focus on the dynamic behavior of the Buy Game and the Greedy Buy Game. Remember that we assume that each edge can be created for the cost of $\alpha > 0$. Thus, the cost of an agent u with strategy $S_u \subseteq V \setminus \{u\}$ in network (G, α) is $c_u(G, \alpha) = \alpha|S_u| + \sum_{w \in V(G)} d_G(u, w)$ in the SUM-version and $c_u(G, \alpha) = \alpha|S_u| + \max_{w \in V(G)} d_G(u, w)$ in the MAX-version.

4.5.1. Convergence Results

We show that best response cycles exist, even if arbitrary strategy-changes are allowed. However, on the positive side, we were not able to construct best response cycles where only one agent is unhappy in every step. Hence, the right move policy may have a substantial impact in (Greedy) Buy Games. In contrast to this, we rule out this glimmer of hope if played on a non-complete host-graph.

Theorem 4.5.1 *The* SUM*-(G)BG and the* MAX*-(G)BG admit best response cycles.*

PROOF (OF THEOREM 4.5.1, SUM-VERSION) We prove both statements by giving a best response cycle, where the best response of any moving agent consists of either buying, deleting or swapping one edge.

The best response cycle $(G_1, \alpha), \ldots, (G_6, \alpha)$, for $7 < \alpha < 8$, is depicted in Fig. 4.13.

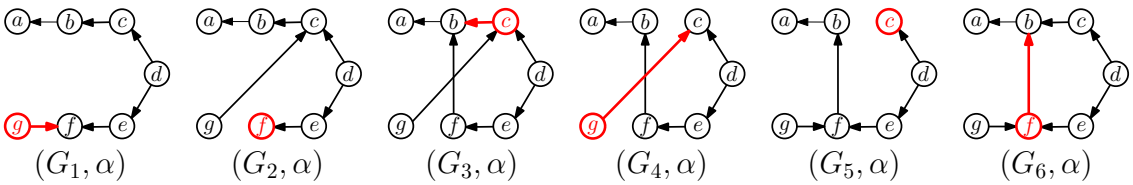


Figure 4.13.: The steps of a best response cycle for the SUM-(G)BG for $7 < \alpha < 8$.

We analyze the steps of the cycle and show that the indicated strategy-change is indeed a best response move – even if there are no restrictions on the admissible strategies.

In network (G_1, α), it is obvious that agent g is unhappy with her situation. The indicated swap gf to gc decreases agent g's cost from $\alpha + 21$ to $\alpha + 15$. This is a best possible move, which can be seen as follows. Clearly, deleting her unique own edge would disconnect the network. Hence in all optimal strategies agent g must purchase at least one edge. Among all strategies, where g buys exactly one edge, that is, among all possible single edge-swaps, we have that buying an edge towards a vertex having minimum cost in $G_1 - g$ is optimal. Here, $G_1 - g$ is the network G_1 with vertex g removed. Thus, swapping her edge towards vertex c is optimal. Buying exactly $1 < k \leq 6$ edges yields cost of at least $k\alpha + k + 2(6 - k) > 6k + 12 \geq 24$,

which is no improvement since $\alpha + 15 < 23$. After agent g has performed her strategy-change we obtain network (G_2, α).

In (G_2, α) we claim that agent f is unhappy and that her best possible move is to buy an edge towards vertex b. First of all, this is an improving move, since the edge fb decreases her cost from 19 to $11 + \alpha$, which is a strict cost decrease since $\alpha < 8$. The target vertex b is optimal, since connecting to c yields the same cost and connecting to any other vertex yields a higher cost. Clearly, agent f cannot delete or swap any edges. Furthermore, buying at least two edges yields cost of more than 19, since $2\alpha > 14$ and there are six other vertices in (G_2, α) to which f must have distance of at least 1. The edge purchase of agent f leads us to network (G_3, α).

In network (G_3, α) we claim that agent c is unhappy and that her best possible move is to delete her edge towards b. Agent c has cost $9 + \alpha$ in G_3. Deleting edge cb yields cost $16 < 9 + \alpha$, since $\alpha > 7$. Clearly, no strategy which buys at least two edges can be optimal for agent c, since $6 + 2\alpha > 16$. On the other hand, swapping her unique edge away from b must increase agent c's cost since at least one distance increases to 3. If agent c deletes her edge cb, then we obtain network (G_4, α).

In (G_4, α), we have that agent g is in a similar situation as she was in (G_1, α). Agent g is again a leaf-vertex of a path of length 6. Thus, by an analogous argument as for agent g in (G_1, α), we have that the swap gc to gf is a best possible move for agent g in (G_4, α). This move leads us to network (G_5, α).

In network (G_5, α) we have that agent c is in a similar situation as agent f in (G_2, α). By analogous arguments, it follows that buying the edge towards b is a best possible move of agent c in (G_5, α). This edge-purchase transforms (G_5, α) into (G_6, α).

Finally, in network (G_6, α) we have that agent f is in a similar situation as agent c in (G_3, α). Thus, by analogous arguments, we have that deleting her edge fb is an optimal move for agent f in (G_6, α). This deleting transforms (G_6, α) into (G_1, α) and we have completed the cycle. ∎

Proof (of Theorem 4.5.1, Max-version) We provide a best response cycle, where in every step of the cycle the moving agent has a best response which consists of a greedy move, that is, the strategy-change is the addition, deletion or swap of one edge. The best response cycle $(G_1, \alpha), \ldots, (G_4, \alpha)$, for $1 < \alpha < 2$, can be seen in Fig. 4.14.

We show for each step of the cycle that the indicated moving agent indeed performs a best response move which transforms the network of one step into the network of the next step of the cycle.

In network (G_1, α) we claim that agent g is unhappy and that a best response move is to buy the edge ga. Clearly, agent g cannot delete or swap any edges. Hence, it suffices to analyze all buy or multi-buy operations. Agent g has cost 5 in (G_1, α). The purchase of edge ga is an improving move, since with this yields a distance-cost of 3 for agent g and since $\alpha < 2$. Note that agent g must buy at least one edge to reduce her distance-cost in (G_1, α). Furthermore, it is easy to see that

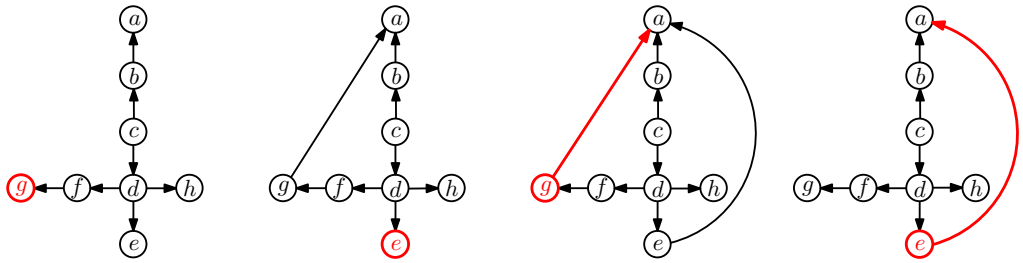


Figure 4.14.: The steps of a best response cycle for the MAX-(G)BG for $1 < \alpha < 2$.

with one additional edge a distance-cost of 3 is best possible. Now, observe that if agent g buys more than one edge, than her distance-cost may decrease, but it cannot decrease by more than 1 per edge. Since $\alpha > 1$, no strategy which buys at least two edges can yield strictly less cost than $3 + \alpha$. The indicated move of agent g transforms (G_1, α) into (G_2, α).

In (G_2, α), agent e, having cost 4, is unhappy with her situation. By buying the edge ea, agent e can decrease her distance-cost to 2. Since $\alpha < 2$, it follows that this is an improving move. Note that agent e cannot delete or swap any edges and that distance-cost of 2 is optimal, unless agent e buys 5 edges, which clearly is too expensive. Hence, buying the edge ea is a best response move for agent e and this move leads to network (G_3, α).

In network (G_3, α), we have that agent g, having cost $3 + \alpha$, is unhappy. By deleting her own edge ga, agent g can achieve a cost of 4, which is strictly less than $3 + \alpha$, since $\alpha > 1$. We claim that deleting edge ga is a best response move for agent g. If agent g swaps her unique own edge, than she cannot achieve a distance-cost of less than 3. Thus, no swap can be an improving move. If agent g buys at least one additional edge, then such a purchase may decrease agent g's distance-cost by 1 per edge, but since $\alpha > 1$, this cannot outperform her current strategy in (G_3, α). Thus, deleting edge ga is the only improving move of agent g and, thus, must be her best response move. This move transforms network (G_3, α) in to (G_4, α).

Finally, in network (G_4, α) we have that agent e, having cost $3 + \alpha$, is unhappy. Deleting her edge ea yields a cost of 4, which is strictly less than $3 + \alpha$. Swapping this edge cannot decrease her distance-cost below 3, which rules out any edge-swaps. If agent e buys at least one additional edge, then she can reduce her distance-cost by at most 1 per additional edge. Clearly, no such strategy may yield less cost than $3 + \alpha$ and, thus, cannot be an improving move. Hence, deleting her edge ea is her unique improving move, which must be her best response move. This move transforms (G_4, α) into (G_1, α). ∎

If we restrict the set of edges which can be built, then we get the worst possible dynamic behavior. In this case there is no hope for convergence if agents are only willing to perform improving moves.

Corollary 4.5.2 *The* Sum*-(G)BG and the* Max*-(G)BG on general host graphs is not weakly acyclic.*

Proof (of Corollary 4.5.2, Sum-version) We use the best response cycle $(G_1, \alpha), \ldots, (G_6, \alpha)$, shown in Fig. 4.13 with $7 < \alpha < 8$ and set the host-graph H to the graph G_1 augmented by the two additional edges bf and cg. With this host graph, we have that in every step of the cycle exactly one agent is unhappy and this agent has exactly one possible improving move. It follows that starting with network (G_1, α) on host-graph H there is no sequence of improving moves which leads to a stable network. ∎

Proof (of Corollary 4.5.2, Max-version) We use the best response cycle $(G_1, \alpha), \ldots, (G_4, \alpha)$ in Fig. 4.14 with $1 < \alpha < 2$ and we let the host-graph H be the graph G_1 with the additional edges ag and ae. It follows that in every step of the cycle, exactly one agent is unhappy and this agent has exactly one improving move. ∎

4.5.2. Empirical Study of Greedy Buy Games

We give empirical results for the convergence time for both versions of the GBG. Note that a best response for both versions of the GBG can be computed in polynomial time, whereas this problem is well-known [FLM+03, MS13] to be NP-hard for the BG. Our results show a remarkably small number of steps needed for convergence in these games, which indicates that distributed local search is a practical method for selfishly creating stable networks. For the Sum-GBG no run took longer than $7n$ steps to converge, whereas for the Max-version we always observed less than $8n$ steps until convergence. Moreover, as in the simulations for the ASG, despite several millions of trials we did not encounter a cyclic instance. This indicates that such instances are rather pathological and may never show up in practice.

As for our experiments in the ASG, we will first describe the experimental setup and then we summarize the obtained results for the convergence time of the Sum-GBG and the Max-GBG.

Experimental Setup

Our simulations for the GBG have a similar setup as the experiments presented in Section 4.4.4. One run of our simulations consists of the generation of a random initial network, then we employ the max cost or the random policy and enforce best responses until the process converges to a stable network. See Section 4.4.4 for a description of these move policies. We measure the number of steps needed for this convergence to happen.

We focus on the GBG, since it is well-known [FLM+03] that in the BG computing a best response for an agent is NP-hard. In contrast, for the GBG this can be done

in polynomial time by simply checking the best possible edge-deletion, edge-swap and edge-addition and re-computing the incurred cost, see Section 3.2. If at least two of these operations yield the same cost decrease, then we prefer deletions before swaps before additions. Such ties are very rare and we have obtained similar results by changing this preference order.

The initial networks are generated analogously to the ASG, but this time we do not have to enforce the budget-constraint. Starting from an empty graph on n vertices we first generate a random spanning tree to enforce connectedness of our networks. Then we insert edges uniformly at random until the desired number of edges is present. Note that we do not allow multi-edges. The ownership of every edge is chosen uniformly at random among the endpoints. In order to investigate the impact of the density of the initial network on the convergence time, we fix the number of edges in the initial network to be n, $2n$ and $4n$, respectively. The impact of the edge-cost parameter α is investigated by setting α to $n/10$, $n/4$, $n/2$ and n, respectively. Demaine et al. [DHMZ12] argue that this is the most interesting range for α, since implies that the average distance is roughly on par with the creation cost of an edge.

The used Python script can be found in Appendix A.2.

Experimental Results and Discussion

We have simulated both the Sum-GBG and the Max-GBG. For each configuration we computed 5000 runs. Here a configuration is determined by the number of agents, the number of edges m in the initial network, the choice for α and the choice for the move policy. For sake of presentation, our plots only contain the results for $m = n$ and $m = 4n$ and $\alpha = n$, $\alpha = n/4$ and $\alpha = n/10$.

Results for the Sum-GBG Our results for the Sum-GBG can be found in Fig. 4.15. It can be seen that the convergence time grows roughly linear in n for all configurations, which implies that these processes scale very well.

Similarly as for the ASG in the Sum-version, we have that the max cost policy outperforms the random policy. However, since the respective curves look similar, there seems to be no qualitative difference between both policies. For the cases where the initial network has $4n$ edges we have that the gap between the max cost policy and the random policy is even smaller. The cases where $m = 2n$, which are not shown here, fit in well with this observation. This can be explained as in the ASG: The max cost policy favors agents "at the perimeter" whose best response moves decrease a high number of individual distances if these moves turn out to be swaps or additions. In contrast, the random policy picks agents which occupy a more central position in the network with higher probability. If such agents happen to swap or add an edge, then this decreases a smaller number of individual distances. This also explains why the gap becomes smaller, when the number of edges in the

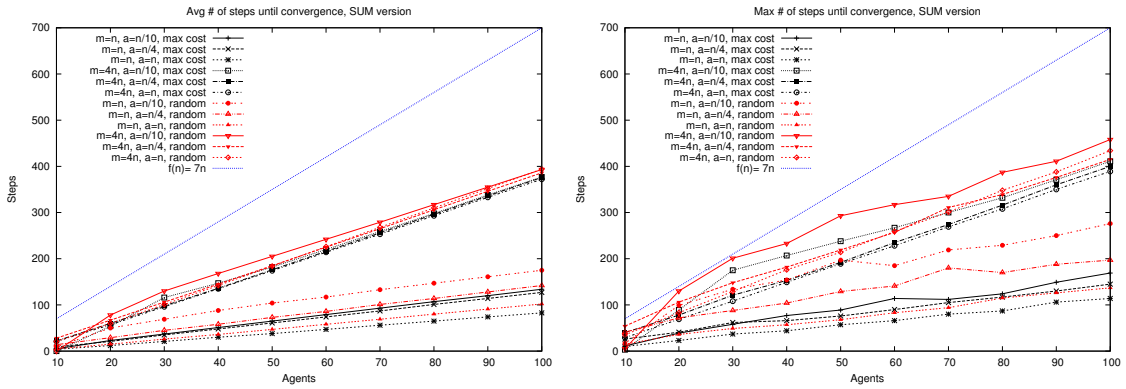


Figure 4.15.: Experimental results for the Sum-GBG. The average number of steps needed for convergence is plotted on the left, the maximum number of steps needed for convergence is plotted on the right. Each point is the average/maximum over the number of steps needed for convergence of 5000 trials with random initial networks having m edges and $\alpha = a$.

initial network becomes larger. With more edges to start with, it follows that the max cost agents have a more central position.

Interestingly, the number of edges in the initial network seems to have an impact on the convergence time, since all curves for $m = 4n$ are well above the respective curves for $m = n$. In the cases where $m = 2n$, we have that the respective curves lie between the shown curves. The reason for this may be the relatively high value for α compared to the diameter of the resulting networks. We have not found any stable network having a diameter larger than 4 and for our values of α almost all stable networks happened to be stars. Since stars have $n - 1$ edges, clearly $m - (n - 1)$ deletion-steps happen during the convergence process. However, note that the convergence time is well above $m-(n-1)$, which shows that swap and add operations also occur fairly often. A typical sample trajectory for the case of $m = 4n$ with $\alpha = n/4$ under the random policy looks as follows: First there is a phase with mostly deletions. Then this phase is followed by a phase where mostly swap and some buy and deletion operations occur. This is followed by a final phase where some swaps and mostly deletions happen. The first phase seems to be due to the fact that by our generation process the number of owned edges in the initial network varies more or less strongly between all agents and that α is relatively high. Agents having too many edges first try to get rid of them until their cost is in a range, where swaps or additions can outperform deletions, that is, where the distance-cost starts to dominate the edge-cost of that agent. The latter seems to explain what happens in the second phase. Here agents are mainly trying to minimize their distance-cost. After that, a high number of individual distances has decreased, which explains why in the last phase some swaps and a lot of deletions occur. Clearly, with less edges to start with, the second phase starts much earlier and the last phase is shorter as

well. Under the max cost policy we observed that the first phase and the second phase are slightly shorter than under the random policy. In the first phase almost all operations are deletions, whereas under the random policy we see more swaps in the first phase. Interestingly the second phase under the max cost policy consists almost entirely of swap operations. Under the random policy we see more deletions and more add-operations in the second phase. Curiously, we generally see more add-operations under the random policy than under the max cost policy. This is counter-intuitive, since at least in the second phase agents with high cost have high distance-cost and therefore add-operations should be appealing for them. However, these small observations can serve as an additional explanation why the max cost policy outperforms the random policy.

The choice of α also has an influence on the convergence time. We see that a smaller α generally yields a higher number of steps needed for convergence. Again, our additional results for $\alpha = n/2$ confirm this observation. With smaller α, we have that the length of the first phase decreases and the length of the second phase increases. It seems that agents buy more edges in the second phase which then leads to a longer last phase which may explain the overall increase in convergence time.

To clarify the influence of different starting topologies, we have compared three different types of initial networks. The results can be found in Fig. 4.16. In the `random` setting, we focus on the initial networks with n vertices and n edges as described in Section 4.5.2. In the random line, or `rl`, setting we generate the initial networks as follows: first a path having n vertices is created and then we choose the ownership of each edge uniformly at random among its endpoints. In the directed line, or `dl`, setting we generate a path having n vertices and then the edge ownership is chosen such that all edges point in the same direction, that is, that the edge-ownership forms a directed path.

Generally we find that the specific topology of the initial network has only marginal impact on the number of steps needed for convergence in the Sum-GBG. As shown in Fig. 4.16, the convergence time differs roughly by a factor of at most 2. We expected that the `random` setting is faster than the `rl` setting and that `rl` is faster than `dl`. The reason for our expectation was that the initial networks in the `random` setting are more star-like than in `rl` or `dl` (which actually is the opposite of star-like) and being star-like seems to be closer to the typical shape of a stable network. However, our simulations show exactly the opposite behavior for all configurations. Under both move policies we see that `dl` is faster than `rl` and `random`. One possible explanation is that due to the high diameter of the initial network in the `dl` setting a move by some agent, especially in the first rounds, decreases the social cost by a larger amount than in the `rl` or `random` setting. Maybe this initial decrease is large enough to reduce the number of moves at the end of the process.

Another interesting point is the comparison of the two move policies. We find that the max cost policy outperforms the random policy, independently of the initial setting. But there are differences: Under the max cost policy we see that the `dl`

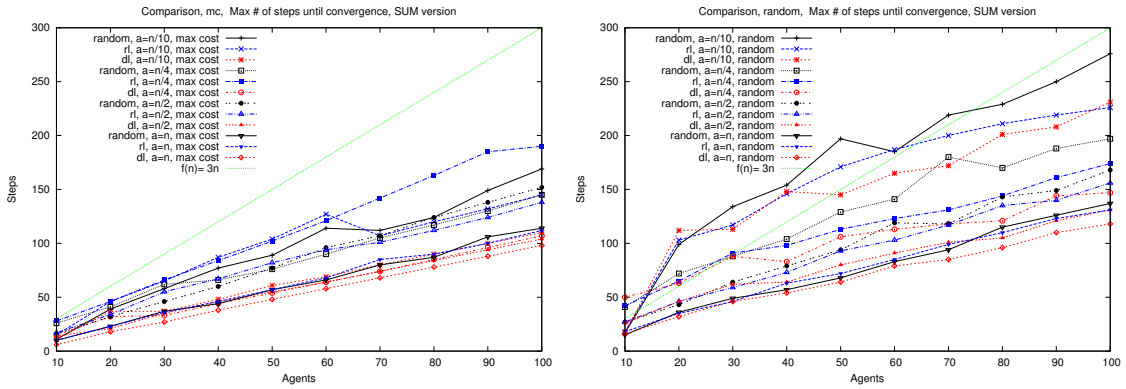


Figure 4.16.: Comparison of different starting topologies for the SUM-GBG. The maximum number of steps needed for convergence under the max cost policy is plotted on the left, the maximum number of steps needed for convergence under the random policy is plotted on the right. Each point is the maximum over the number of steps needed for convergence of 5000 trials where the initial setting is `random`, `rl` or `dl` and $\alpha = a$.

setting is the fastest, independently of the edge-price α. The convergence time in the other settings depends stronger on α and they are very similar. The reason for this may be that only a few moves by max cost agents in the `rl` suffice to obtain a network which is very similar to the initial network in the `random` setting. Under the random policy, we again see for each configuration that `dl` is faster than `rl`, which is faster than `random`. However, we also see that the convergence time depends stronger on α than under the max cost policy. It seems that the value of α has a stronger influence than the respective initial setting. This is not surprising since the max cost policy favors agents whose move decreases the social cost by a rather large amount which leads to faster convergence.

Results for the Max-GBG The results of our simulations for the MAX-GBG are shown in Fig. 4.17. We generally find the same behavior as for the SUM-version. The convergence time grows linear in n but, at least for the cases where $m = 4n$ it takes longer than the respective configurations in the SUM-version.

One difference is that the choice of α seems to have less impact on the convergence time than in the SUM-version. Furthermore, for the cases where $m = 4n$ and $m = 2n$ (omitted in the plot) we see that the process under the max cost policy is slower than under the random policy. Thus, we see the opposite behavior under both move policies.

A typical sample trajectory for $m = 4n$, with $\alpha = n/4$ under the random policy looks much like the trajectory for the SUM-version, but it seems that the final phase is missing. The first phase contains almost exclusively deletions and the second phase contains mostly swaps and some deletions. Add-operations are exceptionally

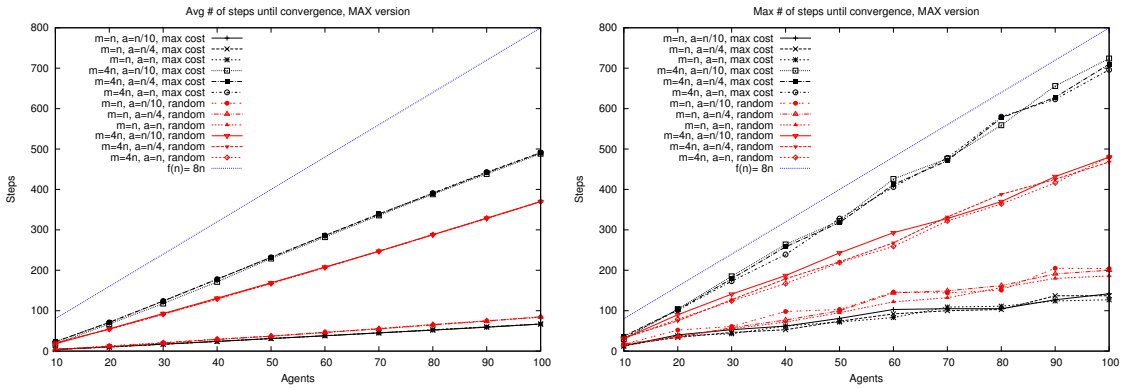


Figure 4.17.: Experimental results for the MAX-GBG. The average number of steps needed for convergence is plotted on the left, the maximum number of steps needed for convergence is plotted on the right. Each point is the average/maximum over the number of steps needed for convergence of 5000 trials with random initial networks having m edges and $\alpha = a$.

rare, which seems to be due to the high value of α. Interestingly, under the max cost policy a typical trajectory looks different. Here we have that the second phase is much longer and almost all moves in this phase are swaps. This may explain why the max cost policy induces a slower convergence towards a stable network. Clearly, in the MAX-version the cost of an agent is mostly determined by her edge-cost, which explains why first a series of deletions and then mostly swap operations can be seen.

As for the SUM-version, we considered different initial topologies. Fig. 4.18 shows a comparison of the three settings `random`, `rl` and `dl`, which we have introduced above.

The comparison shows a stronger impact of the starting topology than in the SUM-version. In the MAX-version we find that the convergence times differ by at most a factor of 5. Interestingly, here we find the expected outcome that under both move policies the `random` setting is faster than `rl` and `dl` and that `rl` is faster than `dl`. Furthermore, the edge-price α has almost no influence on the convergence time compared to the influence of the initial topology. Both move polices yield almost the same convergence times. As argued above, this is not surprising, since we have that under the max cost policy a few moves suffice to obtain a network where a large fraction of agents has maximum cost and then we are almost in the random policy scenario.

Further Remarks As for the ASG, we have not encountered a cyclic instance for the SUM-GBG or the MAX-GBG in any of our several million trials. This indicates that non-convergent initial networks may be very rare. It is not impossible that the random policy or even the max cost policy may guarantee convergence. However, as

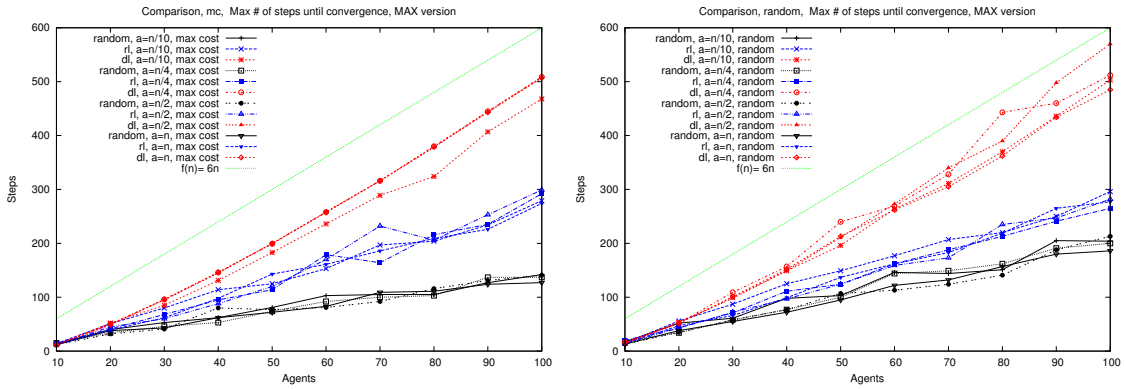


Figure 4.18.: Comparison of different starting topologies for the MAX-GBG. The maximum number of steps needed for convergence under the max cost policy is plotted on the left, the maximum number of steps needed for convergence under the random policy is plotted on the right. Each point is the maximum over the number of steps needed for convergence of 5000 trials where the initial setting is `random`, `rl` or `dl` and $\alpha = a$.

our result for the GBG on general host-graphs shows, there may be initial networks on the complete host-graph for which no sequence of improving moves leads to a stable network. Finding such initial networks seems to be very challenging, but - in contrast - proving guaranteed convergence for some move policy seems even more challenging.

4.6. Dynamics in Bilateral Buy Games with Cost-Sharing

We consider "bilateral network formation", as introduced by Corbo and Parkes [CP05], which we call the *bilateral equal-split BG*. This version explicitly models that bilateral consent is needed in order to create an edge, which is a realistic assumption in some settings. The cost of an edge is split equally among its endpoints and edges are built only if *both* incident agents are willing to pay half of the edge-price. This model implicitly assumes coordination among coalitions of size two and the corresponding solution concept is therefore the pairwise Nash Equilibrium, which can be understood as the minimal coalitional refinement of the pure Nash Equilibrium. The authors of [CP05] show that this solution concept is equivalent to Myerson's proper equilibrium [Mye91], which implies guaranteed convergence if the agents repeatedly play best response strategies against *perturbations* of the other players' strategies, where costly mistakes are made with less probability. We show in this section that these perturbations are necessary for achieving convergence by proving that the bilateral equal-split BG is not weakly acyclic in the SUM-version

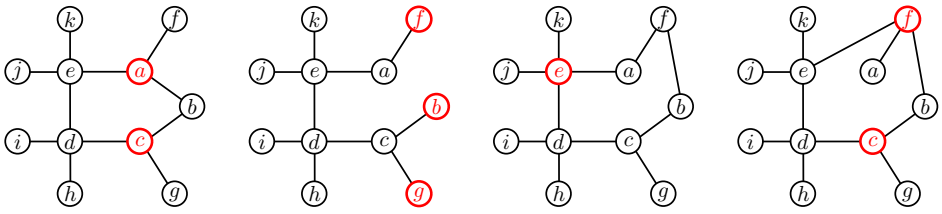
and that it admits best response cycles in the MAX-version. Interestingly, the first result is stronger than the result for the SUM-(G)BG, which yields the counter-intuitive observation that sharing the cost of edges can lead to worse dynamic behavior.

We want to avoid that agents are forced to buy edges. Hence, we assume that the initial network of the network creation process is connected. This assumption may be removed by enforcing a penalty for every disconnected agent, as proposed by Fabrikant et al. [FLM+03] and analyzed by Brandes et al. [BHN08].

The creation of an edge in bilateral equal-split BGs requires coordination of the two incident agents, whereas the deletion of an edge is a unilateral move of one agent. Formally, the edge-cost term in the cost function of an agent changes slightly, since in this version an agent has to pay half of the price for *every* incident edge. Let (G, α) be any network and let $N_G(u)$ be the set of neighbors of agent u in (G, α). Let (G', α) be the network induced by a strategy-change of agent u in network (G, α) and let $N_{G'}(u)$ be the set of agent u's neighbors in (G', α). Thus, agent u changes her strategy from $N_G(u)$ to $N_{G'}(u)$. Such a strategy-change is *feasible* if and only if $c_v(G, \alpha) \geq c_v(G', \alpha)$ for all $v \in N_{G'}(u) \setminus N_G(u)$. Hence, agent u can perform a move from strategy S_u to S'_u if and only if in the network induced by S'_u *all* agents involved in the creation of *new* edges selfishly agree to pay their cost-share. If this is not the case for some agent $x \in S'_u \setminus S_u$, then we say that agent x *blocks* agent u's move from S_u to S'_u.

Theorem 4.6.1 *The* SUM *bilateral equal-split Buy Game is not weakly acyclic.*

PROOF Theorem 4.6.1 We prove the statement by giving a cyclic sequence of networks $(G_0, \alpha), \ldots, (G_2, \alpha)$, where all unhappy agents in network (G_i, α) only have feasible improving moves which lead to a network which is isomorphic to network $(G_{i+1 \bmod 3}, \alpha)$. It follows that starting from network (G_0, α) as initial network *no* sequence of improving moves leads to a network where all agents are happy.



(G_0, α) (G_1, α) (G_2, α) $(G_3, \alpha) \simeq (G_0, \alpha)$

Figure 4.19.: The cyclic sequence of networks $(G_0, \alpha), (G_1, \alpha), (G_2, \alpha)$ for $10 < \alpha < 12$ for the SUM bilateral equal-split Buy Game. Unhappy agents are highlighted.

Now we analyze the states of the cycle. The networks $(G_0, \alpha), (G_1, \alpha)$ and (G_2, α) are shown in Fig. 4.19. We assume that $10 < \alpha < 12$ holds.

Network (G_0, α)**:** We show that only agent a and c are unhappy in (G_0, α) and that their unique improving move is the removal of their edge towards agent b, respectively. After this move, we obtain a network which is isomorphic to network (G_1, α).

Consider the leaf agents f, g, h, i, j and k. Clearly, none of them can delete an edge since this would disconnect the network. Next, swapping their unique edge involves another agent who must be willing to accept another leaf neighbor, but, since $\frac{\alpha}{2} > 1$, no agent will do this. It follows that a leaf agent must buy at least two edges if she wants to improve. Among those edges, there cannot be edges towards other leaf agents. This is true because the network (G_0, α) has diameter 4 and no vertex has more than two neighboring leaves. At best, such an edge can decrease the distance towards the other leaf by 3, towards the neighbor of this leaf by 1 and possibly towards the other leaf connected to this neighbor by 1, which yields at most a distance decrease of $5 < \frac{\alpha}{2}$. All non-leaf agents have distance at most 3 towards any leaf agent and distance at most 2 towards any non-leaf agent. It follows that no non-leaf vertex will accept any edge offered by a leaf agent, independently of which other edges this leaf agent buys. This implies that no leaf agent can perform any strategy-change to decrease her cost.

Now we focus on agent d and e. By symmetry, both agents face the same situation and therefore we restrict our attention to agent d. Her cost in (G_0, α) is $4\frac{\alpha}{2} + 17$. Clearly, agent d cannot remove her edge towards h or i and not both edges to c and e, since this would disconnect the network. Removing edge dc or de alone increases her distance-cost by 7 or 14, respectively. Since $\frac{\alpha}{2} < 7$, these moves are not improving for agent d. Thus, agent d's best possible strategy buys at least three edges including the edges dh and di. Among all strategies of agent d which buy exactly three edges, the strategy $\{a, h, i\}$ is optimal, since a has minimum distance-cost in the network $G_0 - \{d, i, h\}$. Note that the strategy $\{a, h, i\}$ is better than agent d's current strategy $\{c, e, h, i\}$ and it is the only strategy involving three edges with this property. However, the move from $\{c, e, h, i\}$ to $\{a, h, i\}$ will be blocked by agent a, since she currently has cost $3\frac{\alpha}{2} + 20$ and with agent d's new strategy her cost increases to $4\frac{\alpha}{2} + 17$. Agent d's current strategy is best possible among all strategies which buy four edges. This is easy to see, since the edges dh and di are forced and the other two edges should connect to the vertices of a 2-median-set in the graph $G_0 - \{d, h, i\}$. There are two such sets: $\{c, e\}$ and $\{b, e\}$. Thus, the strategy $\{b, e, h, i\}$ yields the same cost as strategy $\{c, e, h, i\}$ for agent d. We claim that no strategy buying more than four edges can outperform agent d's current strategy. Any such strategy buys at least five edges and no edge will connect to a leaf of (G_0, α). The strategy $\{a, c, e, h, i\}$ is the best strategy using five edges and yields cost $5\frac{\alpha}{2} + 15$. With six edges there is only one strategy which yields cost of $6\frac{\alpha}{2} + 14$. Thus, agents d and e cannot perform an improving move in (G_0, α).

Next, we show that agent b is happy in network (G_0, α). Clearly, agent b has to buy at least one edge, since otherwise the network is disconnected. Among all these strategies, the strategies $\{d\}$ and $\{e\}$, which both yield cost $\frac{\alpha}{2} + 25$, are optimal.

Note that both outperform agent b's current strategy $\{a, c\}$ having cost $2\frac{\alpha}{2} + 22$. However, in both cases, agent b cannot change towards the new strategy, since the respective new neighbor will block this change. Consider b's strategy-change from $\{a, c\}$ to $\{d\}$. Before the change, agent d has cost $4\frac{\alpha}{2} + 17$, after the change d's cost is $5\frac{\alpha}{2} + 16$, which is strictly larger. Note that no other strategy of agent b which buys exactly one edge outperforms her current strategy. By deleting her edge to a or c her distance-cost increases by 11. Furthermore, connecting to a leaf is clearly worse than agent b's current strategy. By the same reasoning as above, agents d and e will block every new strategy of b which connects to them. Since connecting to a leaf is clearly sub-optimal, we are left with b's current strategy $\{a, c\}$. Thus, agent b cannot perform an improving move.

Agents a and c are unhappy and, by symmetry, we will focus on agent a, who has cost $3\frac{\alpha}{2} + 20$ in (G_0, α). Clearly, all optimal strategies of agent a must buy the edge towards f and at least one additional edge. The best possible strategy using two edges is $\{d, f\}$, since d is a 1-median vertex of the graph $G_0 - \{a, f\}$. But, agent e would block this strategy, since this strategy-change yields cost $5\frac{\alpha}{2} + 15 > 4\frac{\alpha}{2} + 17$ for agent e. There are two other strategies using two edges and which outperform agent a's current strategy. These strategies are $\{c, f\}$ and $\{e, f\}$. Moving from $\{b, e, f\}$ to $\{c, f\}$ is not possible for agent a, since this move would be blocked by agent c, whose cost would change from $3\frac{\alpha}{2} + 20$ to $4\frac{\alpha}{2} + 18$. The move from $\{b, e, f\}$ to $\{e, f\}$ is possible, since the move only consists of the deletion of an edge, which is a unilateral move. This move decreases agent a's cost from $3\frac{\alpha}{2} + 20$ to $2\frac{\alpha}{2} + 25$, which is indeed an improvement since $\frac{\alpha}{2} > 5$. Now we are left to show that agent a has no other improving moves. Clearly, since connecting towards leaf agents cannot outperform a's current strategy, we can ignore all other strategies which buy two edges. For all larger strategies the same holds true. We have already shown that agent c blocks any connection attempts of agent a. By analogous reasoning, the same is true for agents d and e. It follows that there are no other possible improving moves for agent a. Hence, we have that a's move from $\{b, e, f\}$ to $\{e, f\}$ is her only improving move. By symmetry the same holds for agent c's move from $\{b, d, g\}$ to $\{d, g\}$. In both cases we obtain a network which is isomorphic to network (G_1, α).

Network (G_1, α)**:** We show that in network (G_1, α) only the agents b, f, g are unhappy and that any improving move of one of those agents leads to a network which is isomorphic to network (G_2, α).

Analogous to the discussion above, we have that no non-leaf agent is willing to accept an edge from agents h, i, j or k and that any edge towards another leaf agent does not yield a distance decrease which is high enough to compensate the edge-price. Hence, these agents cannot perform a strategy-change.

Agents d and e are in a very similar situation than in (G_0, α) and, since they cannot buy less edges, it is easy to see that they cannot perform an improving move. Agent a is happy, since she has just performed her only improving move which transformed network (G_0, α) into network (G_1, α). If a would have another improving move, then this would contradict the uniqueness of her move in (G_0, α).

Agent c cannot delete her edges to b and g and has to buy at least three edges. Her best possible strategy using three edges would connect to e instead of d, but c cannot move towards this strategy, since agent e would block it. Connecting to a instead of d is clearly worse. Any strategy which uses four edges cannot outperform agent c current strategy. Since agent e refuses an edge from c, there are only agent a or leaf agents left. Again, connecting to leaf agents yields a cost increase. Connecting to agent a yields a decrease in distance-cost of 4 but, since $\frac{\alpha}{2} > 4$, it follows that the strategy $\{a, b, d, g\}$ is worse than agent c's current strategy. More than 4 edges would involve an edge towards a leaf agent and therefore we conclude that agent c cannot improve on her current strategy $\{b, d, g\}$.

We are left with agents b, f and g. Agents b and g face a similar situation an we will focus on agent b having cost $\frac{\alpha}{2} + 33$. Clearly, agent b has to buy at least one edge and, among all strategies using exactly one edge, agent b's current strategy $\{c\}$ is the only possible strategy, since all other vertices would block an edge-swap from agent b to them. Hence, we consider possible strategies of agent b, which buy at least two edges. Agents d and e will refuse any edge from b, independently of the number of other edges bought by agent b. Thus, d and e cannot be involved in any feasible improving strategy of agent b. Next, by buying an edge towards h, i, j or k agent b cannot decrease her distance-cost more than $4 < \frac{\alpha}{2}$, which rules out that h, i, j or k are involved in an improving strategy. We are left with vertices a, c, f and g as possible targets for edges from b and we have that b's strategy has to choose at least two of them. With these restrictions it is easy to see that buying at least three edges is too expensive for agent b. Hence, we focus on all possible strategies using exactly two edges. Clearly, strategy $\{a, c\}$ is the best possible among them, but it requires that agents a and c are willing to accept b's edge. For agent c this is true, since in (G_1, α) this edge is already present and because bc is a bridge. Unfortunately for b, agent a will block the move from $\{c\}$ to $\{a, c\}$, since a was the agent who unilaterally decided to remove the edge towards b in network (G_0, α). The same is true for a move towards strategy $\{a, g\}$. We are left with the two strategies $\{c, f\}$ and $\{f, g\}$. After moving to strategy $\{f, g\}$ agent b has cost $2\frac{\alpha}{2} + 28$, which, since $\frac{\alpha}{2} > 5$, is higher than agent b's current cost of $\frac{\alpha}{2} + 33$. Finally, the move towards strategy $\{c, f\}$ will be an improving move, since this strategy yields cost $2\frac{\alpha}{2} + 25 < \frac{\alpha}{2} + 33$. Furthermore, the move from $\{c\}$ to $\{c, f\}$ will not be blocked by agent f, since this move decreases agent f's cost from $\frac{\alpha}{2} + 34$ to $2\frac{\alpha}{2} + 26$. Moreover, the move will not be blocked by agent c, since the move strictly decreases agent c's cost. Thus, we have that the move from $\{c\}$ to $\{c, f\}$ is agent b's unique feasible improving move. By symmetry, the same is true for agent g's move from strategy $\{c\}$ to strategy $\{c, f\}$.

Now we consider agent f. By analogous reasoning there is no feasible improving move for agent f towards a strategy which uses exactly one or more than two edges. We are left with analyzing all possible strategies using exactly two edges. Similar to the situation of agent b, only the agents a, b, c and g are possible targets for f's edges. The strategy $\{a, c\}$ looks the most promising for agent f, but it will be

blocked by agent c, since c's decrease in distance-cost caused by this move is only $4 < \frac{\alpha}{2}$. The same is true for strategy $\{b, c\}$ or $\{g, c\}$. Moreover, strategy $\{b, g\}$ yields higher cost then f's current strategy. Hence, we are left with the strategies $\{a, b\}$ and $\{a, g\}$. Both yield cost $2\frac{\alpha}{2} + 26 < \frac{\alpha}{2} + 33$, since $\frac{\alpha}{2} < 7$. A move from $\{a\}$ to $\{a, b\}$ or $\{a, g\}$ is feasible, since agent b's (or g's) cost changes from $\frac{\alpha}{2} + 31$ to $2\frac{\alpha}{2} + 25$, which is a strict cost decrease since $\frac{\alpha}{2} < 6$. Furthermore, it is easy to see that agent a's cost strictly decreases if f moves to strategy $\{a, b\}$ or $\{a, g\}$, which implies that a will not block such a move.

Observe that all possible improving moves by agents b, f or g lead to a network which is isomorphic to (G_2, α).

Network (G_2, α)**:** We show that in network (G_2, α) only agent e is unhappy and that her unique feasible improving move leads to a network which is isomorphic to network (G_0, α).

We consider the leaf agents g, h, i, j and k first. Clearly, they cannot delete their unique edge and no agent would accept an edge if a leaf agent performs an edge-swap. Thus, they must buy at least two edges if they want to outperform their current strategy. Analogous to the discussion above, no edge towards another leaf agent can be part of an improving strategy. Thus, it remains to show that no strategy-change towards a combination of at least two edges to non-leaf agents is a feasible improving move for g, h, i, j or k. The agents d and e will not accept any new edge from a leaf agent since they have only one non-leaf agent in distance 3 and any leaf agent is in distance at most 3. It follows that by accepting such an edge agent d or e can only hope for a distance decrease of $3 < \frac{\alpha}{2}$. The same is true for agents a and c. Agent a has one non-leaf agent in distance 3 and one leaf-agent, which is g, in distance 4. Thus, by accepting an edge coming from a leaf agent, agent a's distance-cost can decrease by at most $4 < \frac{\alpha}{2}$. For agent c the situation is similar, but c does not have a leaf agent in distance 4, which implies that by accepting an edge from a leaf agent only a distance decrease of 3 is possible. We are left with agents b and f, which both have one non-leaf agent in distance 3 and two leaf agents in distance 4. Hence, by accepting an edge from a leaf agent they can possibly reduce their distance towards this leaf by 3, towards its neighbor by 1 and towards the other leaf in distance 4 by 1. In total the best possible distance decrease for agents b or f is $5 < \frac{\alpha}{2}$. It follows that no leaf agent can perform a feasible improving move.

Next, we show that agent d cannot improve on her current strategy $\{c, e, h, i\}$ which yields cost of $4\frac{\alpha}{2} + 17$. Clearly, agent d must buy the edges towards h and i and at least one more edge to connect to the network $G_2 - \{d, h, i\}$. The best possible strategy using three edges connects to a 1-median vertex of $G_2 - \{d, h, i\}$. There are two such vertices: a and f. Both strategies $\{a, h, i\}$ and $\{f, h, i\}$ yield cost $3\frac{\alpha}{2} + 25$ which is higher than agent d's current cost. Hence, no strategy using exactly three edges can outperform d's current strategy. The optimal strategy using 4 edges must connect towards the vertices which form a 2-median-set in the graph $G_2 - \{d, h, i\}$. The 2-median-problem in $G_2 - \{d, h, i\}$ has two solutions: $\{c, e\}$,

$\{b, e\}$. Thus, agent d's current strategy is an optimal strategy using four edges. Now let us look at possible strategies for agent d which use more than four edges. The best strategy using five edges is $\{c, e, f, h, i\}$ and yields cost $5\frac{\alpha}{2} + 15 > 4\frac{\alpha}{2} + 17$. Clearly, the best strategy using six edges is worse. Thus, agent d cannot perform any improving move.

Now, let us consider agent c having cost $3\frac{\alpha}{2}+20$ with her current strategy $\{b, d, g\}$. Agent c must buy the edge towards g and at least one more edge to ensure connectedness of the network. Her best possible strategy using exactly two edges is $\{e, g\}$, since g is the unique 1-median vertex of $G_2 - \{c, g\}$. However, the move from $\{b, d, g\}$ towards $\{e, g\}$ will be blocked by agent e since this move increases her cost from $4\frac{\alpha}{2} + 18$ to $5\frac{\alpha}{2} + 16$. The two second best strategies using two edges are $\{a, g\}$ and $\{d, g\}$ which both yield cost $2\frac{\alpha}{2} + 26 > 3\frac{\alpha}{2} + 20$ since $\frac{\alpha}{2} < 6$. Hence, no strategy using two edges can be a feasible improving strategy for agent c. For all other strategies using more than two edges we have that they cannot contain an edge towards e. This is true since in G_2 agent e only has two agents in distance 3, which implies that by accepting an edge from c agent e can only hope for a distance decrease by $3 < \frac{\alpha}{2}$. It follows that c's best possible strategy using three edges must connect to agent d and g. The best choice for the third edge is agent f, but the move from $\{b, d, g\}$ to $\{d, f, g\}$ will be blocked by agent f since this move changes her cost from $2\frac{\alpha}{2}+26$ to $3\frac{\alpha}{2}+21$ which is a strict cost increase, since $5 < \frac{\alpha}{2}$. Clearly, the third edge cannot connect to a leaf agent of (G_2, α). Hence, we are left with c's current strategy and the strategy $\{a, d, g\}$, which yield the same cost. It follows that c's current strategy is the best possible feasible strategy using three edges. If agent c would buy more than three edges, then, since e is not available and leaf agents are not attractive as well, the best such strategy connects to d and g and chooses two targets from the set $\{a, b, f\}$. It is easy to see that such a strategy yields higher cost than c's current strategy. Moreover, if c buys more than four edges, then the situation gets even worse. Thus, we have that agent c cannot perform a feasible improving move.

Agent b has cost $2\frac{\alpha}{2} + 25$ in network (G_2, α). Agent b cannot move towards a strategy using exactly one edge, since the removal of the edge bf increases b's distance-cost by $6 > \frac{\alpha}{2}$ and removing the edge bc yields an increase in distance-cost by 16. Furthermore, no other agent than c or f would accept an edge from agent b if this edge is b's unique edge. Thus, we have that agent b has to buy at least two edges to outperform her current strategy. Note that agents d and e will refuse to accept any edge offered by agent b, since they have at most two agents in distance 3 an can only hope for a distance decrease by 3 from such an edge. By buying an edge towards a leaf agent, agent b can only hope for a distance decrease by $5 < \frac{\alpha}{2}$, since b has two agents in distance 4 and their common neighbor is in distance 3. Agent a will refuse an edge from b, since this edge must yield a distance decrease for a by at least 6 but this is only possible if b simultaneously buys edges towards c, g, h and i, which clearly is not an improving strategy for agent b. Hence, agent b only has agents c and f available as targets and connecting to both is b's current

strategy. It follows that agent b has no feasible improving move.

Agent f has cost $2\frac{\alpha}{2}+26$. We first show that f has no feasible improving strategy using one edge. Removing edge fa or fb increases f's distance-cost by 11 or 6, respectively. Hence, no such removal yields a cost decrease. Furthermore, it is easy to see that no other agent than a and b would accept an edge from f if f buys no other edges. By an analogous argument as for agent b, it follows that no edge towards a leaf agent is beneficial for f and that no non-leaf agent other than a and b would accept an edge from f. Thus, agent f cannot perform any improving strategy-change.

Next, we show that agent a, having cost $2\frac{\alpha}{2}+23$, cannot move towards an improving strategy. No move towards a strategy using one edge can be feasible and yield a cost decrease. Removing the edge ae or af yields an increase in distance-cost by 18 or 6, respectively, which implies that agent a would not improve. Moreover, no agent other than e and f would accept an edge from a if a buys no other edges. For all strategies which buy more than one edge, we have that agent d would refuse to accept any edge coming from a, since d only has one agent in distance 3 and this implies that d could only gain 2 in distance-cost. Moreover, no edge towards a leaf agent can be part of an improving strategy for a, since a has only one leaf in distance 4 and could decrease her distance-cost by at most 4 by such an edge. Hence, all edges of a must connect to vertices b, c, e or f and it is obvious that in a's best possible strategy the edge towards e should by contained. With this restriction it follows that a's best possible strategy must use exactly two edges, since the strategies $\{b, c, e\}$, $\{c, e, f\}$, $\{b, e, f\}$ and $\{b, c, e, f\}$ are clearly more expensive than a's current strategy. The strategy $\{c, e\}$ outperforms a's current strategy, but the move from $\{e, f\}$ to $\{c, e\}$ will be blocked by agent c since this move increases her cost from $3\frac{\alpha}{2}+20$ to $4\frac{\alpha}{2}+18$. Analogously, agent a's move from $\{e, f\}$ to $\{b, e\}$ will be blocked by agent b since this move increases her cost from $2\frac{\alpha}{2}+25$ to $3\frac{\alpha}{2}+21$. Thus, we have that agent a cannot perform a feasible improving move.

Finally, we show that agent e, having cost $4\frac{\alpha}{2}+18$, is unhappy in (G_2, α). Clearly, agent e must buy the edges towards j and k and at least one additional edge. Since c is the unique 1-median vertex in $G_2 - \{e, j, k\}$, we have that $\{c, j, k\}$ is agent e's best possible strategy which buys three edges. This strategy outperforms e's current strategy, but the move from $\{a, d, j, k\}$ to $\{c, j, k\}$ will be blocked by agent c since this move increases her cost from $3\frac{\alpha}{2}+20$ to $4\frac{\alpha}{2}+17$. The second best strategies using exactly three edges, $\{d, j, k\}$ and $\{b, j, k\}$, both yield cost $3\frac{\alpha}{2}+24$ for agent e. Since $\frac{\alpha}{2} < 6$, this implies that these strategies yield higher cost for agent e. It follows that no move to a strategy with three edges can be feasible and improving for agent e. The best possible strategy using four edges must connect to the vertices in a 2-median-set of $G_2 - \{e, j, k\}$. This set is $\{d, f\}$ and it is unique, hence we have that $\{d, f, j, k\}$ is agent e's best possible strategy using four edges. Note that this strategy yields cost $4\frac{\alpha}{2}+17$ which implies that it outperforms agent e's current strategy. Furthermore, the move from $\{a, d, j, k\}$ towards $\{d, f, j, k\}$ is feasible, since this move decreases agent f's cost from $2\frac{\alpha}{2}+26$ to $3\frac{\alpha}{2}+20$. This is indeed

a strict decrease, since $\frac{\alpha}{2} < 6$. We have found a feasible improving strategy-change for agent e. In the following we will show that this is the only feasible improving strategy-change. The second best strategies using exactly four edges are $\{a, d, j, k\}$, which is e's current strategy and $\{b, d, j, k\}$. Both yield the same cost for agent e. It follows that there are no other possible improving strategies using four edges. Strategies using more than four edges cannot outperform agent e's current strategy. This can be seen as follows. With e's current strategy we have that there are two agents in distance 3 and both do not share a common neighbor. The best possible situation with more than four edges would be to have five vertices in distance 1 and the rest in distance 2. This yields a cost of $5\frac{\alpha}{2} + 15 > 4\frac{\alpha}{2} + 18$. Thus, we have shown that agent e can perform exactly one feasible improving strategy-change and this move transforms network (G_2, α) into a network (labeled (G_3, α) in Fig. 4.19) which is isomorphic to (G_0, α). ∎

For the MAX-version, we can show a slightly weaker result.

Theorem 4.6.2 *The* MAX *bilateral equal-split Buy Game admits best response cycles.*

PROOF Theorem 4.6.2 The four steps $(G_1, \alpha), \ldots, (G_4, \alpha)$ of the best response cycle are depicted in Fig. 4.20. We assume that $2 < \alpha < 4$ holds.

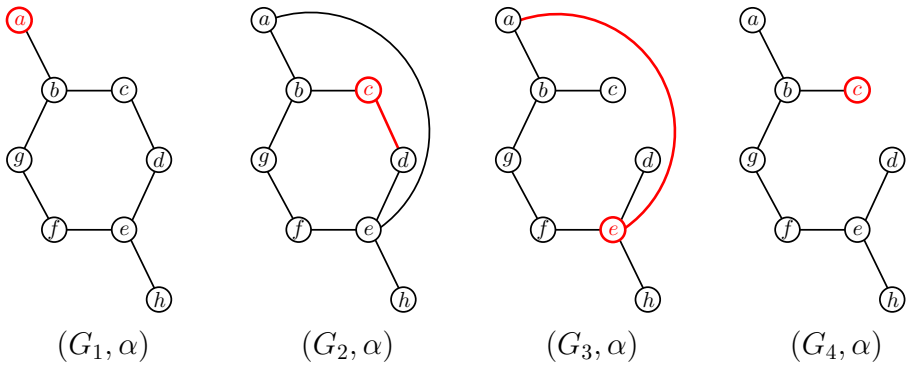


Figure 4.20.: The steps of a best response cycle for $2 < \alpha < 4$ for the MAX bilateral equal-split Buy Game.

In network (G_1, α) we claim that agent a, having cost $\frac{\alpha}{2} + 5$ is unhappy with her situation and that buying the edge ae is the best possible feasible strategy-change for her. With strategy $\{b, e\}$ agent a has cost $2\frac{\alpha}{2} + 2 < \frac{\alpha}{2} + 5$, since $\frac{\alpha}{2} < 2$. Furthermore, a's move from $\{b\}$ to $\{b, e\}$ decreases e's cost from $3\frac{\alpha}{2} + 4$ to $4\frac{\alpha}{2} + 2$.

Observe that agent a cannot remove any edge. By swapping her unique edge, agent a can possibly achieve distance-cost of 4, but we have $2\frac{\alpha}{2} + 2 < \frac{\alpha}{2} + 4$. Thus, no strategy using one edge can outperform her move from $\{b\}$ to $\{b, e\}$. Note that by buying less than seven edges, the best possible distance-cost agent a can hope for is 2. Since the strategy $\{b, e\}$ already achieves this, it follows that agent a's move

from $\{b\}$ to $\{b, e\}$ is indeed a best possible strategy-change. This move transforms network (G_1, α) into network (G_2, α).

In network (G_2, α) we claim that agent c's best possible feasible strategy-change is the removal of edge cd. By removing this edge, agent c's cost changes from $2\frac{\alpha}{2}+3$ to $\frac{\alpha}{2}+4$, which is a strict decrease since $\frac{\alpha}{2} > 1$. Among all strategies which buy one edge, the strategy $\{e\}$ would be optimal for agent c since this is the only strategy which yields distance-cost of 3. However, a move from $\{b, d\}$ to $\{e\}$ will be blocked by agent e, whose cost changes from $4\frac{\alpha}{2}+2$ to $5\frac{\alpha}{2}+2$. It follows that distance-cost of 4 is best possible if agent c buys only one edge. Among all strategies using two edges, the strategy $\{b, e\}$ is the only one which yields distance-cost 2. But, as we have already seen, agent e would block agent c's move from $\{b, d\}$ to $\{b, e\}$. It follows that c's current strategy is the best possible among all strategies which buy two edges. By buying more than two and less than seven edges, agent c can only hope for distance-cost 2, but even with three edges, this yields higher cost than her current strategy. Buying seven edges is clearly too expensive. Hence, we have that the removal of edge cd is a feasible and best possible strategy-change for agent c and this change transforms (G_2, α) into (G_3, α).

Agent e is unhappy in network (G_3, α). We show that e's best possible feasible strategy-change is the removal of edge ea. This move decreases agent e's cost from $4\frac{\alpha}{2}+3$ to $3\frac{\alpha}{2}+4$, which is a strict decrease since $1 < \frac{\alpha}{2}$. In any improving strategy agent e must buy an edge towards d and h and at least one additional edge. Clearly, agent e's best possible strategies using three edges is $\{b, d, h\}$ and $\{d, g, h\}$, since b and g is are the 1-center vertices of $G_3 - \{d, e, h\}$. Both strategies outperform agent e's removal of ea, but a move from $\{a, d, f, h\}$ to $\{b, d, h\}$ will be blocked by agent b whose cost will be increased from $3\frac{\alpha}{2}+3$ to $4\frac{\alpha}{2}+2$ and a move from $\{a, d, f, h\}$ to $\{d, g, h\}$ will be blocked by agent g, since her cost increases from $2\frac{\alpha}{2}+3$ to $3\frac{\alpha}{2}+2$. Furthermore, strategies $\{b, d, h\}$ and $\{d, g, h\}$ are the only strategies using three edges, which yield a distance-cost of 3 for agent e. The same reasoning applies for strategies which buy more than three edges. In this case agents b and g would refuse to accept any edge from agent e, since they can only hope for a distance-cost of 2, which is not low enough to compensate for the additional edge-cost. It follows that agent e's move from $\{a, d, f, h\}$ to $\{d, f, h\}$ is her best possible feasible strategy-change. This move transforms network (G_3, α) into network (G_4, α).

In network (G_4, α) we claim that agent c is unhappy and that her best possible feasible strategy-change is the move from strategy $\{b\}$ to $\{b, d\}$, which decrease her cost from $\frac{\alpha}{2}+5$ to $2\frac{\alpha}{2}+3$, which is indeed a strict decrease since $\frac{\alpha}{2} < 2$. Any strategy-change towards a strategy using only one edge, that is, any edge-swap by agent c, will be blocked by the other involved agent. This is true, since no agent has agent c as her unique agent in maximum distance. It follows that c has to move to a strategy which buys at least two edges, if she wants to outperform her current strategy. The best possible strategy with two edges is $\{b, e\}$, since this set is the unique 2-median-set in the graph $G_4 - \{c\}$ which yields a distance-cost of 2 for agent c. Unfortunately for agent c the move from $\{b\}$ to $\{b, e\}$ will be blocked by

agent e, since this move would increase her cost from $3\frac{\alpha}{2}+4$ to $4\frac{\alpha}{2}+3$. No other strategy using two edges yields distance-cost of 2 for agent c, which implies that she has to buy more than 2 edges to outperform her move to $\{b,d\}$. It is easy to see that with at least three edges and less than seven edges at best a distance-cost of 2 is possible for agent c, but this does not suffice to compensate the higher edge-cost. Furthermore, buying seven edges is clearly too expensive. Thus, the indicated move from $\{b\}$ to $\{b,d\}$ is a best possible feasible move for agent c. This move transforms (G_4,α) into (G_1,α) and we have completed the best response cycle. ∎

5. On the Structure of Selfishly Created Networks

A significant part of the research on Network Creation Games focuses on assessing the impact of the agents' selfish behavior on the overall network quality. Clearly, if there is no or little coordination among the agents, then it cannot be expected that the obtained networks minimize the social cost. The reason for this is that each agent aims to improve the network quality for herself while minimizing the cost spent to achieve this service quality. However, despite this egoistic behavior of agents, recent theoretical results and empirical observations suggest that selfishly built networks are very efficient in terms of the overall cost and of the individually perceived service quality. We believe that to understand this quite surprising phenomenon it is important to understand the structure of equilibrium networks built by selfish agents. This knowledge may then be used in mechanism design for guiding agents towards socially good states or constructively for creating and maintaining robust and efficient overlay or peer to peer networks.

In this chapter we contribute to this endeavor by providing new insights into the structure of equilibrium networks which arise as outcomes of Network Creation Games. Our contributed insights, apart from being of interest on their own, may be used for proving better bounds on the Price of Anarchy for Network Creation Games, since they provide a rich structure to work with.

5.1. Preliminaries

5.1.1. Additional Definitions

Let $G = (V, E)$ be any undirected connected graph. A *bridge* in G is an edge whose removal disconnects G. A *bridge-free component* C_{bf} of G is a maximal induced subgraph of G, where $|V(C_{bf})| \geq 3$ and which does not contain bridges. We call a graph *bridge-free* if it does not contain bridges. Note that we rule out trivial bridge-free components which contain only one vertex. Recall that $G - u$ denotes the graph G after vertex $u \in V(G)$ is removed. We generalize this and let $G - U$, for some set $U \subset V$, denote the graph G after all vertices in U are removed. A *cut-vertex* x of a connected graph G is a vertex, where $G - x$ contains at least two non-empty connected components. A *biconnected component* C_{bc} of a graph G is a maximal induced subgraph of G, where $|V(C_{bc})| \geq 3$ and which does not contain a

cut-vertex. We call a graph *biconnected* if it has no cut-vertex. Note that we rule out trivial biconnected components which contain exactly one edge. A set $U \subset V$ is a *separator* of a connected network G, if $G - U$ contains at least two non-empty connected components. Let $G' = (V', E')$ and $G'' = (V'', E'')$ be induced subgraphs of G. We define $G' \cup G'' := (V' \cup V'', E' \cup E'')$ to be the *union graph* of G' and G'' and $G' \cap G'' := (V' \cap V'', E' \cap E'')$ to be the *shared graph* of G' and G''. We will sometimes abuse notation by writing $uv \cup G'$, where $uv \in E$, which is equivalent to $(\{u, v\}, \{uv\}) \cup G'$.

We will sometimes omit this reference to α, mostly when we focus on graph-theoretic properties of the network.

A more refined view on pure Nash Equilibria will be adopted in one section. A network (G, α) is said to be in *transient* pure Sum-Nash Equilibrium if it is in Sum-NE but there is at least one agent in (G, α) who can perform a cost neutral strategy-change which leads to an unstable network. If there is no such agent in (G, α) then it is in *non-transient* Sum-NE.

We will sometimes refer explicitly to the structure of edge-ownership within a network (G, α). For this purpose, let $\overrightarrow{G}$ be the network (G, α) interpreted as directed graph. That is, we get the directed graph $\overrightarrow{G}$ by replacing every edge in (G, α) with a directed arc which is directed away from its owner.

5.1.2. Related Work

Network Creation Games, as defined in Section 2.2, were introduced with the intention of modeling and understanding networks which are created by selfish agents. Fabrikant, Luthra, Maneva, Papadimitriou and Shenker [FLM+03] showed for the Sum-NCG that the Price of Stability is 1, except for $1 \leq \alpha < 2$ where it is upper bounded by $\frac{4}{3}$, and gave the first general bound of $\mathcal{O}(\sqrt{\alpha})$ on the Price of Anarchy by proving a bound of $\mathcal{O}(\sqrt{\alpha})$ on the diameter and by showing that if any Sum-NE network has diameter at most D, then the Price of Anarchy is bounded by $\mathcal{O}(D)$. Moreover, they proved that computing a best response in the Sum-version is NP-hard and conjectured that above some constant edge-price all (non-transient) NE networks are trees. This conjecture, called the Tree Conjecture, is especially interesting since they have shown that tree networks which are in Sum-NE have constant Price of Anarchy. The Tree Conjecture was later disproved by Albers, Eilts, Even-Dar, Mansour and Roditty [AEED+06]. They gave a generic construction of a non-tree network in non-transient Sum-Nash Equlibrium for $1 < \alpha \leq \sqrt{n/2}$ where $n \geq n_0$ for any positive n_0.

A line of works [Lin03],[AEED+06],[DHMZ12],[MS13] has focused on improving the Price of Anarchy bound for the Sum-NCG. The currently best known upper bound for all ranges of α is $2^{\mathcal{O}(\sqrt{\log n})}$ and is due to Demaine, Hajiaghayi, Mahini and Zadimoghaddam [DHMZ12]. Interestingly, for most ranges of α much better upper bounds are known: As shown by Demaine et al [DHMZ12], the Price of Anarchy

is constant if $\alpha < n^{1-\epsilon}$, for any fixed $\epsilon \geq \frac{1}{\log n}$. On the other end of the spectrum Mihalák and Schlegel [MS13] proved a constant bound if $\alpha > 273n$, by establishing that all networks in SUM-NE for such α must be trees. Very recently, Graham, Hamilton, Levavi and Loh [GHLL13] improved the constant upper bound on the Price of Anarchy in the SUM-NCG for the case where $\alpha < \sqrt[3]{n}$ to $1 + o(1)$. Halevi and Mansour [HM07] studied a version where agents have interests and want to be connected to only a subset of the other agents. For this case they showed an upper bound of $\mathcal{O}(\sqrt{n})$ on the Price of Anarchy in the SUM-version. Demaine, Hajiaghayi, Mahini and Zadimoghaddam [DHMZ09] considered a cooperative version of the SUM-NCG and a version where the game is played on a host graph, which specifies which edges may be bought. For the cooperative SUM-version they prove a general polylogarithmic upper bound on the Price of Anarchy whereas for the version played on a host graph they provide an upper bound of $\mathcal{O}(\sqrt{\alpha})$ and $\min\{\mathcal{O}(\sqrt{\alpha}), n^2/\alpha\}$ for $\alpha < n$ and $\alpha \geq n$, respectively, and a lower bound of $\Omega(\min\{\alpha/n, n^2/\alpha\})$ on this ratio. Another cooperative version of the SUM-NCG was studied by Andelman, Feldman and Mansour [AFM09] who introduced the strong Price of Anarchy and showed that this ratio is constant for the SUM-NCG if $\alpha \geq 2$.

The Price of Anarchy for the MAX-NCG was studied in [DHMZ12] and [MS13]. The currently best bounds are due to Mihalák and Schlegel [MS13] who proved a general upper bound of $2^{\mathcal{O}(\sqrt{\log n})}$ on the Price of Anarchy and showed a constant bound if $\alpha \in \mathcal{O}(n^{-\frac{1}{2}})$ or if $\alpha > 129$. The latter is achieved by proving that all networks in MAX-NE for such α must be trees and by showing that Price of Anarchy for tree networks in the MAX-NCG is constant. Along the way, the authors of [MS13] proved that computing a best response in the MAX-NCG is NP-hard as well. Bilò, Gualà, Leucci and Proietti [BGLP12] studied the MAX-NCG on a host graph and proved an upper bound on the Price of Anarchy of $\mathcal{O}(n/(\alpha + r_H))$, where r_H is the radius of the host graph H, if $\alpha < n$, and a constant upper bound if $\alpha \geq n$. They also showed a lower bound of $\Omega\left(\sqrt{n/(1+\alpha)}\right)$ for $\alpha \in o(n)$.

Alon, Demaine, Hajiaghayi and Leighton [ADHL13] showed an upper bound of $2^{\mathcal{O}(\sqrt{\log n})}$ on the diameter of networks in SUM-Swap Equilibrium. Furthermore, the authors gave a construction for a network in MAX-SE having diameter in $\Theta(\sqrt{n})$. It is easy to see that networks which minimize the social cost in both versions must have diameter at most 2. Thus, these bounds on the diameter carry over to the Price of Anarchy of the respective version. This yields an upper bound of $2^{\mathcal{O}(\sqrt{\log n})}$ for the Price of Anarchy in the SUM-Swap Game and a lower bound of $\Omega(\sqrt{n})$ in the MAX-Swap Game. Ehsani, Fazli, Mehrabian, Sadeghabad, Safari, Saghafian and ShokatFadaee [EFM+11] gave the same upper bound of $2^{\mathcal{O}(\sqrt{\log n})}$ for a bounded budget version of the SUM-NCG. Mihalák and Schlegel [MS12] observed that this proof carries over to SUM-Asymmetric Swap Games as well. Moreover, they proved that all networks in SUM-Asymmetric Swap Equilibrium contain at most one bridge-free component. Furthermore, they gave a tight bound of $\Theta(\log n)$ on the diameter of tree networks in SUM-ASE and showed that if the minimum degree of any vertex

in the bridge-free component is at least n^ϵ, for $\frac{4\log 3}{\log n} < \epsilon < 1$, then the diameter of such a network in SUM-ASE is bounded by $\mathcal{O}(\log n)$ as well. They conjectured that the diameter of any network in SUM-ASE is bounded by $\mathcal{O}(\log n)$. Nikoletseas, Panagopoulou, Raptopoulos and Spirakis [NPRS13] proved polylogarithmic upper bounds on the diameter of networks in SUM-Swap Equilibrium if these networks are dense enough or have large k-vicinity. Interestingly, they use the probabilistic method to provide a structural characterization of stable networks for this version. Last but not least, Cord-Landwehr, Hüllmann, Kling and Setzer [CLHKS12] considered Swap Games on tree networks where agents have communication interests. They show that MAX-SE tree networks can have a diameter of $\Theta(\sqrt{n})$ which implies the same bound for the Price of Anarchy. If the game is played on a host graph, then this ratio can be as high as $\Theta(n)$. For the SUM-version with interests on trees they provide a lower bound of $\Omega(\sqrt{n})$ on the diameter of a SUM-SE tree network.

5.1.3. Our Contribution

We show a multitude of structural properties for the whole spectrum of equilibria for the SUM-version of Network Creation Games.

Starting with networks in SUM-SE, we give a strong characterization of their structure. They can have at most one cut-vertex and if the diameter is at least 3, then there can be at most one biconnected component. This implies that all leaf-vertices of the network must be connected to the unique cut-vertex. Moreover, we give a substantially easier proof of a theorem by Alon et al. [ADHL13] and, along the way, we show how to fix a flaw in their construction.

Then we prove bounds on the diameter of SUM-ASE and SUM-GE networks in the case where the network has exactly n edges. For SUM-ASE networks we show an upper bound of $\mathcal{O}(\log n)$ and for SUM-GE networks we give a constant upper bound. The latter implies constant Price of Anarchy for SUM-GE networks having n edges.

For SUM-GE networks we extend and strengthen a result of Mihalák and Schlegel [MS12] by showing that all SUM-GE networks must have a very specific shape: They consist of at most one biconnected component and possibly some attached trees. Since SUM-NE are a subset of SUM-GE this property directly carries over. Furthermore, we show that bounding the diameter of the biconnected component suffices to bound the Price of Anarchy for both equilibria.

Since the famous Tree Conjecture, little progress was made on finding the right range of α where the structure of equilibrium networks tips from arbitrary networks to tree networks. We improve the currently known lower bounds on the edge-cost parameter α for that case.

Our main contribution is an extensive study of the structure of SUM-Nash Equilibrium networks for the case where $\alpha > 2n - 6$. Most known structural results are targeted at much lower ranges of α and, to the best of our knowledge, very little is known for higher edge-cost. We introduce two main tools, Min-Cycles and

Critical Pairs, and show their versatility in proving several strong structural properties. Min-Cycles are special cycles where a shortest path between any pair of cycle-vertices lies on the cycle itself. We believe that Min-Cycles are interesting for analyzing distance-related graph properties. Such cycles can be found easily and we show that in equilibrium networks these Min-Cycles can overlap only in a very specific way. The reason for this is our second tool, called Critical Pairs. Such a pair consists of two agents which lie in a very specific distance relationship. We use this distance structure in showing that no network in Sum-Nash Equilibrium can contain a Critical Pair if $\alpha > 2n - 6$. Having this tool in hand, we can show that the edge-ownership in Min-Cycles of Sum-NE networks must have a certain structure and that Min-Cycles may overlap only in exactly one relatively short path. On the one hand, if this overlapping path forms a separator of the network, then the network cannot be in Sum-ASE. On the other hand, if the overlapping path is not a separator, then this implies the existence of a cycle of Min-Cycles in the network. We focus on such cycles of Min-Cycles and show that the length and shape of such cycles is heavily restricted. It turns out that any Sum-NE network for some $\alpha > 2n - 6$ having at least $n + 1$ edges must contain an even wheel of Min-Cycles. Furthermore the ownership in all Min-Cycles must be directed.

All in all, we provide new tools for analyzing the structure of equilibrium networks. These tools lead to various side constraints which Sum-NE networks have to fulfill. We are confident that with these or similar structure enriching tools it is possible to finally settle the long standing open problem of proving the Price of Anarchy of Sum-NE networks to be constant. Moreover, we believe that these ideas may be carried over to the study of the structural properties of equilibrium networks in the Max-version.

5.2. On the Structure of Sum Swap Equilibria

We start with stating a result from Alon, Demaine, Hajiaghayi and Leighton.

Lemma 5.2.1 ([ADHL13]) *If a vertex u has eccentricity at most 2 in a network G, then agent u cannot decrease her cost by swapping an edge in G. This holds for the* Sum- *and the* Max-*version.*

Note that the diameter of a network G is a trivial upper bound to the eccentricity of any vertex in G. Hence, Lemma 5.2.1 implies the following:

Corollary 5.2.2 *Every connected network with diameter at most 2 is in* Sum-*SE and in* Max-*SE.*

The story does not end here. The next theorem shows that there are swap-stable networks having a larger diameter than 2. This observation is due to Alon et al. [ADHL13] but we give a substantially easier proof of this fact.

Theorem 5.2.3 ([ADHL13]) *There is a network in* Sum-*SE having diameter* 3.

Remark 5.2.4 *The original proof in [ADHL13] is flawed. There the authors present a network having diameter* 3 *and show that in this network no agent can perform an improving move in the* Sum-*Swap Game. This statement is not true: Agents* d_1, d_2 *and* d_3 *can perform improving edge-swaps. For example, agent* d_1 *can perform the edge-swap* $d_1c_{1,1}$ *to* $d_1c_{2,1}$ *and thereby decrease her cost by* 1*. However, the given construction can be fixed by introducing a new vertex, which is connected to* d_1, d_2, d_3 *and* a*. We give a much smaller example of a* Sum-*SE network having diameter* 3 *which leads to a substantially easier proof of its swap-stability.*

Proof (of Theorem 5.2.3) We analyse the network G depicted in Fig. 5.1.

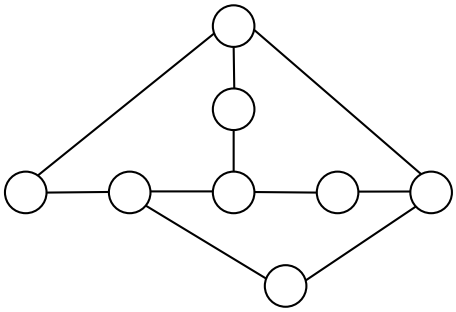


Figure 5.1.: A Sum-SE network having diameter 3.

Note that all vertices having degree 3 have eccentricity 2. Thus, by Lemma 5.2.1, the respective agents cannot perform an improving edge-swap. Every vertex having degree 2 has exactly one vertex in distance 3 and, thus, these agents each also have two vertices in distance 1, four vertices in distance 2. Thus, they all have cost $c = 13$. Each degree-2 agent u in G has a unique shortest path to all her four vertices which are at distance 2 from u. Furthermore, each incident edge of u is part of exactly two such shortest paths. Hence, any possible edge-swap of a degree-2 vertex u in G leads to a graph G', where at least one vertex has distance 3 to vertex u. Thus, since the number of neighbors of u does not change, agent u must have cost of at least c in G', which implies that x cannot perform an improving move. ■

Mihalák and Schlegel [MS12] proved the following structural property. Since every network in Sum-SE is also in Sum-ASE, this property directly carries over to Sum-SE networks.

Theorem 5.2.5 ([MS12]) *Every network in* Sum-*ASE has at most one bridge-free component.*

They gave an example network in Sum-ASE having one brige-free component which contained a cut-vertex and two biconnected components. In the following we will

give a stronger characterization for networks in SUM-SE. We will show in the following that networks in SUM-SE can have at most one cut-vertex and that all SUM-SE networks having at least diameter 3 can have at most one biconnected component. Moreover, if such a network has leaves, then all leaves must be connected to the unique cut-vertex.

Lemma 5.2.6 *Let x be a cut-vertex of a* SUM*-SE network G and let $G_1, \dots, G_q$ be the connected components of $G-x$. We have that if $|G_i| \leq \lfloor \frac{n}{2} \rfloor$, for some $1 \leq i \leq q$, then for all $u \in V(G_i)$ it holds that $d_G(u,x) = 1$.*

PROOF Assume that for some G_i with $|G_i| \leq \frac{n}{2}$ there is an agent $u \in V(G_i)$ with $d_G(u,x) = 2$. We claim that agent u has an improving response and thus G cannot be in SUM-SE. Let the shorest path from u to x use the vertex w. Consider the edge-swap uw to ux. Since $|V(G) \setminus V(G_i)| \geq \frac{n}{2}$ this swap decreases agent u's shortest path to $|V(G) \setminus V(G_i)|$ many vertices by one and it increases agent u's shortest path to $|G_i| - 1$ many vertices by at most one. Since $|G_i| - 1 < \frac{n}{2}$, this yields a strict decrease in cost for agent u, which contradicts the swap-stability of G. ∎

Lemma 5.2.7 *Let G be a network in* SUM*-SE having diameter at least 3 and let x be a cut-vertex of G. Let $G_1, \dots, G_q$ be the connected components of $G - x$. We have that if $|G_i| \leq \lfloor \frac{n}{2} \rfloor$, for some $1 \leq i \leq q$, then $|G_i| = 1$.*

PROOF Consider a component G_i with $2 \leq |G_i| \leq \lfloor \frac{n}{2} \rfloor$. By Lemma 5.2.6, we have that all vertices in $V(G_i)$ are neighbors of x in G. Thus, for G_i to be connected there must be at least one additional edge in G_i, which is not incident to x. Let uv, for some $u, v \in V(G_i)$, be such an edge. Since G has diameter at least 3, there must be a component G_j with $|G_j| > \lfloor \frac{n}{2} \rfloor$. Otherwise, again by Lemma 5.2.6, we have that $d_G(x,w) = 1$, for all vertices $w \neq x$ of G, which contradicts that G has diameter 3. Furthermore, there must be a vertex z in G_j with $d_G(x,z) = 2$. Observe that, by x being a cut-vertex, we have $d_G(u,z) = 3$.

Now consider agent u and the edge-swap uv to uz. This swap yields a cost decrease of at least 2 for agent u. Since all vertices in G_i, especially vertex v, are connected to x, we have that the swap increases agent u's distance only to vertex v by 1. Thus, the swap yields a strict decrease in cost for agent u, which contradicts the swap-stability of G. Hence, there are no edges between vertices of G_i in G, which are not incident to x. It follows that each G_i contains only one vertex. ∎

Theorem 5.2.8 *If a* SUM*-SE network G has diameter at least 3 and no leaves, then G is biconnected.*

PROOF Assume that G is in SUM-SE, has diameter at least 3 and no leaves. Furthermore, assume towards a contradiction that G contains at least one cut-vertex x. Let $G_1, \dots, G_q$ be the connected components of $G-x$. Observe that there can be at most one component G_i with $|G_i| > \lfloor \frac{n}{2} \rfloor$. Hence, since x is a cut-vertex, there must

be at least one component G_j, with $i \neq j$, such that $|G_j| \leq \lfloor \frac{n}{2} \rfloor$. By Lemma 5.2.7, we have that $|G_j| = 1$, which implies that G must contain a leaf. Hence, we have a contradiction. ∎

Lemma 5.2.9 *If a* Sum*-SE network G has a bridge-free component C and diameter at least 3, then C must be a biconnected component.*

Proof Consider a bridge-free component C of a Sum-SE network G having diameter at least 3 and assume that C has a cut-vertex x. Let $G_1, \ldots, G_q$ denote the connected components of $G-x$. If we have $|G_i| \leq \frac{n}{2}$, for all i, then, by Lemma 5.2.6, all vertices of $G - x$ must be neighbors of x, which implies that G has diameter at most 2, which is a contradiction. Hence, assume that there is a connected component G_j containing more than $\frac{n}{2}$ vertices. Clearly, there can be only one such component. By Lemma 5.2.7 and the fact that a bridge-free component contains at least three vertices and is connected, we have that the connected component G_j must contain $C - x$. Thus, $C - x$ must be connected, which contradicts the fact that x is a cut-vertex of C. ∎

Theorem 5.2.10 *Every* Sum*-SE network has at most one cut-vertex.*

Proof Let G be in Sum-SE. By Theorem 5.2.5, G has at most one bridge-free component. If G has no bridge-free component, then it is a tree. In this case G must be a star and therefore G has exactly one cut-vertex.

Now, consider that G has a bridge-free component C. There are three cases:

1. If G has diameter 1, then it must be the complete graph, which implies that there is no cut-vertex.

2. Let G have diameter 2. Assume that G has at least two cut-vertices and let x and y be two of them. Since G is connected, there must be a shortest path P connecting x and y. Furthermore, since x is a cut-vertex, there must be a vertex u, which is a neighbor of x, and every shortest path from u to y contains x. Analogously, since y is a cut-vertex there must be a vertex v, which is a neighbor of y, and every shortest path from v to x contains y. Since $d(x, y) \geq 1$, $d(u, x) = 1$ and $d(v, y) = 1$, it follows that $d(u, v) \geq 3$ and we have a contradiction.

3. Let G have diameter at least 3. By Lemma 5.2.9, we have that the bridge-free component C must be a biconnected component. Assume that G has $k \geq 2$ cut-vertices $x_1, \ldots, x_k$. By Lemma 5.2.6 and by Lemma 5.2.7, we have that each x_i has some neighboring leaves. Here a leaf is a vertex which has degree 1. Let L be the set of all leaves of G and consider a leaf $l \in L$, which is connected to some cut-vertex x_i. Clearly, the best response of agent l is to connect to the vertex of $G-l$, which has the smallest cost within the network $G-l$. Thus, x_i must be a 1-median vertex of $G-l$. Now consider a leaf $l' \neq l$, which is connected to $x_j \neq x_i$.

Hence, x_j must be a 1-median vertex in $G - l'$. Let $c(u)$ and $c'(u)$ denote the cost of agent u in $G - l$ and $G - l'$, respectively. We have that $c(x_i) \leq c(x_p)$ and $c'(x_j) \leq c'(x_q)$, for $i \neq p$ and $j \neq q$. Observe that $d_G(x_i, l') = d_G(x_j, l) = D \geq 2$ and $d_G(x_i, l) = d_G(x_j, l') = 1$. Thus we have, $c'(x_i) = c(x_i) + 1 - D < c(x_i)$ and $c'(x_j) = c(x_j) + D - 1 > c(x_j)$. This yields $c'(x_i) < c(x_i) \leq c(x_j) < c'(x_j)$, which contradicts the swap-stability of G, because agent l' has an improving response. Thus, we have that all leaves must be connected to the same cut-vertex and it follows that this cut-vertex is unique. ∎

Observe that Theorem 5.2.8 and Theorem 5.2.10 characterize the shape of any Sum-SE network G having diameter at least 3. Such a network is biconnected if it has no leaves, or all leaves of G are connected to a unique cut-vertex of G. Note that there are Sum-SE networks having diameter 2 and no leaves which contain a cut-vertex. The left network in Fig.5.2 without edge-ownership is an example

5.3. On the Structure of Sum Asymmetric Swap Equilibria

It was shown by Mihalák and Schlegel [MS12] that every tree network in Sum-ASE has diameter in $\mathcal{O}(\log n)$. This bound is tight, since a complete binary tree is in Sum-ASE if all edges are owned by the vertex which is closer to the root. We go one step further and bound the diameter of n-agent networks in Sum-ASE having exactly n edges.

Theorem 5.3.1 *If a network in* Sum-*Asymmetric Swap Equilibrium has exactly one cycle, then this cycle has length at most* 5.

Proof We assume towards a contradiction that there is a network G in Sum-Asymmetric Swap Equilibrium having exactly one cycle which contains at least 6 nodes. We show that there is a cycle-agent who can swap an edge and thereby strictly decrease her cost. Let $a_1, \ldots, a_k$, with $k \geq 6$, denote the cycle-agents in cyclic order. Let G_i be the network G where the edges $a_{i-1}a_i$ and a_ia_{i+1} are removed. Let U_i be the set of agents in the connected component of G_i which contains a_i. We will study the network G', which is the induced subgraph of G containing only the cycle-agents $a_1, \ldots, a_k$ and where every cycle-agent a_i has weight $w(a_i) = |U_i|$. Note that $w(a_i) \geq 1$ for all i, since $a_i \in U_i$. The network G' defines the cyclic sequence of weights

$$S(G') = w(a_1), w(a_2), \ldots, w(a_k), w(a_1) \ .$$

We will argue about *connected subsequences* $Z(G')$ of $S(G')$, where $Z(G')$ is a connected subsequence of $S(G')$ if $Z(G')$ is a subsequence of $S(G')$ with the property that neighboring elements in $Z(G')$ must be neighboring elements in $S(G')$.

There are two cases:

1. If all $w(a_i)$ are equal, then we choose a cycle-agent a_j who owns at least one edge and w.l.o.g. let this edge be $a_j a_{j+1}$. We claim that agent a_j can perform the edge-swap $a_j a_{j+1}$ to $a_j a_{j+2}$ in network G and thereby strictly decrease her cost.

   This is easy to see, since this swap increases a_j's distance to $w(a_{j+1})$ many agents by 1 but it decreases her distances to $w(a_{j+2}) + w(a_{j+3})$ agents by 1. The latter is true, because the cycle has length at least 6 which implies that a_j's distance to a_{j+3} before the swap was 3.

2. The more interesting case is if not all weights are equal. We show for this case that if G is in Sum-Asymmetric Swap Equilibrium then the sequence $S(G')$ must have a certain shape.

   First of all, assume that $S(G')$ contains a connected subsequence

   $$Q(G') = w(a_j), w(a_{j+1}), w(a_{j+2}) \ ,$$

   where $w(a_j) > w(a_{j+1}) > w(a_{j+2})$. We have that the edge $a_{j+1}a_{j+2}$ must be owned by agent a_{j+1}, since otherwise agent a_{j+2} could swap towards a_j and thereby strictly decrease her cost. Now, if $w(a_{j+3}) \geq w(a_{j+2})$, then this implies that agent a_{j+1} could perform the edge-swap $a_{j+1}a_{j+2}$ to $a_{j+1}a_{j+3}$ and thereby strictly decrease her cost. This is true, since the cycle has length at least 6 which implies that this swap increases a_{j+1}'s cost by $w(a_{j+2})$ but it decreases her cost by $w(a_{j+3}) + w(a_{j+4})$. It follows that $w(a_{j+2}) > w(a_{j+3})$ must hold for G being in Sum-Asymmetric Swap Equilibrium. By iterating this argument we have that the weights in $S(G')$, starting from $w(a_j)$, must be strictly monotonically decreasing, but since $S(G')$ is cyclic this is impossible. Thus, no such connected subsequence $Q(G')$ can exist if G is in Sum-Asymmetric Swap Equilibrium. By an analogous argument it follows that $S(G')$ cannot contain a connected subsequence

   $$P(G') = w(a_j), w(a_{j+1}), w(a_{j+2}) \ ,$$

   where $w(a_j) < w(a_{j+1}) < w(a_{j+2})$. It follows that the sequence $S(G')$ must be alternating. Consider a connected subsequence

   $$R(G') = w(a_j), w(a_{j+1}), w(a_{j+2}) \ ,$$

   where $w(a_j) > w(a_{j+1})$ and $w(a_{j+1}) < w(a_{j+2})$. The edge $a_j a_{j+1}$ must be owned by agent a_{j+1}, since otherwise agent a_j could swap towards a_{j+2} and strictly decrease her cost. Analogously, edge $a_{j+1}a_{j+2}$ must be owned by agent a_{j+1}. It follows that all cycle-edges in G' must be owned by the endpoint having lower weight, if the respective weights are different. Since we assume that not all weights are equal, there must be a connected subsequence

   $$X(G') = w(a_j), w(a_{j+1}), w(a_{j+2})$$

of $S(G')$, where $w(a_j) > w(a_{j+1})$, $w(a_{j+1}) < w(a_{j+2})$ and $w(a_j) \leq w(a_{j+2})$. Again, this is true because $S(G')$ is cyclic. We claim that agent a_{j-1} can strictly decrease her cost by swapping an edge. Since $S(G')$ is alternating, we have that $w(a_{j-1}) < w(a_j)$ and by the above observation we know that a_{j-1} must be the owner of the edge $a_{j-1}a_j$. Now, if a_{j-1} swaps this edge towards a_{j+1}, then her cost increases by $w(a_j)$, but it decreases by $w(a_{j+1}) + w(a_{j+2}) > w(a_j)$, because the cycle has length at least 6 which implies that a_{j-1}'s distance to a_{j+2} before the swap is 3. This is a contradiction to G being in Sum-Asymmetric Swap Equilibrium. ■

Corollary 5.3.2 *Let G be a n-node network having exactly n edges. If G is in* Sum-*ASE, then the diameter of G is in $\mathcal{O}(\log n)$. If (G, α) is in* Sum-*GE for some $\alpha > 0$, then the diameter of G is in $\mathcal{O}(1)$.*

Proof The first statement holds, since G must consist of a cycle with attached trees. Since trees in Sum-ASE can have diameter of at most $\mathcal{O}(\log n)$ [MS12] and by Lemma 5.3.1, we have that the diameter of the cycle is at most 2. The second statement follows from Lemma 5.3.1 and Corollary 5.4.4 and from the fact that any Sum-GE network must be in Sum-ASE as well[1]. ■

5.4. On the Structure of Sum-Greedy Equilibria

In this section we prove a general structural result for all networks in Sum-GE. Recall Theorem 5.2.5 from above which states that networks in Sum-ASE have at most one bridge-free component. The theorem is tight in the sense that there are networks in Sum-ASE which have one bridge-free component but where this component contains a cut-vertex. We will show that this cannot happen in Sum-GE networks.

Lemma 5.4.1 *Let (G, α) be a network in* Sum-*GE having a bridge-free component C_{bf} which contains a cut-vertex $x \in V(C_{bf})$. If $\alpha > 1$, then G has diameter at least 3.*

Proof We assume towards a contradiction that (G, α) is in Sum-GE and has a bridge-free component C_{bf} which contains a cut-vertex x. Furthermore, assume that G has diameter 2 (diameter 1 is obviously impossible).

Since C_{bf} is a bridge-free component of G, we have that C_{bf} must contain at least 3 vertices. Moreover, since C_{bf} contains a cut-vertex x, we have that $G - x$ consists of the connected components $A_1, \ldots, A_k$, for some $k \geq 2$. Since G has diameter 2 and since x is a cut-vertex, we have that $d_G(a, x) = 1$, for all $a \in V(G) \setminus \{x\}$. Consider the subgraph G_i of G, which is induced by the set $V(A_i) \cup \{x\}$. Since C_{bf}

[1] See the discussion in Section 2.2.2.

is a bridge-free component, we have that each induced subgraph G_i must contain a cycle of length at least 3 which contains the cut-vertex x. In any such cycle, there must be an edge, which is not incident to vertex x. Let uv be this edge and let u be the owner of uv. Observe that $u, v \in V(A_i)$ for some i. Deleting edge uv would increase agent u's distance to vertex v by 1, but, since all agents in $V(A_i)$ have distance 1 to x, no other distance of agent u changes. Since $\alpha > 1$, we have that agent u can strictly decrease her cost by deleting the edge uv and it follows that (G, α) cannot be in Sum-GE. This contradicts our assumption. ∎

Theorem 5.4.2 *Let $\alpha \neq 1$ and let (G, α) be a network in Sum-GE. If G contains a bridge-free component, then this component is biconnected.*

Proof For $\alpha < 1$, the statement follows trivially, since (G, α) must be a complete network. Thus, let us assume that $\alpha > 1$. Let (G, α) be a network in Sum-GE and let C_{bf} be a bridge-free component of G. By Theorem 5.2.5 and since Sum-GE networks are in Sum-ASE, this component is unique. We assume towards a contradiction that C_{bf} is not biconnected, that is, that C_{bf} contains a cut-vertex x. Let $A_1, \ldots, A_k$, for some $k \geq 2$, denote the connected components of $G - x$ and let G_i be the subgraph of G which is induced by the set $V(A_i) \cup \{x\}$. Each G_i must contain a cycle of length at least 3, which contains the cut-vertex x, since otherwise there must be a bridge in C_{bf}. By Lemma 5.4.1, we know that G must have diameter at least 3. It follows that at least one subgraph G_j contains a vertex w which has distance at least 2 towards x. We fix G_j and w such that w has maximum distance to x among all vertices in $V(G)$. Now, let us consider a different induced subgraph $G_l \neq G_j$.

Such a subgraph exists, since $k \geq 2$. Moreover, we know that G_l contains a cycle C_l which contains vertex x. Let $u \in V(C_l)$ be a cycle-vertex who owns a cycle-edge and which has maximum distance to x. Let v denote the cycle-neighbor of u to which agent u owns the cycle-edge uv. If agent u owns two cycle-edges, then we fix v to be the cycle-neighbor of u which has maximum distance to x. Since (G, α) is in Sum-GE, we know that agent u cannot decrease her cost by deleting edge uv. Let N_v^u be the set of vertices of G, which have vertex v on *all* their shortest paths towards vertex u. By the choice of u and v, it follows that $x \notin N_v^u$ and that $N_v^u \subset V(A_l)$. Observe that if agent u deletes her edge towards v, then she is still connected to all vertices in N_v^u, since a shortest path from u to x and all shortest paths from x to z, for all $z \in N_v^u$, remain intact. Since this edge-deletion does not yield a strict cost decrease, it follows that

$$\alpha + \sum_{z \in N_v^u} d_G(u, z) \leq \sum_{z \in N_v^u} (d_G(u, x) + d_G(x, z)) \leq \sum_{z \in N_v^u} (d_G(w, x) + d_G(x, z)),$$

where the second inequality holds, since $d_G(u, x) \leq d_G(w, x)$, by choice of w. Note that the right sum denotes agent w's distance-cost towards all vertices in N_v^u. This is true, since $N_v^u \subset V(A_l)$, $w \notin V(G_l)$ and because x is a cut-vertex of G, which implies that w has x on all its shortest paths towards vertices in N_v^u.

If agent w buys the edge wv, then the above inequalities guarantee that this purchase does not increase her cost. This is easy to see, since with the edge wv agent w has the same distance-cost as agent u towards all the vertices in N_v^u and additionally she has to pay α for the edge wv itself. It remains to show that buying the edge wv strictly decreases agent w's cost. By assumption, we have $d_G(w,x) \geq 2$, which implies $d_G(w,u) \geq 3$, since $u \neq x$. Thus, buying the edge wv decreases agent w's distance towards u by at least 1. Since $u \notin N_v^u$, it follows that agent w can strictly decrease her cost by buying one edge, which is a contradiction to (G,α) being in Sum-GE. ∎

Remark 5.4.3 *Fig. 5.2 (left) shows a* Sum*-GE network where the bridge-free component contains a cut-vertex for* $\alpha = 1$.

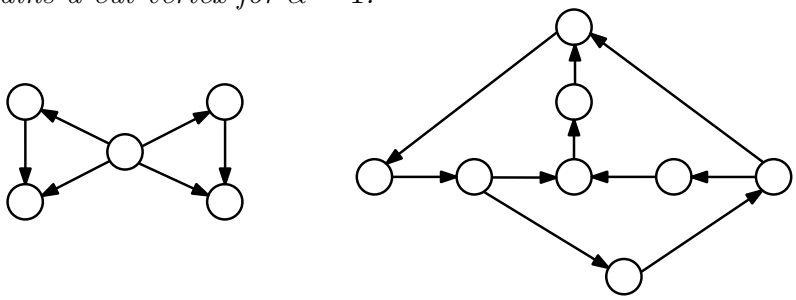


Figure 5.2.: Left: A Sum-GE network for $\alpha = 1$ where the bridge-free component contains a cut-vertex. Right: A biconnected diameter 3 Sum-GE network for $2 < \alpha < 3$.

Corollary 5.4.4 *Let* (G,α) *be a non-tree network in* Sum*-GE. If the unique biconnected component of* (G,α) *has diameter* D*, then* G *has diameter* $\mathcal{O}(D)$.

Proof Clearly, if (G,α) is a biconnected network, then the statement follows trivially. Thus, let us assume that (G,α) contains at least one cut-vertex. In this case, the statement follows by the argument given in the proof of Theorem 5.4.2. Let C_{bc} be the unique biconnected component of G and let $w \notin V(C_{bc})$ be a vertex of G having maximum distance to its nearest vertex x of C_{bc}. Clearly, vertex x must be a cut-vertex of G. We assume that $d_G(w,x) \geq 2$, since otherwise we are trivially done. We show that $d_G(w,x) < D$, which implies the statement.

Let C_l be the longest simple cycle in C_{bc} which contains x and we choose vertices u and v as in the proof of Theorem 5.4.2. Clearly, since C_{bc} has diameter D, we have that $d_G(u,x) \leq D$. Thus, if $d_G(w,x) \geq D$, then we have that agent w can buy the edge wv and thereby strictly decrease her cost, which contradicts that (G,α) is in Sum-Greedy Equilibrium. ∎

Now we show that bounding the diameter of the unique biconnected component of all networks in Sum-Greedy Equilibrium suffices to bound the Price of Anarchy.

Lemma 5.4.5 (Lemma 19.4 in [NRTV07]) *If a network* (G,α) *in* Sum*-NE has diameter* D*, then its social cost is at most* $\mathcal{O}(D)$ *times the minimum possible cost.*

By inspecting the proof of Lemma 19.4 in [NRTV07], it is easy to see that only greedy strategy-changes are used. Hence, the above statement directly carries over to SUM-GE. Together with Corollary 5.4.4, we obtain the following statement.

Corollary 5.4.6 *If the biconnected component of a network (G, α) in* SUM-*GE has diameter D, then its social cost is at most $\mathcal{O}(D)$ times the minimum possible cost.*

This leads to the following statement.

Corollary 5.4.7 *The Price of Anarchy in the* SUM-*NCG and in the* SUM-*GBG is constant for all connected n-vertex networks having at most n edges.*

PROOF For connected n-vertex networks having $n - 1$ edges, that is, for tree networks, this statement for the SUM-NCG was already shown by Fabrikant et al. [FLM+03]. For tree networks in the SUM-GBG the statement follows easily, since, by Theorem 3.3.2, we have that the set of tree networks in SUM-GE and the set of tree networks in SUM-NE coincides.

The statement for exactly n edges in the SUM-NCG follows from Corollary 5.3.2 and the fact that every SUM-NE is in SUM-GE. For the SUM-GBG we use Theorem 3.3.9, which guarantees that every network in SUM-GE is in 3-approximate SUM-NE. Thus, agents in any SUM-GE network on n-vertices have in the worst-case a cost which is three times their cost in the SUM-NE network on n-vertices having the highest social cost. Thus, the social cost of any n-vertex network in SUM-GE is at most three times the cost of the n-vertex network in SUM-NE having the highest social cost. Since this cost is at most a constant times the minimum possible social cost for n-vertex networks, it follows that the Price of Anarchy in the SUM-GBG on n-vertex networks having n-edges is constant. ∎

Extensive computer simulations and attempts to construct SUM-GE networks having large diameter lead us to to following conjecture. We remark that there are biconnected SUM-GE networks having diameter 3, see Fig. 5.2 (right). Even constructing a diameter 4 SUM-GE network seems challenging.

Conjecture 5.4.8 *The diameter of the unique biconnected component of a* SUM-*GE network is constant.*

Note that settling the above conjecture would settle the long-standing problem of proving that the Price of Anarchy in the SUM-NCG is constant.

5.5. The Boundary between Tree and Non-Tree Equilibria

We investigate for which values of α non-tree equilibria are possible in the SUM-version. It was shown by Albers et al. [AEED+06] that for any positive n_0 there is

a non-transient non-tree network in SUM-NE containing $n \geq n_0$ agents for all $1 < \alpha \leq \sqrt{n/2}$. This result disproved the Tree Conjecture by Fabrikant et al. [FLM+03]. Their construction uses finite affine planes and is rather involved. Interestingly in the same paper the authors give another construction which shows that transient non-tree networks in SUM-NE for $\alpha \leq n/2$ exist. This result can be improved as follows.

Theorem 5.5.1 *For any $n_0 \geq 5$ there is a transient non-tree* SUM*-Nash Equilibrium network on $n \geq n_0$ agents for $\frac{3}{5}n < \alpha \leq \frac{4}{5}n$.*

PROOF We give a family of networks $(G_0, \alpha_0), (G_1, \alpha_1), \ldots,$ with $\alpha_i = \frac{4}{5}|V(G_i)|$, which are in transient SUM-Nash Equilibrium.

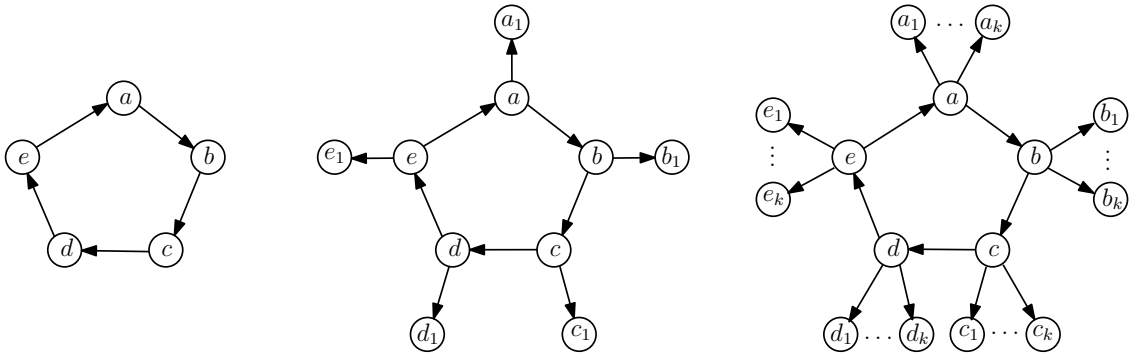


Figure 5.3.: The networks (G_0, α_0), (G_1, α_1) and (G_k, α_k).

The network (G_k, α) is constructed as follows: Take a directed cycle of length 5 and connect every cycle-vertex to exactly k new vertices with edges pointing towards the non-cycle vertices. Let a, b, c, d, e be the cycle-vertices and let u_i be the non-cycle vertices of vertex u for $u \in \{a, b, c, d, e\}$ and $1 \leq i \leq k$. See Fig. 5.3 for an illustration of the construction. The network (G_k, α_k) is derived from this directed network by interpreting edge-directions as edge-ownership information[2] and ignoring directions otherwise.

For (G_k, α_k) to be a transient Nash Equilibrium network, we have to make sure (1) that no leaf-agent can improve by buying at least one edge (since they do not own any edge) and (2) that every cycle-agent has chosen an optimal strategy.

We first show (1). By symmetry of the construction, we can focus on one leaf-vertex only. Moreover, by Lemma 3.3.1, we have that if an agent can improve by buying j edges, then this agent can improve by buying one edge as well. Thus, it suffices to show that leaf-agents cannot improve by buying one edge. Let us consider agent a_1. A best possible edge for a_1 is the edge $a_1 d$. The edge $a_1 d$ decreases agent a_1's distance to agent d by 2 and to agent c by 1. Furthermore, this edge decreases a_1's distances to d_i by 2 and to c_i' by 1, for $1 \leq i \leq k$. Hence, the distance decrease

[2]Edges point away from their owner.

of edge a_1d for agent a_1 is $3k+3$. Thus, to satisfy (1), we have to enforce that $\alpha_k > 3k+3$. If $k=0$, then (1) is trivially satisfied.

Now we consider a cycle-agent of (G_k, α_k) and again, by symmetry, it suffices to argue for one specific agent. Let this be agent a. Observe that agent a cannot remove any edge towards a leaf, since this would disconnect the network. Let (G'_k, α_k) be the network (G_k, α_k) without agent a's edge ab. In (G'_k, α_k) agent a's distance to each of the agents $b, b_1, \ldots, b_k$ increases by 3 and to each of the agents $c, c_1, \ldots, c_k$ increases by 1. Observe that no other distance increases. Hence, we have that agent a's distance-cost in (G'_k, α_k) is increased by $4k+4$. It follows that we have to enforce that $\alpha_k \leq 4k+4$, since otherwise agent a would be better off not to buy edge ab. Furthermore, observe that the edge ab is a best possible additional edge for a in the network (G'_k, α_k). This is true since it is always better for agent a to buy an edge towards a cycle-agent than towards a leaf-agent and because the edge ac yields the same distance decrease of $4k+4$. Thus, agent a's strategy in (G_k, α_k) is a best possible strategy which buys exactly $k+1$ edges.

Now we show that agent a cannot improve by buying one additional edge, which, by the same argument as above, implies that a cannot improve by buying $j \geq 1$ edges. It is easy to see that a best possible additional edge for a in (G_k, α_k) is edge ac, which yields a distance decrease of $k+1$. Hence, if $\alpha_k > k+1$, then no set of additional edges yields a cost decrease for agent a.

It remains to show that a cannot strictly decrease her cost by removing edge ab and buying at least two edges instead. First, let us assume that a can remove ab, buy two edges ax and ay and thereby strictly decrease her cost compared to a's cost in (G_k, α_k). Observe that $x \neq b$ and $y \neq b$ must hold, since otherwise the edge not connecting to b would be an additional edge in (G_k, α_k). Furthermore, it is easy to see that $x \neq y$ must hold and that x or y cannot be leaves, since leaves are always dominated by their corresponding cycle-neighbors. Hence, the only possible strategy for agent a, which satisfies the mentioned constraints, is to buy the edges ac and ad. This strategy yields a distance decrease of $5k+5$ compared to buying no edge. Clearly, every edge in an SUM-equilibrium strategy must yield at least a distance decrease of α_k, since otherwise the agent would be better off without buying that edge. Since $5k+5 < 2(3k+3) < 2\alpha_k$, we have that agent a's new cost is strictly higher than a's cost in (G_k, α_k). Observe that removing edge ab and buying three edges ax, ay, az, with $x \neq b, y \neq b$ and $z \neq b$, yields a distance decrease of $6k+6 < 3(3k+3) < 3\alpha_k$. Hence, agent a cannot strictly decrease her cost by buying three edges. For more than three edges, where no edge is allowed to connect to b, an analogous argument yields that a cannot strictly decrease her cost. Hence, agent a cannot change her strategy to strictly decrease her cost.

Thus, we have that (G_k, α_k) is in transient SUM-Nash Equilibrium for $3k+3 < \alpha_k \leq 4k+4$. Since the number of agents in (G_k, α_k) is $5k+5$ we have that $\frac{3}{5}n < \alpha_k \leq \frac{4}{5}n$. Observe that for all k, the edge ab yields the same distance decrease as edge ac for agent a. Hence, (G_k, α_k) is a transient SUM-Nash Equilibrium, even if we choose $3k+3 < \alpha_k < 4k+4$. ∎

Remark 5.5.2 *Independently from us the above construction was given as an example by Eilts [Eil08]. To the best of our knowledge, for $k \geq 1$ the network $(G_k, 3k+3 < \alpha \leq 4k+4)$ is the only known non-tree* Sum*-Nash Equilibrium network having a diameter larger than* 3*.*

If we consider Sum-Greedy Equilibria instead of Sum-Nash Equilibria, then we can find non-tree equilibria for even higher edge-cost α. In fact, we can choose α arbitrarily close to n.

Theorem 5.5.3 *For any $\epsilon > 0$ and any $n_0 \geq 9$ there is a transient non-tree* Sum*-Greedy Equilibrium network on $n \geq n_0$ agents for $n - \epsilon < \alpha < n$.*

Proof The desired network (G'_k, α'_k) is constructed as follows: We modify the construction of network (G_k, α_k) provided in the proof of Theorem 5.5.1, starting with $k = 1$. We remove agents $a_1, \ldots, a_k$ and agent a now owns both incident cycle-edges towards b and e. See Fig. 5.4 for an illustration.

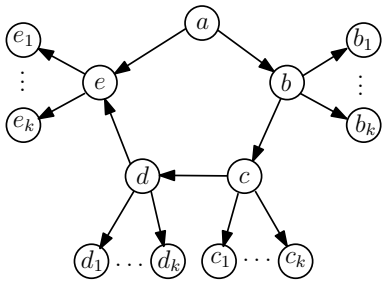


Figure 5.4.: A transient non-tree Sum-Greedy Equilibrium network on n agents for $n - \epsilon < \alpha < n$ for any $\epsilon > 0$.

We claim that for any $\epsilon > 0$ and any $n_0 \geq 9$ there is a k such that (G'_k, α'_k) has at least $n \geq n_0$ agents and is in transient Sum-Greedy Equilibrium for $n - \epsilon < \alpha'_k < n$. This can be seen as follows: Analogous to the proof of Theorem 5.5.1, we have that in (G'_k, α'_k) no leaf-agent can improve by buying an edge if $\alpha'_k > 3k+3$ and that no cycle-agent wants to delete an edge if $\alpha'_k \leq 4k+4$. Furthermore, no cycle-agent can perform an edge-swap and thereby strictly decrease her cost. Since (G'_k, α'_k) has $n = 4k+5$ agents, this implies that we can choose $\alpha'_k = \frac{4k+4}{4k+5}n$. Clearly, since $\lim_{k\to\infty} \frac{4k+4}{4k+5} = 1$, the edge-cost α'_k can get arbitrarily close to the number of agents in the network.

Observe that (G'_k, α'_k) is only in Sum-Greedy Equilibrium but not in Sum-Nash Equilibrium, since agent a could remove both her edges and buy one edge to agent c instead. This strategy-change yields a strict cost decrease for agent a. ∎

5.6. On the Structure of Sum-Nash Equilibria for high α

In this section we provide new insights into the structure of SUM-NE networks, which we believe are promising for a new line of attack on settling the Price of Anarchy for the SUM-NCG.

We start with some easy properties of cycles in a SUM-NE network. Then we introduce our main actors: Min-Cycles and Critical Pairs. Finally we set our actors to work.

Lemma 5.6.1 *If $\alpha > \lfloor \frac{n}{2} \rfloor$, then no* SUM*-GE network can have a cycle of length* 3.

PROOF Assume that there is a network (G, α) which is in SUM-Greedy Equilibrium for some $\alpha > \lfloor \frac{n}{2} \rfloor$ which contains a cycle of length 3. Let this cycle be uvw. There are two cases.

1. Assume that one of the cycle agents owns two cycle-edges. Let this be agent u. In this case both edges must each yield a distance-cost decrease of at least α for agent u. That is, removing the edge uv or uw increases agent u's distance-cost by at least α, respectively. Let A_u^v denote the set of agents to which *all* of agent u's shortest paths in (G, α) use the edge uv. Analogously, let A_u^w be the set of agents to which all of agent u's shortest paths in (G, α) use the edge uw. Removing the edge uv increases agent u's distance to all agents in A_u^v by exactly one. Since we assume that deleting uv does not yield a strict cost decrease for agent u, we have $\alpha \leq |A_u^v|$. Analogously, we obtain that $\alpha \leq |A_u^w|$. Now, notice that $A_u^v \cap A_u^w = \emptyset$, which implies that the set of agents in G can be partitioned into $V = X \cup A_u^v \cup A_u^w$, with $X \cap A_u^v = \emptyset$ and $X \cap A_u^w = \emptyset$. Since $u \in X$, it follows that $|A_u^v| + |A_u^w| \leq n - 1$. Since $\alpha \leq \min\{|A_u^v|, |A_u^w|\}$ we have that $\alpha \leq \lfloor \frac{n}{2} \rfloor$.

2. The second case is if all cycle agents own exactly one cycle edge. W.l.o.g let uv be owned by u, vw be owned by v and wu be owned by w. Thus, $\alpha \leq \min\{|A_u^v|, |A_v^w|, |A_w^u|\}$. We claim that the sets A_u^v, A_v^w and A_w^u are pairwise disjoint. This can be seen as follows: Let $x \in A_u^v \cap A_v^w$. Let $d(v, x) = k$. Since $x \in A_v^w$ we have that all shortest paths from agent v to x must use the edge vw. Thus, $d(w, x) < k - 1$. It follows that $x \notin A_u^v$, since agent u has a shortest path to x of length $j < k + 1$. The other disjointness proofs are analogous. It follows that $|A_v^u| + |A_v^w| + |A_w^u| \leq n$, which implies $\alpha \leq \frac{n}{3} \leq \lfloor \frac{n}{2} \rfloor$. ∎

Lemma 5.6.2 *If $\alpha > 2n-6$, then there is no* SUM*-GE network having a chordless*[3] *cycle of length at most* 4.

[3] A *chord* is an edge which joins two cycle-vertices which are not adjacent in the cycle.

PROOF Let (G, α) be in SUM-Greedy Equilibrium and let C be a chordless cycle of length at most 4 in (G, α). Clearly, by Lemma 5.6.1, we have that C must have length exactly 4.

Let v be a vertex of C who owns at least one edge of C. Let this be the edge vx. Since (G, α) is in SUM-Greedy Equilibrium, it follows that removing the edge vx does not decrease agent v's cost. By this deletion, agent v's edge-cost decreases by α and her distance-cost increases by at most $2(n-3)$, since agent v's distance to all other vertices of C, including herself, does not change and all other distances may increase by 2, since the cycle has length 4. It follows that $\alpha \leq 2n-6$ must hold for (G, α) being in SUM-Greedy Equilibrium.

5.6.1. Min-Cycles

We introduce Min-Cycles and define various structural properties.

Definition 5.6.3 (Min-Cycle) *Let C be a cycle in a network G. We call the cycle C a* Min-Cycle*, if for any two cycle vertices $u \in V(C)$ and $v \in V(C)$ we have that $d_C(u,v) = d_G(u,v)$.*

Remark 5.6.4 *Note that any smallest cycle in a network G is a Min-Cycle of G.*

Definition 5.6.5 (Overlapping Cycles and Well-Connected Cycles) *Let C and C' be two cycles in a network G. We say that the cycles C and C'* overlap *if they share at least one* edge*, that is, if $E(C) \cap E(C') \neq \emptyset$. We call $E(C) \cap E(C')$ the* overlapping edge set *of C and C'. We say that the overlapping cycles C and C' are* well-connected*, if their shared subgraph $C \cap C'$ forms a simple path in G.*

Definition 5.6.6 (Path of Min-Cycles) *A* path of Min-Cycles *from C_1 to C_j is a sequence of different Min-Cycles $C_1, C_2, \ldots, C_j$ such that only neighboring Min-Cycles in the sequence overlap. That is, Min-Cycle C_i overlaps with C_{i-1} and C_{i+1}, for $2 \leq i \leq j-1$, but not with any other Min-Cycle from the sequence. The length of a path of Min-Cycles is the number of cycles in the sequence minus* 1*.*

Definition 5.6.7 (Connected Set of Min-Cycles) *Let $\mathcal{C} = \{C_1, \ldots, C_l\}$ be a set of Min-Cycles of a graph G. We say that $\mathcal{C}$ is* connected*, if for every pair $C_i, C_j \in \mathcal{C}$ there is a path of Min-Cycles from C_i to C_j which entirely consists of Min-Cycles from $\mathcal{C}$.*

Lemma 5.6.8 *Let C be a cycle in G which contains the edge uv and uw. There is a Min-Cycle C' which contains the edge uv and there is a Min-Cycle C'', which contains the edge uw. Furthermore, there exists a connected set of Min-Cycles $\mathcal{C}$ which contains C' and C'' and all Min-Cycles in $\mathcal{C}$ contain vertex u.*

PROOF If C itself is a Min-Cycle, then we are trivially done by setting $\mathcal{C} = \{C\}$. Thus, assume that C is not a Min-Cycle, that is, there must be a pair of vertices whose shortest path on C is strictly longer than their shortest path in G.

First, let us assume that there is a pair $x, y \in V(C)$, where $d_G(x,y) < d_C(x,y)$ and whose shortest path P^{xy} in G does *not* contain vertex u. W.l.o.g. let x be closer than y to v on the path $C - \{u\}$. Let P_C^{ij} be a shortest path from i to j on the cycle C. Note that $x \neq u$ and $y \neq u$ must hold since vertex u is not on the path P^{xy}. In this case we can replace the cycle C by the strictly shorter cycle

$$\tilde{C} = uv \cup P_C^{vx} \cup P^{xy} \cup P_C^{yw} \cup uw \ ,$$

which contains the edges uv and uw. See Fig. 5.5 (left).

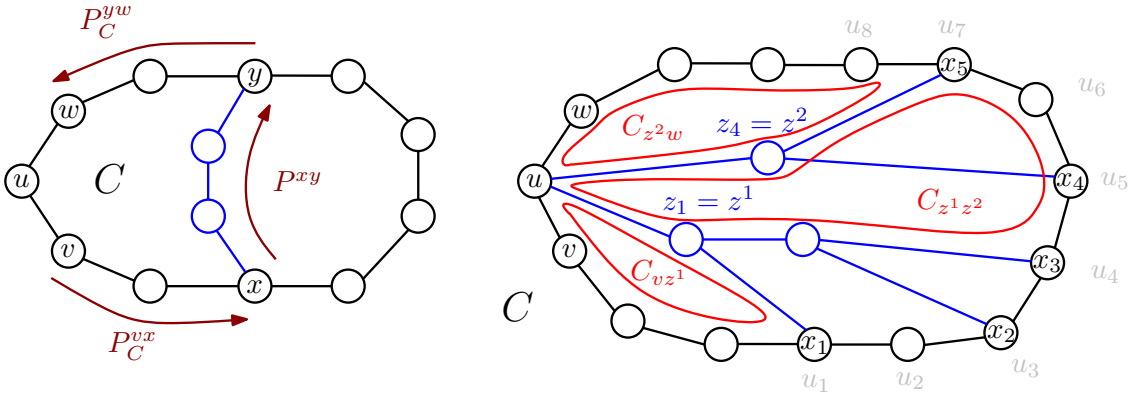


Figure 5.5.: Left: A pair x, y having a shortest path outside C which does not contain u. Right: The pairs u, x_i and the smaller cycles $C_{vz^1}, \ldots, C_{z^k w}$. In this example we have $l(z^1) = x_1, h(z^1) = x_3$ and $h(z^2) = x_5$.

Thus, we assume in the following that no such pair x, y exists. Having ruled out such pairs the only possibility for C not being a Min-Cycle is that there is a set of vertices $U = \{u_1, \ldots, u_l\} \subset V(C)$, where $d_G(u, u_i) < d_C(u, u_i)$ for all i. Among those vertices U, there must be a subset of vertices $X \subseteq U$, which have a shortest path to vertex u which does not use any edge of the cycle C. Let $x_i \in X$ be the vertex which is the i-th closest vertex to v on the path from v to w in $C - u$. See Fig. 5.5 (right).

Let P^{ux_i} be a shortest path in G from u to x_i and let z_i be the neighbor of u on P^{ux_i}. Note that $z_i = x_i$ is possible and that $z_i \notin V(C) \setminus \{x_i\}$, that is, $z_i \neq v$ and $z_i \neq w$, since otherwise the pair v, x_i or w, x_i has a shortest path which does not contain vertex u.

The ordering $x_1, \ldots, x_i$ induces an ordering for the z_i-vertices as follows: Let z_1 be the neighbor of u on P^{ux_1}. Let $Z_1 \subseteq X$ be the set of vertices from X which all have z_1 on their shortest path to vertex u in G. We set $X' = X \setminus Z_1$ and let l be the smallest index in X'. Vertex z_l is the neighbor of u on P^{ux_l} and the set $Z_l \subset X'$ is defined analogously to Z_1. We remove Z_l from X' and iterate this process on the

remaining x_i-vertices until all of them are assigned to a z_j-vertex. For any z_j-vertex, let $s(z_j)$ be the x_i vertex with the smallest index in the set Z_j. Furthermore, let $h(z_j)$ be the x_i-vertex having the highest index in Z_j. Let Y be the set of z_j-vertices and we can order it increasingly by their index. Let $z^1, z^2, \ldots, z^k$ be this ordering of Y. The vertices in Y "split" the cycle C into a sequence of $k+1$ smaller cycles

$$S_C = C_{vz^1}, C_{z^1z^2}, C_{z^2z^3}, \ldots, C_{z^{k-1}z^k}, C_{z^kw} \ .$$

They are defined as follows:

$$C_{vz^1} = P_C^{us(z^1)} \cup P^{us(z^1)}, \quad C_{z^qz^{q+1}} = P^{uh(z^q)} \cup P_C^{h(z^q)h(z^{q+1})} \cup P^{uh(z^{q+1})} \ ,$$

for all $1 \leq q \leq k-1$, and

$$C_{z^kw} = P^{uh(z^k)} \cup uw \cup P_C^{wz^k} \ ,$$

where $P_C^{wz^k}$ is the w-z^k path along cycle C which does not contain the edge uw. See Fig. 5.5 (right) for an illustration. Observe that all cycles are strictly shorter than cycle C and that all cycles contain vertex u. Furthermore, the cycle C_{vz^1} contains the edge uv and the cycle C_{z^kw} contains the edge uw. Moreover, each neighboring pair of cycles in the sequence S_C overlaps in at least one edge.

For all those $k+1$ cycles we do the following: If the cycle C_{ab} is a Min-Cycle, then we set $\mathcal{C} = \mathcal{C} \cup \{C_{ab}\}$. If not, then we proceed recursively for this cycle C_{ab} until we have found a Min-Cycle containing the edge ua and a Min-Cycle containing the edge ub and connected set of Min-Cycles $\mathcal{C}_{ab}$ which contains both of them. We set $\mathcal{C} = \mathcal{C} \cup \mathcal{C}_{ab}$. Note that the recursion must terminate, because the considered cycles become strictly smaller in every iteration. Thus we get a sequence of sets of Min-Cycles

$$\mathcal{S}_C = \mathcal{C}_{vz^1}, \mathcal{C}_{z^1z^2}, \mathcal{C}_{z^2z^3}, \ldots, \mathcal{C}_{z^{k-1}z^k}, \mathcal{C}_{z^kw} \ ,$$

where each set in the sequence is a connected set of Min-Cycles and for every two neighboring sets $\mathcal{C}_{z^{i-1}z^i}$ and $\mathcal{C}_{z^iz^{i+1}}$ there are Min-Cycles $C' \in \mathcal{C}_{z^{i-1}z^i}$ and $C'' \in \mathcal{C}_{z^iz^{i+1}}$ which overlap in the edge uz^i.

Thus, we get a connected set of Min-Cycles

$$\mathcal{C} = \mathcal{C}_{vz^1} \cup \mathcal{C}_{z^1z^2} \cup \mathcal{C}_{z^2z^3} \cup \cdots \cup \mathcal{C}_{z^{k-1}z^k} \cup \mathcal{C}_{z^kw} \ ,$$

which contains a Min-Cycle having the edge uv and a Min-Cycle having the edge uw and all Min-Cycles in $\mathcal{C}$ contain vertex u by construction. ∎

Definition 5.6.9 (Min-Cycle Cover) *Let G be a graph and let G_{BC} be a biconnected component of G. A* Min-Cycle cover *of G_{BC} is a connected set of Min-Cycles $\mathcal{C} = \{C_1, \ldots, C_l\}$ such that every edge of G_{BC} is contained in at least one Min-Cycle from $\mathcal{C}$.*

Corollary 5.6.10 *Every biconnected component G_{BC} of a network G has a Min-Cycle cover $\mathcal{C}$.*

PROOF The Min-Cycle cover $\mathcal{C}$ can be constructed as follows. Since G_{BC} is a connected component, it must contain a shortest cycle C. By definition, any shortest cycle in a network must be a Min-Cycle as well.[4] We set $\mathcal{C} = \{C\}$. Let $E(\mathcal{C}) = \bigcup_{C_i \in \mathcal{C}} E(C_i)$ and let $V(\mathcal{C}) = \bigcup_{C_i \in \mathcal{C}} V(C_i)$. Clearly, we have that $E(\mathcal{C}) \subseteq E(G_{BC})$ and in the following this property will be invariant.

We iterate the following process until $E(\mathcal{C}) = E(G_{BC})$ holds: If $E(\mathcal{C}) \neq E(G_{BC})$, then, since G_{BC} is connected, there must be a non-bridge edge uw, where $u \in V(\mathcal{C})$ and $w \in V(G_{BC}) \setminus V(\mathcal{C})$. This edge uw is not yet covered by a Min-Cycle. By definition, vertex u cannot be a cut-vertex and u is contained in at least one Min-Cycle $C_u \in \mathcal{C}$. Let v be a neighbor of u in the Min-Cycle C_u. Since uw is not a bridge and since u is not a cut-vertex, there must be a cycle C which contains the edges uv and uw. By Lemma 5.6.8, G_{BC} must have a Min-Cycle C' which contains the edge uv and a Min-Cycle C'' which contains the edge uw and a connected set of Min-Cycles $\mathcal{C}_{vw}$ which contains C' and C''. Note that the cycles C_u and C' overlap in edge uv. Thus, by setting $\mathcal{C} = \mathcal{C} \cup \mathcal{C}_{vw}$ we obtain a larger connected set of Min-Cycles, which now contains the edge uw. ∎

Definition 5.6.11 (Cycle, Wheel and Ring of Min-Cycles) *A* cycle of Min-Cycles *is a path of Min-Cycles $C = C_1, C_2, \ldots, C_k, C_1$, where $k > 2$ and where only neighboring Min-Cycles in the sequence overlap. If the cycle contains an odd (even) number of Min-Cycles, then we say that the cycle of Min-Cycle is odd (even).*

If all Min-Cycles in a cycle of Min-Cycles C contain the same vertex u, then we call C a wheel of Min-Cycles*. Otherwise we call C a* ring of Min-Cycles*. See Fig. 5.6 for an illustration.*

Remark 5.6.12 *Observe that we explicitly rule out cycles of Min-Cycles of length two. In such a cycle $\mathcal{C} = \{C, C'\}$, we have that the Min-Cycles C and C' cannot be well-connected, since $C \cap C'$ consists of at least two vertex disjoint paths. We will show later that we can assume that all overlapping Min-Cycles must be well-connected.*

Lemma 5.6.13 *Let C_1 and C_2 be two well-connected Min-Cycles in a network G and let P be their shared path, that is, $P = C_1 \cap C_2$. If $V(P)$ is not a separator of G, then G contains a cycle of Min-Cycles.*

PROOF If $V(P)$ is not a separator of G, then there must be vertices $a \in V(C_1) \setminus V(P)$ and $b \in V(C(w) \setminus V(P))$ and a shortest path P_{ab} in G, which does not use any edge of C_1 or C_2. Let P_{ab} have the vertex sequence $a, z_1, z_2, \ldots, z_k, b$, for some

[4] Note that the opposite is not true in general.

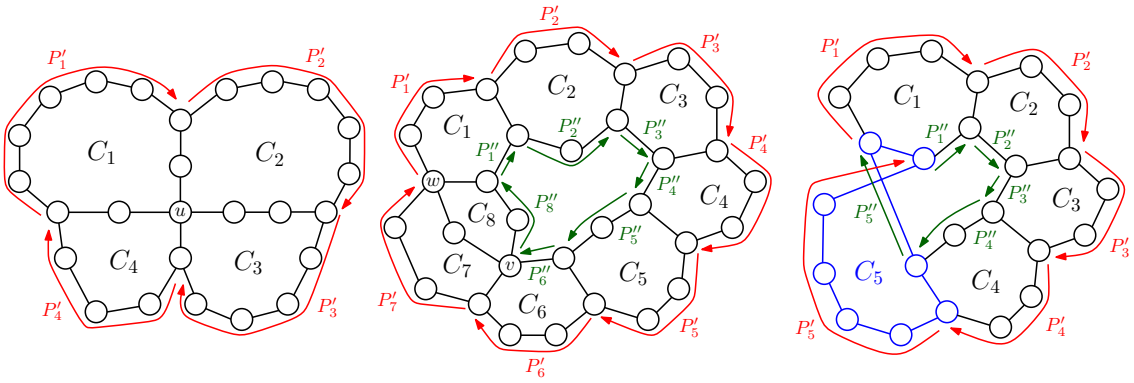


Figure 5.6.: Left: An even wheel of Min-Cycles. The paths P''_1, P''_2, P''_3 and P''_4 all consist only of the single vertex u. Middle: An even ring of Min-Cycles having no twist. Note that the path P''_7 consists only of vertex v and path P'_8 consists only of vertex w. Right: An odd ring of Min-Cycles having a twist. All shown overlapping Min-Cycles are well-connected.

k, and we have that $z_i \notin V(C_1) \cup V(C_2)$ for $1 \leq i \leq k$. Let p be one endpoint of path P. Let Q_{ap} be the path from a to p on C_1 which does not use any edge of P and let Q_{pb} be the path from p to b on C_2 which does not use any edge of P. Let aa' be the first edge on Q_{ap} and let $b'b$ be the last edge on Q_{pb}. Let $R_{ab} = Q_{ap} \cup Q_{pb}$.

We will proceed along the path P_{ab} and we will iteratively use Lemma 5.6.8 until we have found a cycle of Min-Cycles in G. We start with the cycle $P_{ab} \cup R_{ab}$ and we consider the edges az_1 and aa'. By Lemma 5.6.8 there must be a set of Min-Cycles $\mathcal{C}^{a'z_1}$ which contains a Min-Cycle $C_{aa'}$ which contains edge aa' and a Min-Cycle C_{az_1} which contains edge az_1. Moreover, $C_{aa'}$ and C_{az_1} are connected via a path of Min-Cycles from $\mathcal{C}^{a'z_1}$. Let q_1 be the other neighbor of z_1 in C_{az_1}. We proceed with the edges z_2z_1 and z_1q_1, which both are part of a cycle in G and obtain a set of Min-Cycles connecting $C_{aa'}$ and a Min-Cycle which contains the edge z_1z_2. We iterate until we have a set of Min-Cycles connecting $C_{aa'}$ with a Min-Cycle C_{z_kb}, which contains the edge z_kb. In a last step, we consider the edges z_kb and bb' and apply Lemma 5.6.8 again to obtain a set of Min-Cycles connecting $C_{aa'}$ and a Min-Cycle $C_{bb'}$ which contains edge bb'. If we have found a cycle of Min-Cycles with this process, then we are already done. Otherwise, observe that $C_{aa'}$ overlaps with C_1 and that $C_{bb'}$ overlaps with C_2. Thus, we have found a cycle of Min-Cycles $C_1, C_{aa'}, \ldots, C_{az_1}, \ldots, C_{z_kb}, \ldots, C_{bb'}, C_2$. ∎

Definition 5.6.14 (Twist in a Cycle of Min-Cycles) *Let $\mathcal{C} = C_1, \ldots, C_k$ be a cycle of Min-Cycles in a network G. For each cycle $C_i \in \mathcal{C}$ there exist two vertex-disjoint paths P'_i and P''_i on the cycle C_i, which are as short as possible and which connect $C_{i-1 \bmod k}$ and $C_{i+1 \bmod k}$. We say that $\mathcal{C}$ has a* twist *if $P'_1 \cup P''_1 \cup P'_2 \cup P''_2 \cup \cdots \cup P'_k \cup P''_k$ forms one cycle in G. Otherwise $P'_1 \cup P''_1 \cup P'_2 \cup P''_2 \cup \cdots \cup P'_k \cup P''_k$ either forms two vertex-disjoint cycles in G or one cycle an one separate vertex and we say that $\mathcal{C}$ has no twist. See Fig. 5.6.*

Remark 5.6.15 *Note that wheels of Min-Cycles cannot have a twist. This is true since the union of all P_i' and P_i'' paths must form a cycle and a separate vertex, where the separate vertex is the one vertex which is contained in all Min-Cycles.*

Lemma 5.6.16 *Let G be a network where all overlapping Min-Cycles are well-connected. If G contains a cycle of Min-Cycles, then G contains a wheel of Min-Cycles.*

Proof Let $\mathcal{C} = C_1, \ldots, C_k$ be the cycle of Min-Cycles in G. Since we assume that all overlapping Min-Cycles in G are well-connected, it follows that $k \geq 3$. Clearly, if $\mathcal{C}$ is a wheel of Min-Cycles, then we are done. Hence, we assume in the following that $\mathcal{C}$ is a ring of Min-Cycles.

Let C_p and C_{p+1} be neighboring Min-Cycles in $\mathcal{C}$ and let $P = C_p \cap C_{p+1}$ be their shared path in G. Let u be an endpoint of P and let $\mathcal{C}_u = \{C_q \mid C_q \in \mathcal{C} \text{ and } u \in V(C_q)\}$ be the subset of Min-Cycles in $\mathcal{C}$ which all contain vertex u. If $\mathcal{C}_u$ is a cycle of Min-Cycles in G, then we have found a wheel of Min-Cycles, since each Min-Cycle in $\mathcal{C}_u$ contains vertex u. Thus, we can assume that $\mathcal{C}_u$ contains a maximal path of Min-Cycles $\mathcal{P}_u \subset \mathcal{C}_u$. Let C_i and C_j be the first and the last Min-Cycle on the path $\mathcal{P}_u$.

We claim that we can choose neighbors $v \in V(C_i)$ and $w \in V(C_j)$ of vertex u such that $v, w \notin V(P)$, $v \in V(C_i) \setminus V(C_j)$ and $w \in V(C_j) \setminus V(C_i)$ holds. To see this, notice that u has two neighbors in C_i and two neighbors in C_j and exactly one neighbor in P. Thus, if one neighbor in C_i or C_j belongs to $V(P)$, then we can choose the respective other neighbor. Hence, we have satisfied that $v, w \notin V(P)$. Moreover, if we assume that $v \in V(C_i) \cap V(C_j)$, then, since both v and w are neighbors of u, it follows that either $v = w$, which implies that u is not an endpoint of P or that $\mathcal{C}_u$ is a wheel of Min-Cycles, or, if $v \neq w$ and $v \in V(C_i) \cap V(C_j)$, it follows that C_i and C_j overlap in the edge uv, which again implies that $\mathcal{C}_u$ is a wheel of Min-Cycles. The argument for the case $w \in V(C_i) \cap V(C_j)$ is analogous.

With our choice of u, v and w we are in a situation where we have a path of Min-Cycles $\mathcal{P}_u$ where all Min-Cycles of the path contain vertex u and the first Min-Cycle on $\mathcal{P}_u$, w.l.o.g. Min-Cycle C_i, contains the edge uv and the last Min-Cycle on $\mathcal{P}_u$, w.l.o.g. Min-Cycle C_j, contains the edge uw. Now, since $\mathcal{C}$ is a cycle of Min-Cycles and since $\mathcal{P}_u \subset \mathcal{C}$, there must exist a cycle C^* in G which contains both edges uv and uw and which does not contain any edge from any Min-Cycle in $\mathcal{P}_u \setminus \{C_i, C_j\}$.

We can apply Lemma 5.6.8 on cycle C^* and the vertices u, v, w and get a connected set of Min-Cycles $\mathcal{C}'$, where all Min-Cycles in $\mathcal{C}'$ contain vertex u. If $\mathcal{C}'$ is a path of Min-Cycles, then this implies that $\mathcal{P}_u \cup \mathcal{C}'$ is a wheel of Min-Cycles or that there are two overlapping Min-Cycles $C_r \in \mathcal{P}_u$ and $C_s \in \mathcal{C}'$ which are not well-connected, which contradicts our assumption. On the other hand, if $\mathcal{C}'$ is not a path of Min-Cycles, then, since all Min-Cycles in $\mathcal{C}'$ contain vertex u, it follows that there are two Min-Cycles $C_r, C_s \in \mathcal{C}'$ which are not well-connected and we have a contradiction. ∎

5.6.2. Critical Pairs

Definition 5.6.17 (Critical Pair) *Let (G, α) be a network and let v be an agent of (G, α) who owns two non-bridge edges va and vb. Let u be another agent in (G, α) who owns at least one edge towards $x \neq v$. We say that u and v form a* Critical Pair *if the following four properties hold.*

(1) $d_G(v, u) = k \geq 2$.

(2) Agent u has a shortest path to vertex a which does not use the edge va.

(3) Agent u has a shortest path to vertex b which does not use the edge vb.

(4) Agent v has a shortest path to vertex x which does not use the edge ux.

See Fig. 5.7 (left) for an illustration.

The next theorem shows that Critical Pairs are indeed a powerful tool for analyzing Sum-Nash Equilibria.

Theorem 5.6.18 *If $\alpha > 2n - 6$, then no* Sum*-NE network can have a Critical Pair.*

Proof Let v be an agent who owns two non-bridge edges of network (G, α) towards a and b and let u be another agent who owns at least one edge towards $x \neq v$. Let v and u form a Critical Pair in (G, α). Furthermore, we assume that (G, α) is in Sum-Nash Equilibrium.

We will first consider a possible strategy-change performed by agent u. Let (G', α) be the network obtained from (G, α) if agent u removes the edge ux and buys the edge uv. See Fig. 5.7 (middle) for an illustration.

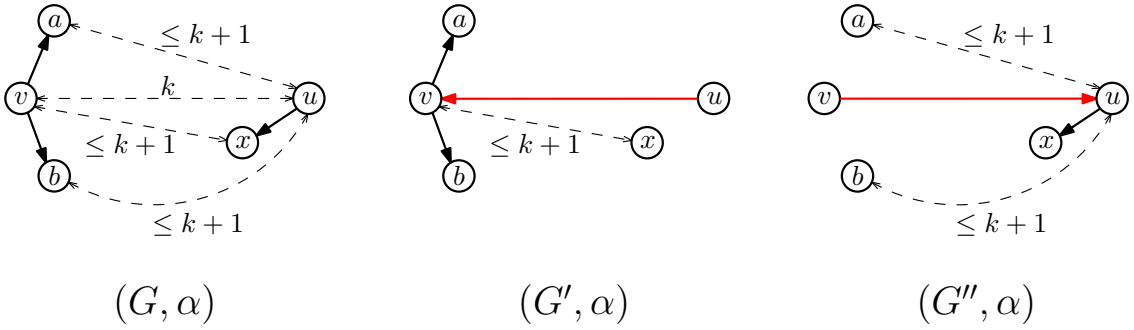


Figure 5.7.: Illustration of a Critical Pair u, v and possible strategy-changes by agent u and agent v.

Since we assume that (G, α) is in Sum-Nash Equilibrium, it follows that agent u's cost in (G', α) cannot be less than her cost in (G, α). Thus, we have

$$c_u(G, \alpha) - c_u(G', \alpha) \leq 0 \ . \tag{5.1}$$

We claim that agent u's strategy-change does not increase agent v's distance-cost, that is, $\delta_v(G') \leq \delta_v(G)$. This is true, since v and u form a Critical Pair. We have that all shortest paths of agent v in (G, α) which traverse vertex x but not vertex u are still present in (G', α). All shortest paths of agent v in (G, α) which first traverse x and then u are strictly shorter in network (G', α). Finally, for each shortest path of v in (G, α) which first traverses u and then x there exists another shortest path in (G, α) which only traverses x but not u and which has the same length. Thus, this path exists in network (G', α) as well.

We can upper bound agent u's cost in network (G', α) as follows:

$$c_u(G', \alpha) \leq e_u(G, \alpha) + n - 1 + \delta_v(G') - 1 \ ,$$

since $e_u(G', \alpha) = e_u(G, \alpha)$ and

$$\delta_u(G') = \sum_{z \in V(G')} d_{G'}(u, z) \leq \sum_{z \in V(G') \setminus \{u\}} \Big(1 + d_{G'}(v, z)\Big) = n - 1 + \delta_v(G') - 1 \ .$$

Since agent u's strategy-change does not increase agent v's distance to any vertex, we have

$$\begin{aligned} \delta_v(G') - 1 = \delta_v(G') - d_{G'}(u, v) &= \sum_{z \in V(G') \setminus \{u\}} d_{G'}(v, z) \\ &\leq \sum_{z \in V(G) \setminus \{u\}} d_G(v, z) = \delta_v(G) - d_G(u, v) \ . \end{aligned}$$

This implies that

$$c_u(G', \alpha) \leq e_u(G, \alpha) + n - 1 + \delta_v(G) - d_G(u, v) \ .$$

Together with inequality (5.1), this yields

$$c_u(G, \alpha) - \Big(e_u(G, \alpha) + n - 1 + \delta_v(G) - d_G(u, v)\Big) \leq c_u(G, \alpha) - c_u(G', \alpha) \leq 0 \ ,$$

which, since $c_u(G, \alpha) = e_u(G, \alpha) + \delta_u(G)$, gives

$$\delta_u(G) - n + 1 - \delta_v(G) + d_G(u, v) \leq 0 \ .$$

Thus, we have the following upper bound for $\delta_u(G)$:

$$\delta_u(G) \leq \delta_v(G) + n - d_G(u, v) - 1 \ . \tag{5.2}$$

Now let us consider a possible strategy-change by agent v in network (G, α). Suppose that agent v removes both her edges towards a and b and buys the edge towards u instead. Let (G'', α) be the network induced by this strategy-change. See Fig. 5.7 (right).

Since (G, α) is in Sum-Nash Equilibrium, we have that agent v cannot have strictly less cost in network (G'', α) compared to her cost in network (G, α), that is,

$$c_v(G, \alpha) - c_v(G'', \alpha) \leq 0 \ . \tag{5.3}$$

We claim that $\delta_u(G'') \leq \delta_u(G)$. This is true, since v and u form a Critical Pair. All shortest paths from u in (G, α) which traverse vertex a but not vertex v are still present in network (G'', α). The same holds true for all shortest paths of agent u in (G, α) which traverse vertex b but not vertex v. Moreover, all shortest paths of u in (G, α) which first traverse a and then traverse v or which first traverse b and then traverse v must be strictly shorter in network (G'', α). Finally, for each shortest path of u in (G, α) which first traverses v and then a or b, there must be another path in (G, α) which only traverses a or b but not v and which has the same length. This path is present in network (G'', α).

Analogous to above, we can upper bound agent v's cost in network (G'', α) as follows:

$$c_v(G'', \alpha) \leq e_v(G, \alpha) - \alpha + n - 1 + \delta_u(G'') - 1 \ .$$

Here we have used that $e_v(G'', \alpha) = e_v(G, \alpha) - \alpha$ because v's strategy-change removes two edges and buys only one edge. Since v's strategy-change does not increase agent u's distance to any vertex, we have, by an analogous argument as above, that $\delta_u(G'') - 1 \leq \delta_u(G) - d_G(u, v)$, which implies

$$c_v(G'', \alpha) \leq e_v(G, \alpha) - \alpha + n - 1 + \delta_u(G) - d_G(u, v) \ .$$

Together with inequality (5.3) this yields

$$c_v(G, \alpha) - \Big(e_v(G, \alpha) - \alpha + n - 1 + \delta_u(G) - d_G(u, v)\Big) \leq c_v(G, \alpha) - c_v(G'', \alpha) \leq 0 \ ,$$

which gives

$$\delta_v(G) + \alpha - n + 1 - \delta_u(G) + d_G(u, v) \leq 0 \ .$$

From this we get an upper bound for α:

$$\alpha \leq \delta_u(G) - \delta_v(G) + n - 1 - d_G(u, v). \tag{5.4}$$

Inequality (5.4) together with inequality (5.2) yields

$$\begin{aligned} \alpha &\leq \delta_v(G) + n - d_G(u, v) - 1 - \delta_v(G) + n - 1 - d_G(u, v) \\ &= 2n - 2d_G(u, v) - 2 \\ &\leq 2n - 6 \ , \end{aligned}$$

where the last inequality follows from the assumption that $d_G(u, v) \geq 2$. Thus, if $\alpha > 2n - 6$, then we have that agent u or agent v can perform a strategy-change which strictly decreases her cost, which contradicts the assumption that (G, α) is in Sum-Nash Equilibrium. ∎

5.6.3. The Relation between Min-Cycles and Critical Pairs

We begin with showing that if there are no Critical Pairs in a network (G, α), then the edge-ownership within a Min-Cycle in (G, α) must be very specific.

Lemma 5.6.19 *Let C be a Min-Cycle having length at least 5 in a network (G, α). If an agent $v \in V(C)$ owns two edges of the Min-Cycle C, then (G, α) contains a Critical Pair.*

Proof Let C be a Min-Cycle in network (G, α) and let $|V(C)| \geq 5$. Furthermore, let $v \in V(C)$ own two edges of the Min-Cycle towards a and b.

If $|V(C)|$ is even, then there is a unique vertex $z \in V(C)$ having maximum distance $k \geq 3$ towards v. By definition of a Min-Cycle, we have that $d_G(z, a) = d_G(z, b) = k - 1$. Let x and y be the cycle-neighbors of z in C and, by definition of a Min-Cycle, we have that $d_G(v, x) = d_G(v, y) = k - 1$. If z owns an edge of C, then we have that v and z form a Critical Pair. If z does not own an edge of C, then x must own the edge xz. Since $k \geq 3$, we have that $d_G(x, v) \geq 2$. W.l.o.g. let a be the unique vertex of C which has maximum distance to x. We have $d_G(x, a) \leq d_G(x, v) + 1$ and we have that there is a shortest path from x to a which does not traverse v. It follows that v and x form a Critical Pair.

If $|V(C)|$ is odd, then there are exactly two vertices $x, y \in V(C)$ which have maximum distance $k \geq 2$ towards v. The edge xy must have an owner and w.l.o.g. let this be agent x. In this case, it is easy to see that v and x form a Critical Pair. ■

Definition 5.6.20 (Non-Critical Network) *We call a network (G, α)* non-critical *if there is no Critical Pair in (G, α).*

Corollary 5.6.21 *If (G, α) is non-critical for some $\alpha > 2n - 6$, then we have for every Min-Cycle C in (G, α) that $\overrightarrow{C}$ must be a directed cycle.*

Proof Recall that $\overrightarrow{C}$ is the cycle C where every edge is replaced by a directed arc which points away from the owner of the respective edge.

Since $\alpha > 2n - 6$ and Lemma 5.6.2, we have all cycles in (G, α) must have length at least 5. Thus, assume that for some Min-Cycle C in (G, α) we have that $\overrightarrow{C}$ is not a directed cycle. Then, by the pigeonhole principle, there must be an agent who owns exactly two edges of C and by Lemma 5.6.19, it follows that (G, α) must contain a Critical Pair. ■

Since we focus on high edge-cost $\alpha > 2n - 6$ and having Lemma 5.6.19, we may assume in the following that every agent in a Min-Cycle owns exactly one edge of the Min-Cycle. The next observation yields that overlapping Min-Cycles in a non-critical network must overlap in exactly one path.

Lemma 5.6.22 *Let C_1 and C_2 be two overlapping Min-Cycles in a network (G, α), let every agent in C_i own exactly one edge of C_i. If C_1 and C_2 are not well-connected, then (G, α) contains a Critical Pair.*

Proof Clearly, since agents of C_i own exactly one edge of C_i, both $\overrightarrow{C_1}$ and $\overrightarrow{C_2}$ must be directed cycles. Let $uv \in E(C_1) \cap E(C_2)$ be a shared edge and let P be a maximal path of shared edges which contains uv. Let $\overrightarrow{P} = a_1, a_2, \ldots, a_l$ be the directed version of path P, where the arcs (a_j, a_{j+1}) are directed from a_j to a_{j+1}, for $1 \leq j \leq l-1$. Let $X = (V(C_1) \cap V(C_2)) \setminus V(P)$. Since we assume that C_1 and C_2 are not well-connected, it follows that $X \neq \emptyset$. Hence, let $x \in X$ be the vertex in X which has the smallest distance on C_1 to any vertex in $V(P)$. There are two cases:

1. If $d_{C_1}(x, a_l) \leq d_{C_1}(x, a_1)$, then let $\overrightarrow{P_1}$ be the directed path on $\overrightarrow{C_1}$ from a_l to x and let $\overrightarrow{P_2}$ be the path on $\overrightarrow{C_2}$ from a_l to x. Clearly, $d_{C_1}(x, a_l) \geq 2$, since otherwise there would be a multi-edge in (G, α). Let p be the neighbor of a_l on $\overrightarrow{P_1}$ and let q be the neighbor of a_l on $\overrightarrow{P_2}$. See Fig. 5.8 (left). Note that both $\overrightarrow{P_1}$ and $\overrightarrow{P_2}$ must be directed from a_l to x. This is true, since the edge $a_{l-1}a_l$ is owned by a_{l-1} and is contained in both directed cycles.

   The next observation is that $|\overrightarrow{P_1}| = |\overrightarrow{P_2}|$. This holds because otherwise either C_1 or C_2 cannot be a Min-Cycle in C since there is a shorter path between a_l and x which is not on the respective cycle.

   Let $\overrightarrow{P_3}$ be the directed path from x to a_1 on $\overrightarrow{C_1}$. Since $d_{C_1}(x, a_l) \leq d_{C_1}(x, a_1)$, we have that $|\overrightarrow{P_3}| \geq |\overrightarrow{P_1}|$ and that $|\overrightarrow{P_3}| + |\overrightarrow{P}| > |\overrightarrow{P_1}|$, since $\overrightarrow{P}$ contains at least the edge uv. Let $\overrightarrow{P'} = \overrightarrow{P_3} + \overrightarrow{P}$ and let z be a vertex on C_1 which has maximum distance to a_l on C_1. Note that since $|\overrightarrow{P_1}| \geq 2$, $|\overrightarrow{P_3}| \geq |\overrightarrow{P_2}|$ and $|\overrightarrow{P}| \geq 1$, we have that $d_{C_1}(a_l, z) \geq 2$.

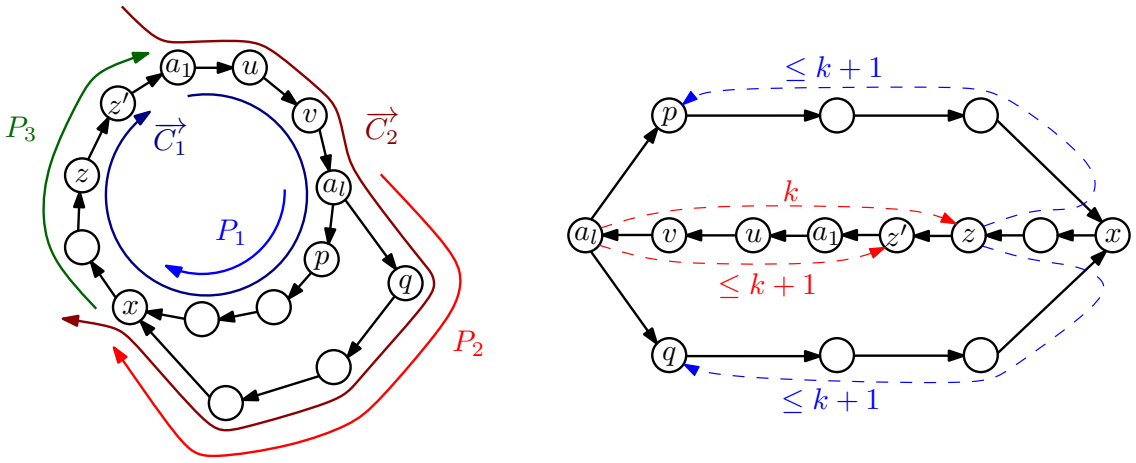


Figure 5.8.: Left: The setting in case 1. Right: Illustration of the Critical Pair a_l, z.

   We claim that the pair a_l, z forms a Critical Pair in (G, α). This can be seen as follows: Let $d_{C_1}(a_l, z) = k \geq 2$. Clearly, since both a_l and z are contained in

the Min-Cycle C_1, we have that there is no shorter a_l-z-path in (G, α), that is, $d_G(a_l, z) = k$. Let z' be the neighbor of z on C_1 to which z owns an edge. Since the path $\overrightarrow{P'}$ in $\overrightarrow{G}$ is directed from x to a_l, since $V(\overrightarrow{P'}) \subset V(C_1)$ and by choice of z, we have that $d_{C_1}(z', a_l) \leq d_{C_1}(z, a_l) = k$ and that a_l has a shortest path to z' in (G, α) of length less than $k+1$ which does not use the edge zz'. Furthermore, by choice of z, agent z has shortest paths to p and q which have length at most $k+1$ and which do not use the edge $a_l p$ or $a_l q$. See Fig. 5.8 (right).

2. If $d_{C_1}(x, a_l) > d_{C_1}(x, a_1)$, then let $\overrightarrow{P_1}$ be the on $\overrightarrow{C_1}$ from x to a_1 and let $\overrightarrow{P_2}$ be the path on $\overrightarrow{C_2}$ from x to a_1. See Fig. 5.9 (left). Analogous to case 1 we have that both paths must be directed from x to a_1 that $|\overrightarrow{P_1}| = |\overrightarrow{P_2}|$ and that $|\overrightarrow{P_1}| \geq 2$. Let p be the neighbor of x on $\overrightarrow{P_1}$ and let q be the neighbor of x on $\overrightarrow{P_2}$. Let $\overrightarrow{P_3}$ be the directed path from a_l to x on $\overrightarrow{C_1}$ and we have $|\overrightarrow{P_3}| > |\overrightarrow{P_1}|$. Let $\overrightarrow{P'} = \overrightarrow{P} + \overrightarrow{P_3}$ and let z be a vertex on C_1 having maximum distance to x on C_1 and analogous to case 1 we have $d_{C_1}(x, z) \geq 2$.

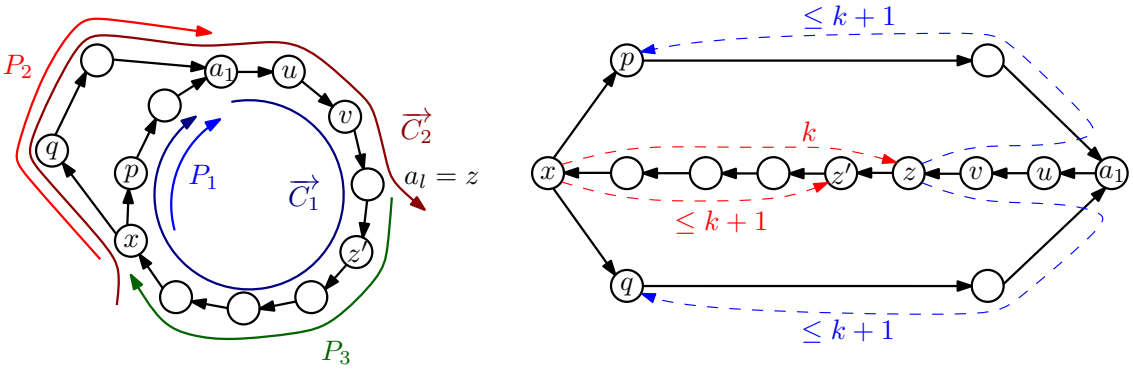


Figure 5.9.: Left: The setting in case 2. Right: Illustration of the Critical Pair x, z.

We claim that x, z forms a Critical Pair in (G, α). The proof is analogous to case 1: Since C_1 is a Min-Cycle, we have $d_{C_1}(x, z) = d_G(x, z) = k \geq 2$. Let z' be the neighbor of z on C_1 to which z owns an edge. By choice of z and since $V(\overrightarrow{P'}) \subset V(C_1)$, there is a shortest path in (G, α) from x to z' which has length at most $k+1$ and which does not use the edge zz'. Moreover, again by choice of z, agent z has shortest paths to p and q which have length at most $k+1$ and which do not use the edges xp or xq. See Fig. 5.9 (right).

In both cases we have established that (G, α) contains a Critical Pair. ∎

Remark 5.6.23 *Note that by Lemma 5.6.22 we have that there are no cycles of Min-Cycles of length* 2 *unless there exists a Critical Pair.*

We assume in the following that if two Min-Cycles overlap, then they overlap in exactly one path. The next result restricts the length of such a path.

Lemma 5.6.24 *Let C_1 and C_2 be two overlapping Min-Cycles in a network (G, α), where no agent in C_i owns two edges of C_i and $|C_i| \geq 5$. Let P be the maximal path of shared edges. If $|P| \geq \min\left\{\frac{|C_1|}{2} - 1, \frac{|C_2|}{2} - 1\right\}$, then (G, α) contains a Critical Pair.*

Proof We assume that $|C_1| \leq |C_2|$ and let $P = a_1, \ldots, a_l$ such that agent a_i owns the edge $a_i a_{i+1}$ for $1 \leq i \leq l-1$. We show that there exists a Critical Pair, if $|P| = l - 1 \geq \frac{|C_1|}{2} - 1$. Let P_1 be the path along C_1 from a_l to a_1 which does not use the edge $a_l a_{l-1}$ and let P_2 be the path along C_2 from a_l to a_1 which does not use the edge $a_l a_{l-1}$. Let p be the neighbor of a_l on P_1 and let q be the neighbor of a_l on P_2. Since $P \cup P_j = C_j$, for $j \in \{1, 2\}$, we have $|P_1| \leq |P_2|$. See Fig. 5.10 (left).

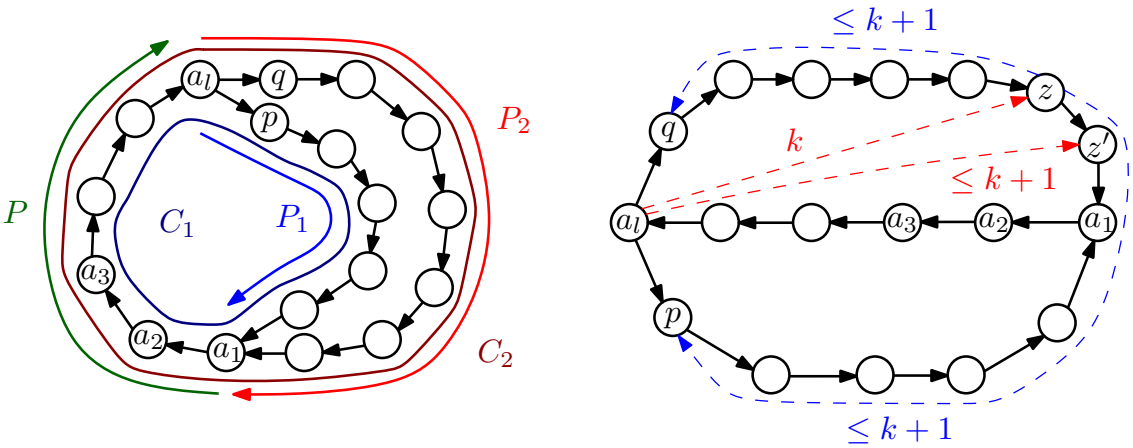


Figure 5.10.: Left: The overlapping cycles C_1 and C_2. Right: Illustration of the Critical Pair a_l, z in the case $|P_2| > |P_1|$.

There are two cases:

1. If $|P_1| = |P_2|$, then we claim that the pair a_l, a_1 forms a Critical Pair. This can be seen as follows: Since $l - 1 \geq \frac{|C_1|}{2} - 1$ and $|C_1| = |P| + |P_1|$ it follows that $|P_1| \leq l + 1$. We have

$$d_G(a_l, a_1) = l - 1 \geq \frac{|C_1|}{2} - 1 \geq 2 \ ,$$

since $|C_1| \geq 5$. Furthermore, a_1 owns the edge towards a_2 and we have that $d_G(a_l, a_2) \leq l$ and there is a shortest path from a_l to a_2 which does not use the edge $a_1 a_2$. Agent a_1 has a shortest path of length $|P_1| - 1 \leq l$ to p which does not use the edge $a_l p$ and, since $|P_2| = |P_1|$ agent a_1 also has a shortest path of length at most l to q which does not use the edge $a_l q$.

2. If $|P_2| > |P_1|$, then consider the cycle $C' = P_1 \cup P_2$ and let z be a vertex on C' which has maximum distance to a_l in (G, α). See Fig. 5.10 (right). We claim

that a_l, z forms a Critical Pair in (G, α). Let z' be the neighbor of z on C' to which z owns an edge. Since $|P_2| > |P_1|$, it follows that $z, z' \in V(P_2)$. We have

$$d_G(a_l, z) = k \geq \frac{|P_1| + |P_2|}{2} \geq 2 \ ,$$

which implies $k \geq l$. Moreover, agent a_l has a shortest path to z' of length at most $k+1$ which does not use the edge zz'. The latter is true by choice of z and since z' must be closer to a_1 than z. The path from a_l to z' via P_1 and from a_1 to z' has length at most $k+1$. By choice of z, agent z has a shortest path to q of length at most $k+1$ which does not use the edge $a_l q$, because both q and z lie on the path P_2. Furthermore, agent z has a shortest path to p of length at most $k+1$ and which does not use the edge $a_l p$. This is true, since z's path P' to p via a_1 and then from a_1 along P_1 to p has length

$$|P'| = d_G(z, a_1) + |P_1| - 1 \leq d_G(z, a_1) + l \ ,$$

which shows that this path has at most the length of the path via a_1 and P and edge $a_l p$. Furthermore, by choice of z we have $d_G(z, a_1) \leq k - l + 1$, since otherwise vertex z' has strictly larger distance than z to a_l in G. Thus, $|P'| \leq k+1$. ∎

Lemma 5.6.25 *Let (G, α) be a network which does not contain a Critical Pair and all Min-Cycles in (G, α) have length at least 4. Let C_1 and C_2 be Min-Cycles in (G, α) which overlap in the path P. If $V(P)$ is a separator of G, then G is not in* Sum-*ASE.*

Proof First of all, note that since we assume that (G, α) does not contain a Critical Pair, we have, by Lemma 5.6.22, that $V(C_1) \cap V(C_2) = V(P)$. Moreover, by Lemma 5.6.19, we have that every agent in C_1 owns exactly one edge of C_1 and every agent in C_2 owns exactly one edge of C_2.

Since $V(P)$ is a separator of G we have that the network $G' = G - P$ consists of at least two connected components, that is $G' = G'_1 \cup \cdots \cup G'_l$, for some $l \geq 2$. Let G'_1 be the connected component of G' which contains all vertices $V(C_1) \setminus V(P)$ and let G'_2 be the connected component of G' which contains all vertices $V(C_2) \setminus V(P)$. Let $G'_3, \ldots, G'_l$ be the remaining connected components of G', if there are more than two. Since all vertices of G are either contained in $V(P)$ or in some connected component of G', it follows that

$$|V(G'_1)| + |V(P)| + \sum_{j=3}^{l} |V(G'_j)| \geq \frac{n}{2}$$

or

$$|V(G'_2)| + |V(P)| + \sum_{j=3}^{l} |V(G'_j)| \geq \frac{n}{2} \ .$$

This implies that $|V(G'_1)| \leq \frac{n}{2}$ or $|V(G'_2)| \leq \frac{n}{2}$. W.l.o.g. we will assume that $|V(G'_1)| \leq \frac{n}{2}$ holds.

Let $|V(P)| = k$, for some $k \geq 2$. Lemma 5.6.24 and our assumption that no Critical Pair exists yields that

$$k < \min\left\{\frac{|V(C_1)|}{2} - 1, \frac{|V(C_2)|}{2} - 1\right\} .$$

It follows that

$$|V(C_1) \setminus V(P)| \geq |V(C_1)| - \frac{|V(C_1)|}{2} + 1 = \frac{|V(C_1)|}{2} + 1 \geq k + 3 .$$

Thus, the path $Q = C_1 - P$ has length at least $k + 3$. Since (G, α) has no Critical Pair, we have that the edge-ownership in Q must be directed along the path. Let $V(Q) = \{q_1, \ldots, q_r\}$, where q_i and q_{i+1}, for $1 \leq i \leq r - 1$ are neighbors on the path and let q_1 be the unique agent of Q who does not own an edge of Q. Thus the edge $q_i q_{i+1}$, for $1 \leq i \leq r - 1$ is owned by agent q_{i+1}. We will focus on agent q_2 who owns the edge $q_1 q_2$. Since C_1 is a Min-Cycle and since $|V(Q)| \geq k + 3$ we have that $d_G(q_2, q_r) \geq k + 1$. Let $p \in V(P)$ be the other neighbor of q_1 on Min-Cycle C_1 and let p' be the other neighbor of q_r on C_1.

We claim that agent q_2 can strictly decrease her cost by swapping the edge $q_1 q_2$ towards edge pq_2. Since C_1 is a Min-Cycle, we have that edge $pq_2 \notin E(G)$ so this is indeed a valid edge-swap. By performing this move, agent q_2 decreases her distance to *all* vertices in $V(P)$ by 1. This is true because C_1 is a Min-Cycle, $d_G(q_2, q_r) \geq k + 1$ and $d_G(q_2, p') \geq k + 1$. Since $V(P)$ is a separator of G, this implies that the swap decreases agent q_2's distances to *all* vertices in the set

$$X = V(G'_2) \cup V(P) \cup \bigcup_{j=3}^{l} V(G'_j)$$

by 1. Our observation from above yields that $|X| \geq \frac{n}{2}$. Thus, the swap decreases agent q_2's distance-cost by at least $\frac{n}{2}$. On the other hand, this swap increases agent q_2's distances to some vertices $Y \subset V(G'_1)$ by exactly 1, since the length of q_2's shortest path to vertex q_1 is increased by 1. We have that $|Y| < \frac{n}{2}$, since, by assumption, $|V(G'_1)| \leq \frac{n}{2}$ and $q_2 \notin Y$. Thus, the swap from edge $q_1 q_2$ to edge pq_2 strictly decreases agent q_2's cost, which implies that G cannot be in SUM-Asymmetric Swap Equilibrium. ∎

The so far presented results bring us very close to proving that the Price of Anarchy in the SUM-NCG for $\alpha > 2n - 6$ is constant. Essentially, we reduce the problem to one special case which has strong structural constraints. This can be seen as follows.

By Theorem 5.4.2, any network (G, α) having more than one biconnected component cannot be in Sum-GE. Thus, we may assume that the network (G, α) has exactly one biconnected component C_{bc}.[5]

If C_{bc} is a cycle, that is, if (G, α) is a n-vertex network having exactly n edges, then, by Corollary 5.4.6 and Theorem 3.3.9 it follows that the Price of Anarchy is constant.

Otherwise, that is, if (G, α) is a n-vertex network having at least $n+1$ edges, we can, by Corollary 5.6.10, construct a Min-Cycle cover $\mathcal{C}$ of C_{bc}.

If any Min-Cycle in $\mathcal{C}$ has length at most 4, then, by Lemma 5.6.2, (G, α) cannot be in Sum-GE, which implies that (G, α) cannot be in Sum-NE. Moreover, if any two overlapping Min-Cycles in $\mathcal{C}$ are not well-connected, then, by Lemma 5.6.22, there must be a Critical Pair in (G, α) and with Theorem 5.6.18 it follows that (G, α) cannot be in Sum-NE.

If there are two overlapping Min-Cycles such that the vertices of their overlapping path form a separator of G, then, by Lemma 5.6.25, we know that either G is not in Sum-ASE or that (G, α) contains a Critical Pair. In both cases we know that (G, α) is not in Sum-NE.

The only remaining case is that the vertices of all overlapping paths of two Min-Cycles do not form a separator of G. But then, by Lemma 5.6.13, there must be a cycle of Min-Cycles in (G, α). We know even more: By Lemma 5.6.16, it follows that (G, α) either contains an odd wheel of Min-Cycles or an even wheel of Min-Cycles.

Thus, we have shown that for $\alpha > 2n - 6$ any n-vertex network (G, α) having at least $n+1$ edges either is not in Sum-NE or it must contain an odd wheel of Min-Cycles or an even wheel of Min-Cycles.

The next theorem rules out the case that (G, α) contains an odd wheel of Min-Cycles. Note that by Remark 5.6.15 we have that a wheel of Min-Cycles cannot have a twist. The theorem is even stronger, it also rules out even cycles of Min-Cycles having a twist.

Theorem 5.6.26 *Let (G, α) be a network and let $\mathcal{C}$ be a cycle of Min-Cycles contained in (G, α) and let every Min-Cycle $C \in \mathcal{C}$ have length at least 5. Network (G, α) contains a Critical Pair if $\mathcal{C}$ is an odd cycle of Min-Cycles with no twist or if $\mathcal{C}$ is an even cycle of Min-Cycles with a twist.*

Proof Let $\mathcal{C} = C_1, C_2, \ldots, C_k, C_1$ be a cycle of Min-Cycles contained in (G, α). By Lemma 5.6.19 we can assume that for every Min-Cycle $C_i \in \mathcal{C}$ we have that $\overrightarrow{C_i}$ is a directed cycle, since otherwise we are trivially done. Furthermore, by Lemma 5.6.22, we can assume that all overlapping Min-Cycles are well-connected.

We show that if $\mathcal{C}$ is an odd cycle of Min-Cycles with no twist or if $\mathcal{C}$ is an even cycle of Min-Cycles with a twist, then we get a contradiction to the assumption above. We consider each case separately.

[5] If there is no such component, then we know that (G, α) must be a tree and we already know [FLM+03] that the Price of Anarchy is constant for tree networks.

1. Let $\mathcal{C}$ be an odd cycle of Min-Cycles with no twist. In this case we have that $\mathcal{C}$ is *planar*. Let $\mathcal{C}^e = C_1^e, C_2^e, \dots C_k^e, C_1^e$ be any planar embedding of $\overrightarrow{\mathcal{C}}$ and w.l.o.g. let C_1^e be directed in clockwise order. Since C_1^e and C_2^e share at least one edge, it follows that C_2^e must be directed in counter-clockwise direction. Generally, all directed Min-Cycles having an odd index must be directed clockwise and all directed Min-Cycles having an even index must be directed counter-clockwise. Furthermore, overlapping directed Min-Cycles must have opposite direction. Thus, if $\mathcal{C}^e$ contains an odd number of directed Min-Cycles, then there must be two neighboring directed Min-Cycles, which both have odd index. Thus, both must be directed in clockwise direction, but since they overlap, we get a contradiction.

   It follows that if $|\mathcal{C}|$ is odd, then in $\overrightarrow{\mathcal{C}}$ it is impossible that each agent u has outdegree 1 in every directed Min-Cycle $\overrightarrow{C_j}$ which contains u. Thus, there must be a Min-Cycle $C_j \in \mathcal{C}$ where one agent of C_j owns two edges of C_j and we have a contradiction since $\overrightarrow{C_j}$ is not a directed cycle.

2. Let $\mathcal{C}$ be an even cycle of Min-Cycles with a twist. There must exist an embedding $\mathcal{C}^e = C_1^e, C_2^e, \dots, C_k^e, C_1^e$ of $\overrightarrow{\mathcal{C}}$ in the plane such that exactly two edges cross. Moreover, these two crossing edges must belong to the same cycle in $\mathcal{C}^e$ and w.l.o.g. let this be cycle C_k^e. We can assume that C_1^e is directed in clockwise direction and by an analogous argumentation as above we have that among the cycles $C_1^e, C_2^e, \dots, C_{k-1}^e$ all cycles with odd index are directed clockwise and all cycles with even index are directed counter-clockwise. Since k is even, we have that $k-1$ must be odd. Thus, C_1^e and C_{k-1}^e are both directed clockwise. Let P_k' and P_k'' be the two vertex-disjoint paths, which both consist of vertices from $V(C_k)$ and which connect C_{k-1}^e and C_1^e in $\mathcal{C}^e$. Since two edges from C_k^e cross in $\mathcal{C}^e$, it follows that both paths P_k' and P_k'' contain at least one edge. Let $P_{k-1,k} = C_{k-1}^e \cap C_k^e$ be the shared subgraph of C_{k-1}^e and C_k^e and let $P_{k,1} = C_k^e \cap C_1^e$ be the shared subgraph of C_k^e and C_1^e. By Lemma 5.6.22 and our assumption from above, we have that if $P_{k-1,k}$ or $P_{k,1}$ is not a directed path in $\mathcal{C}^e$, then we have found a Critical Pair. Thus, we can assume that $P_{k-1,k}$ and $P_{k,1}$ both are directed paths in $\mathcal{C}^e$ and it follows that we can decompose the cycle C_k^e as follows: $C_k^e = P_{k-1,k} \cup P_k' \cup P_{k,1} \cup P_k''$ and we can traverse the cycle C_k^e be traversing the paths in this order. Let a be the first vertex in the directed path $P_{k-1,k}$ and let b be its last vertex. Let c be the first vertex in the directed path $P_{k,1}$ and let d be its last vertex. It follows that P_k' either contains the vertices a and c or the vertices b and d. In the first case, if P_k' is directed from a to c, then it follows that agent a owns two edges of the Min-Cycle C_k in G. Otherwise, if P_k' is directed from c to a, then agent c owns two edges of C_k in G. The case where P_k' contains b and d is analogous. In both cases we have identified a Min-Cycle of $\mathcal{C}$ which is not a directed cycle in $\overrightarrow{\mathcal{C}}$ and we have a contradiction. ∎

The above theorem leads us to a strong structural property of all SUM-NE networks for large edge-cost α.

Corollary 5.6.27 *If $\alpha > 2n-6$, then no* Sum-*NE network (G, α) can have an odd cycle of Min-Cycles with no twist or an even cycle of Min-Cycles with a twist.*

Proof Since $\alpha > 2n - 6$, we have by Lemma 5.6.2 that all Min-Cycles in (G, α) must contain at least five vertices. Now, if (G, α) contains an odd cycle of Min-Cycles with no twist or an even cycle of Min-Cycles with a twist, then, by Theorem 5.6.26, it follows that (G, α) contains a Critical Pair. In this case, Theorem 5.6.18 directly yields that (G, α) cannot be in Sum-NE. ∎

As outlined above, the only remaining case is that (G, α) contains an even wheel of Min-Cycles. Which leads us to the following property.

Corollary 5.6.28 *Any non-tree* Sum-*NE network with n vertices and at least $n+1$ edges must contain an even wheel of Min-Cycles if $\alpha > 2n - 6$.*

Finally, we conjecture that this structural property is too strong to be fulfilled in any Sum-NE network.

Conjecture 5.6.29 *If $\alpha > 2n - 6$, then no network in* Sum-*NE can contain an even wheel of Min-Cycles.*

We emphasize that settling this innocent-looking conjecture would imply a constant Price of Anarchy in the Sum-Network Creation Game for $\alpha > 2n-6$. This would be a considerable improvement over the current best bound of $\alpha > 273n$ by Mihalák and Schlegel [MS13] and would significantly reduce the range of α for which no constant bound is known to $[n^{1-\epsilon}, 2n - 6]$, for any fixed $\epsilon \geq \frac{1}{\log n}$.

6. Discussion and Open Problems

In this thesis we have considered Network Creation Games, as introduced by Fabrikant, Luthra, Maneva, Papadimitriou and Shenker [FLM+03], and several versions thereof. Our analysis of these games focused on three different points of view: The approximation perspective, the dynamics perspective and the structure perspective. All those perspectives yield very different problems that have to be tackled and we have coped with them using very different techniques. But interestingly, each perspective also has led to insights for other perspectives.

The Approximation Perspective The study of the approximation of equilibrium states in the Network Creation Games arose from the fact that computing a best response in both the Sum- and the Max-version is computationally hard. If we want to understand selfish network creation as observed in real networks, then we cannot assume that agents have unlimited computational power. Moreover, selfish agents in real networks will refrain from radical infrastructural changes if these lead to only a tiny improvement in service quality. Besides the exposition to a high risk of service disruption during the change, such radical restructurings are more likely to lead to adaptions by other agents. In our study of networks created by very simple agents, we have incorporated both these aspects, that is, we have restricted agents who can compute their best response in polynomial time and who can only perform smooth strategy-changes. We believe that studying networks created by such simple agents is well-suited to understand (communication-)networks in practice. Our agents employ a simple local search heuristic and it seems realistic to assume that many agents in practice will use such sub-optimal algorithms.

We have shown that at least for the Sum-NCG, these very simple agents create networks which are remarkably stable and efficient, that is, they are very close to pure Nash Equilibria in terms of stability and in terms of social cost. Specifically we have shown that any network in Sum-GE is in 3-approximate Sum-NE. Furthermore, if we focus on tree networks, then we have established the quite surprising result that all tree networks in Sum-GE are in Sum-NE as well. Thus, if the created network is a tree network, then simple agents incur no loss in stability or social cost. Unfortunately, the situations is worse for the Max-NCG. We could show that only for tree networks there is no substantial difference between simple agents and agents having the computational power to solve NP-hard problems. For non-tree networks, the outcomes found by simple agents may be far away from the optimal outcomes, at least in terms of stability. The reason for this dichotomy between the Sum- and the Max-version is the different locality of the agents' distance-cost

function. In the SUM-version, the distance-cost of an agent is determined by several more or less independent parts of the networks. Changing edges in one part of the network often has no or only low influence on the distances to agents in some other part. Thus, simple moves like adding, deleting or swapping one edge can decrease an agent's cost locally without affecting many other distances. In contrast, in the MAX-version the distance-cost of an agent is spread throughout the network. It is possible that many agents are in maximum distance to some agent and that the respective shortest paths do not overlap. Thus, to decrease the distance cost in such a situation it is necessary to change many shortest paths with one operation. Clearly this is exactly what our simple agents are not able to do.

One very interesting open problem regarding the approximation of equilibria is determining the exact approximation ratio in the SUM-NCG. We have shown that any SUM-GE network is in 3-approximate SUM-NE and that there are SUM-GE networks which are not in β-approximate SUM-NE for any $\beta < \frac{3}{2}$. We are confident that the upper bound may be improved. Improving the lower bound seems challenging.

The Dynamics Perspective Nash Equilibria or other stable states are the main actors in (Algorithmic) Game Theory. They represent outcomes of a strategic game where - almost magically - all selfish agents are happy in the sense that they do not want to change their current strategy if all other agents stick to their current strategy as well. But, a solution concept for a strategic game is a purely descriptive notion. There is no built-in guidance on how to find these interesting outcomes. Especially for Network Creation Games it is very hard to construct non-trivial stable networks "by hand". The reason is simple: One has to satisfy the needs of n different selfish agents at the same time. That is, even with central coordination it seems to be very difficult to find equilibria. Clearly, the same task without central coordination seems to be even harder. This naturally raises the question how a collection of selfish agents can find such desirable stable states. Studying social networks or the evolution of the Internet suggests that stable networks may be found by repeated and sequential interaction among the agents. Starting from any initial network, agents may sequentially adapt their strategies until this process eventually stabilizes and an equilibrium network emerges.

We have studied such a stabilization process, or game dynamics, for several variants of the Network Creation Game. In our dynamics, agents move sequentially and every move will decrease the moving agent's cost. Our results imply that besides the class of (Asymmetric) Swap Games having trees as initial networks, there is no hope for a convergence guarantee for such dynamics in Network Creation Games. On the other hand, if stabilization occurs, then this happens in a surprisingly low number of agents' moves. This observation holds for the cases where convergence is guaranteed and for all the billions of runs we have simulated in our empirical study. Especially the strongly positive results of the latter indicate why and how

stable networks may be actually found in practice. Thus, such repeated adaptions of selfish agents may be a suitable method for creating stable and efficient overlay networks of for creating ad-hoc networks among mobile network nodes.

In all our negative theoretical results, that is, in all our best response cycle constructions, we have that there is one step in the cycle, where the moving agent can decrease her cost only by a tiny amount. If this agent performs the corresponding strategy-change then the cycle keeps on going. This observation raises the question, whether convergence towards c-approximate equilibria, for some constant c, is always guaranteed. Another question is, if this observation can be turned into a stabilization mechanism as follows: Agents perform improving moves but if they find themselves repeatedly in the same situation, then they gradually become more and more tolerant of sub-optimal outcomes. Thus, agents gradually relax their desire to minimize their local cost to achieve global stability. This reminds of the well-known simulated annealing heuristic in optimization.

The Structure Perspective Most of the works which have analyzed Network Creation Games essentially focused on studying the structure of (swap)-stable networks. Besides some almost trivial stable networks like trees or for some α even cliques, there is a surprisingly broad spectrum of other (swap)-stable networks having all sorts of properties. Clearly, the exact structure of equilibrium networks depends heavily on the edge-cost parameter α and this is the reason why most previous work specified on proving structural results for certain ranges of α. Interestingly, the study of (Asymmetric) Swap Games tries to avoid this parameter dependence and properties of networks in Asymmetric Swap Equilibrium carry over to the other considered solution concepts. The downside of this approach is that the Asymmetric Swap Equilibrium is a very weak solution concept and much stronger structural properties are to be expected for networks in Nash Equilibrium or in Greedy Equilibrium. (Asymmetric) Swap Games remove one of the most interesting features of Network Creation Games: the possibility to create or delete links.

In this thesis we have contributed new structural insights for all considered solution concepts. We have strengthened some of the properties known for networks in Asymmetric Swap Equilibrium to networks in Greedy Equilibrium and we especially focus on non-tree networks in Greedy or Nash Equilibrium for relatively high edge-cost α.

It is conjectured by Alon et al. [ADHL13] that networks in Sum-Swap Equilibrium have polylogarithmic diameter. Mihalák and Schlegel [MS12] conjecture that this diameter is constant and that the diameter of n-vertex networks in Sum-Asymmetric Swap Equilibrium is in $\mathcal{O}(\log n)$. We believe that both conjectures by Mihalák and Schlegel are correct. Moreover we conjecture that the diameter of non-tree networks in Sum-Greedy Equilibrium is constant. Settling this question is one of the most interesting and challenging open problems for Network Creation Games. Since it is known that the Price of Anarchy for tree networks in Sum-Nash

Equilibrium is constant and since tree networks in SUM-GE are in SUM-NE and since SUM-NE is a subset of SUM-GE, proving a constant diameter for non-tree networks in SUM-GE would imply a constant Price of Anarchy in the SUM-NCG and thus answer the decade old question which essentially started the study of Network Creation Games. We believe that an important step towards answering this question is to find the value of α where all stable networks in the SUM-version are guaranteed to be trees. If our Conjecture 5.6.29 is true, then this would imply that for $\alpha > 2n - 6$ any n-vertex network in Nash Equilibrium must have $n - 1$ or n edges, that is, these networks are trees or they contain exactly one cycle. We believe that an even stronger statement may be true: For $\alpha \geq n$ all networks in SUM-Greedy Equilibrium are trees. This would be a tight result, since there are non-tree networks in SUM-GE for $n - \epsilon < \alpha < n$, for any $\epsilon > 0$.

Further Research Directions So far, Network Creation Games capture the agents' desire to have good connection quality at low cost. The outcomes of these games are often tree networks or very sparse non-tree networks. Thus, they provide the necessary structure to maintain short communication paths but almost no redundant paths. This implies that such networks may be highly vulnerable to edge or node failure. We believe that incorporating a certain desire for robustness in the agents' cost function is highly realistic and should lead to networks which are much closer to real networks built by selfish agents.

There are several ways to incorporate robustness. On way is to restrict the degrees of the vertices in the network. This leads to better behavior under single node-failure since there are no communication-critical central nodes as in star-like topologies. Another way would be to enforce that the network is k-connected by penalizing agents who could be cut off from the network by removing at most k edges. Still another way is to introduce an adversary who removes edges at random and the agents' cost is the expected number of agents to which they become disconnected. This approach was proposed by Kliemann [Kli11, Kli13] but with a restricted adversary who is allowed to remove only one edge.

Another interesting research direction is the removal of the implicit assumption that agents know the complete network topology. Especially for large networks, it seems unrealistic that any agent has a global view of the current network structure. Thus, agents could be restricted to act within some local neighborhood and thereby implicitly influencing the global structure. Distances to all agents could still be used in the cost function but agents now no longer know their exact shortest paths to any other node, but only some prefix of these paths.

Last but not least, there is an obvious extension of the NCG which could be investigated. Distances in the NCG are measured by hop-count but what happens if edges have lengths? That is, edges may still be bought, but there is a host graph which specifies the length of any edge. So far, the works [DHMZ09, BGLP12] assume that edges in the host graph only have length 1 or length ∞.

Bibliography

[ADHL13] N. Alon, E. Demaine, M. Hajiaghayi, and T. Leighton. Basic network creation games. *SIAM Journal on Discrete Mathematics*, 27(2):656–668, 2013.

[ADK+08] Elliot Anshelevich, Anirban Dasgupta, Jon Kleinberg, Eva Tardos, Tom Wexler, and Tim Roughgarden. The price of stability for network design with fair cost allocation. *SIAM Journal on Computing*, 38(4):1602–1623, 2008.

[AEED+06] Susanne Albers, Stefan Eilts, Eyal Even-Dar, Yishay Mansour, and Liam Roditty. On nash equilibria for a network creation game. In *Proceedings of the seventeenth annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '06, pages 89–98, New York, NY, USA, 2006. ACM.

[AFM09] Nir Andelman, Michal Feldman, and Yishay Mansour. Strong price of anarchy. *Games and Economic Behavior*, 65(2):289–317, 2009.

[AGK+04] Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing*, 33(3):544–562, 2004.

[AL10] Susanne Albers and Pascal Lenzner. On approximate nash equilibria in network design. In Amin Saberi, editor, *Internet and Network Economics*, volume 6484 of *Lecture Notes in Computer Science*, pages 14–25. Springer Berlin Heidelberg, 2010.

[Alb03] Susanne Albers. Online algorithms: a survey. *Mathematical Programming*, 97(1-2):3–26, 2003.

[AS12] Krzysztof Apt and Sunil Simon. A classification of weakly acyclic games. In Maria Serna, editor, *Algorithmic Game Theory*, volume 7615 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin / Heidelberg, 2012.

[BG00] Venkatesh Bala and Sanjeev Goyal. A noncooperative model of network formation. *Econometrica*, 68(5):1181–1229, 2000.

[BGLP12] Davide Bilò, Luciano Gualà, Stefano Leucci, and Guido Proietti. The max-distance network creation game on general host graphs. In PaulW. Goldberg, editor, *Internet and Network Economics*, volume 7695 of *Lecture Notes in Computer Science*, pages 392–405. Springer Berlin Heidelberg, 2012.

[BHN08] Ulrik Brandes, Martin Hoefer, and Bobo Nick. Network creation games with disconnected equilibria. In Christos Papadimitriou and Shuzhong Zhang, editors, *Internet and Network Economics*, volume 5385 of *Lecture Notes in Computer Science*, pages 394–401. Springer Berlin Heidelberg, 2008.

[BK11] Michael Brautbar and Michael Kearns. A clustering coefficient network formation game. In Giuseppe Persiano, editor, *Algorithmic Game Theory*, volume 6982 of *Lecture Notes in Computer Science*, pages 224–235. Springer Berlin Heidelberg, 2011.

[BS08] Nadine Baumann and Sebastian Stiller. The price of anarchy of a network creation game with exponential payoff. In Burkhard Monien and Ulf-Peter Schroeder, editors, *Algorithmic Game Theory*, volume 4997 of *Lecture Notes in Computer Science*, pages 218–229. Springer Berlin Heidelberg, 2008.

[CLHKS12] Andreas Cord-Landwehr, Martina Hüllmann, Peter Kling, and Alexander Setzer. Basic network creation games with communication interests. In Maria Serna, editor, *Algorithmic Game Theory*, volume 7615 of *Lecture Notes in Computer Science*, pages 72–83. Springer Berlin / Heidelberg, 2012.

[CP05] Jacomo Corbo and David Parkes. The price of selfish behavior in bilateral network formation. In *Proceedings of the twenty-fourth annual ACM symposium on Principles of distributed computing*, PODC '05, pages 99–107, New York, NY, USA, 2005. ACM.

[DHMZ09] Erik D. Demaine, Mohammad Taghi Hajiaghayi, Hamid Mahini, and Morteza Zadimoghaddam. The price of anarchy in cooperative network creation games. *SIGecom Exchanges*, 8(2):2:1–2:20, December 2009.

[DHMZ12] Erik D. Demaine, Mohammad Taghi Hajiaghayi, Hamid Mahini, and Morteza Zadimoghaddam. The price of anarchy in network creation games. *ACM Transactions on Algorithms*, 8(2):13, 2012.

[Die10] Reinhard Diestel. *Graph Theory*, volume 173 of *Graduate Texts in Mathematics*. Springer-Verlag, Heidelberg, fourth edition, 2010.

[EFM+11] Shayan Ehsani, MohammadAmin Fazli, Abbas Mehrabian, Sina Sadeghian Sadeghabad, MohammadAli Safari, Morteza Saghafian, and Saber ShokatFadaee. On a bounded budget network creation game. In *Proceedings of the 23rd ACM symposium on Parallelism in algorithms and architectures*, SPAA '11, pages 207–214, New York, NY, USA, 2011. ACM.

[Eil08] Stefan Eilts. *Approximative Nash-Gleichgewichte in Netzwerk-Spielen.* PhD thesis, Freiburg (Breisgau), Univ., Diss., 2009, 2008.

[FLM+03] Alex Fabrikant, Ankur Luthra, Elitza Maneva, Christos H. Papadimitriou, and Scott Shenker. On a network creation game. In *Proceedings of the twenty-second annual Symposium on Principles of Distributed Computing*, PODC '03, pages 347–351, New York, NY, USA, 2003. ACM.

[GHLL13] Ronald Graham, Linus Hamilton, Ariel Levavi, and Po-Shen Loh. Anarchy is free in network creation. *CoRR*, abs/1307.3113, 2013.

[GJ79] Michael R Garey and David S Johnson. *Computers and intractability*, volume 174. Freeman New York, 1979.

[GKSB12] András Gulyás, Attila Kõrösi, Dávid Szabó, and Gergely Biczók. On greedy network formation. In *Proceedings of ACM SIGMETRICS/Performance W-PIN*, 2012.

[Gol71] A. J. Goldman. Optimal center location in simple networks. *Transportation Science*, 5(2):212–221, 1971.

[HM07] Yair Halevi and Yishay Mansour. A network creation game with nonuniform interests. In Xiaotie Deng and FanChung Graham, editors, *Internet and Network Economics*, volume 4858 of *Lecture Notes in Computer Science*, pages 287–292. Springer Berlin Heidelberg, 2007.

[Jac03] M. O. Jackson. A survey of models of network formation: Stability and efficiency. *Group Formation in Economics: Networks, Clubs and Coalitions*, 2003.

[Jac10] Matthew O Jackson. *Social and economic networks.* Princeton University Press, 2010.

[JW96] Matthew O Jackson and Asher Wolinsky. A strategic model of social and economic networks. *Journal of economic theory*, 71(1):44–74, 1996.

[KH79a] O. Kariv and S. L. Hakimi. An algorithmic approach to network location problems. i: The p-centers. *SIAM Journal on Applied Mathematics*, 37(3):pp. 513–538, 1979.

[KH79b] O. Kariv and S. L. Hakimi. An algorithmic approach to network location problems. ii: The p-medians. *SIAM Journal on Applied Mathematics*, 37(3):pp. 539–560, 1979.

[KL13] Bernd Kawald and Pascal Lenzner. On dynamics in selfish network creation. In *Proceedings of the 25th ACM symposium on Parallelism in algorithms and architectures*, SPAA '13, pages 83–92, New York, NY, USA, 2013. ACM.

[Kli11] Lasse Kliemann. The price of anarchy for network formation in an adversary model. *Games*, 2(3):302–332, 2011.

[Kli13] Lasse Kliemann. The price of anarchy in bilateral network formation in an adversary model. *arXiv preprint arXiv:1308.1832*, 2013.

[KP99] Elias Koutsoupias and Christos Papadimitriou. Worst-case equilibria. In *Proceedings of the 16th annual Conference on Theoretical Aspects of Computer Science*, STACS'99, pages 404–413, Berlin, Heidelberg, 1999. Springer-Verlag.

[Len11] Pascal Lenzner. On dynamics in basic network creation games. In Giuseppe Persiano, editor, *Algorithmic Game Theory*, volume 6982 of *Lecture Notes in Computer Science*, pages 254–265. Springer Berlin / Heidelberg, 2011.

[Len12] Pascal Lenzner. Greedy selfish network creation. In PaulW. Goldberg, editor, *Internet and Network Economics*, volume 7695 of *Lecture Notes in Computer Science*, pages 142–155. Springer Berlin Heidelberg, 2012.

[Lin03] Henry Lin. On the price of anarchy of a network creation game. Class final project. 2003.

[LRSC01] Charles E Leiserson, Ronald L Rivest, Clifford Stein, and Thomas H Cormen. *Introduction to algorithms*. The MIT press, 2001.

[MS96] Dov Monderer and Lloyd S. Shapley. Potential games. *Games and Economic Behavior*, 14(1):124 – 143, 1996.

[MS12] Matúš Mihalák and Jan Christoph Schlegel. Asymmetric swap-equilibrium: A unifying equilibrium concept for network creation games. In Branislav Rovan, Vladimiro Sassone, and Peter Widmayer, editors, *Mathematical Foundations of Computer Science 2012*, volume 7464 of *LNCS*, pages 693–704. Springer Berlin / Heidelberg, 2012.

[MS13] Matúš Mihalák and Jan Christoph Schlegel. The price of anarchy in network creation games is (mostly) constant. *Theory of Computing Systems*, 53(1):53–72, 2013.

[Mye91] R.B. Myerson. *Game theory: analysis of conflict*. Harvard University Press, 1991.

[Nas50] John F. Nash. Equilibrium points in n-person games. *PNAS*, 36(1):48–49, 1950.

[NPRS13] Sotiris E. Nikoletseas, Panagiota N. Panagopoulou, Christoforos Raptopoulos, and Paul G. Spirakis. On the structure of equilibria in basic network formation. *CoRR*, abs/1306.1677, 2013.

[NRTV07] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, New York, NY, USA, 2007.

[OR94] Martin J Osborne and Ariel Rubinstein. *A Course in Game Theory*. The MIT Press, 1994.

[Pap01] Christos Papadimitriou. Algorithms, games, and the internet. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, STOC '01, pages 749–753, New York, NY, USA, 2001. ACM.

[SLB09] Yoav Shoham and Kevin Leyton-Brown. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2009.

[ST85] Daniel D Sleator and Robert E Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.

[Vaz01] Vijay V. Vazirani. *Approximation algorithms*. Springer, 2001.

[VNM44] John Von Neumann and Oskar Morgenstern. *The theory of games and economic behavior*. Princeton university press, 1944.

[Voo00] Mark Voorneveld. Best-response potential games. *Economics Letters*, 66(3):289 – 295, 2000.

[Wat01] Alison Watts. A dynamic model of network formation. *Games and Economic Behavior*, 34(2):331–341, 2001.

[You93] H. Peyton Young. The evolution of conventions. *Econometrica*, 61(1):pp. 57–84, 1993.

A. Code for Simulations

For all our simulations we used scripts written in Python 2.7.3 with the Python graph package NetworkX version 1.7.[1] We use undirected graphs with edge-labels specifying the owner of the respective egde.

A.1. Python Code for the Empirical Study of the ASG

We give the code for the SUM-ASG employing the max cost policy. The initial network is randomly generated such that it is connected and every agent owns exactly k edges.

Listing A.1: Code for SUM ASG with identical budgets and max cost policy

```
from sys import stdin, stderr, argv, stdout
from collections import deque
import networkx as nx
import random
import math
import pickle

def calc_cost(G,v,alpha):
    s = 0
    dist = 1
    q1 = deque()
    q1.append(v)
    q2 = deque()
    unmarked = set(G.nodes())
    while q1 or q2:
        if q1:
            while q1:
                u = q1.pop()
                if u in unmarked:
                    unmarked.remove(u)
                for w in G.neighbors_iter(u):
                    if w in unmarked:
                        q2.append(w)
                        unmarked.remove(w)
                        s = s+dist
            if unmarked:
                dist = dist+1
        if q2:
            while q2:
                u = q2.pop()
                if u in unmarked:
                    unmarked.remove(u)
```

[1] The NetworkX package is available at `http://networkx.github.io/`.

```
33                  for w in G.neighbors_iter(u):
34                      if w in unmarked:
35                          q1.append(w)
36                          unmarked.remove(w)
37                          s = s+dist
38              if unmarked:
39                  dist = dist+1
40      if unmarked:         #if network disconnected
41          return 99999999
42      else:
43          return s
44
45  def check_best_swap(G,v,alpha,costbefore):
46      decrease = 0
47      bad_neighbor = -1
48      new_neighbor = -1
49      change_list = []
50      N = set(G.neighbors(v))
51      Q = set()  #set of neighbors to which v owns an edge
52      for u in N:
53          if G[v][u]['owner'] == v:
54              Q.add(u)
55      R = set(G.nodes()) - N
56      R.remove(v) #set of non-neighbors of v
57      for u in Q:
58          for w in R:
59              G.remove_edge(v,u)
60              G.add_edge(v,w,owner=v)
61              newcost = calc_cost(G,v,alpha)
62              G.remove_edge(v,w)
63              G.add_edge(v,u,owner=v)
64              costdiff = costbefore - newcost
65              if costdiff > 0:
66                  if costdiff > decrease:
67                      decrease = costdiff
68                      change_list = []
69                      change = (u,w)
70                      change_list.append(change)
71                  elif costdiff == decrease:
72                      change = (u,w)
73                      change_list.append(change)
74      if change_list:
75          changepair = random.sample(change_list,1)[0]
76          bad_neighbor = changepair[0]
77          new_neighbor = changepair[1]
78      return decrease,bad_neighbor,new_neighbor
79
80  def count_own_edges(G,v):
81      count = 0
82      for u in G.neighbors_iter(v):
83          if G[v][u]['owner'] == v:
84              count = count + 1
85      return count
86
87  def perform_local_search_step(G,v,alpha):
88      changed = False
89      H = G.copy()
90      costbefore = calc_cost(H,v,alpha)
91      own_edges = count_own_edges(H,v)
92      if own_edges > 0:
93          check_swap = check_best_swap(H,v,alpha,costbefore)
94      else:
95          check_swap = [0]
96      decs = check_swap[0]
97      maxdec = decs
```

```
98      if maxdec == 0:
99          improving = False
100     else:
101         changed = True
102         H.remove_edge(v,check_swap[1])
103         H.add_edge(v,check_swap[2],owner=v)
104     if not changed:
105         return True,G
106     else:
107         return False,H
108
109 def check_costs(G,alpha):
110     cost_list = []
111     for v in G.nodes_iter():
112         cost_v = calc_cost(G,v,alpha)
113         cost_list.append((v,cost_v))
114     return sorted(cost_list, key=lambda entry:entry[1])
115
116 def randomconverge(G,alpha):
117     random.seed()
118     stable = False
119     steps = 0
120     diam_start = nx.diameter(G)
121     while not stable:
122         node_list = check_costs(G,alpha)
123         stable = True
124         while node_list:
125             i = node_list.pop()[0]
126             test = perform_local_search_step(G,i,alpha)
127             if not test[0]:
128                 stable = False
129                 new_graph = test[1]
130                 break
131         if stable:
132             diam = nx.diameter(G)
133             return G,diam_start,diam,steps
134         else:
135             steps = steps + 1
136             G = new_graph
137
138 def graphgen(n,m,k):
139 #generates random connected graph where every node owns k edges
140         random.seed()
141         done = False
142         while not done:
143             done = True
144             G = nx.empty_graph(n)
145             connected = set()
146             remaining = set(range(0,n))
147             ownerset = set(range(0,n))
148             ownerlist = []
149             for i in range(0,n):
150                 ownerlist.append(k)
151             start = random.randint(0,n-1)
152             connected.add(start)
153             remaining.remove(start)
154             for i in range(0,n-1): #generates random spanning tree
155                 u_list = random.sample(remaining,1)
156                 u = u_list[0]
157                 v_list = random.sample(connected,1)
158                 v = v_list[0]
159                 remaining.remove(u)
160                 connected.add(u)
161                 owners = [u,v]
162                 if ownerlist[v] == 0:   #if v has enough edges
```

```
163                     randowner = u
164                 else:
165                     randowner = random.sample(owners,1)[0]
166                 if randowner in ownerset:
167                     ownerlist[randowner] = ownerlist[randowner]-1
168                     if ownerlist[randowner] == 0:
169                         ownerset.remove(randowner)
170                 G.add_edge(u,v,owner=randowner)
171             edgenum = n-1
172             if m > n-1:
173                 while edgenum < m:
174                     owner = False
175                     if ownerset:
176                         owner = True
177                         x = random.choice(list(ownerset))
178                         if G.degree(x) == n-1:
179                             done = False
180                             break
181                         ownerlist[x] = ownerlist[x]-1
182                         if ownerlist[x] == 0:
183                             ownerset.remove(x)
184                     else:
185                         x_done = False
186                         x_candidates = set(range(0,n))
187                         while not x_done:
188                             x = random.choice(list(x_candidates))
189                             if G.degree(x) == n-1:
190                                 x_candidates.remove(x)
191                             else:
192                                 x_done = True
193                     checkedge = False
194                     y_candidates = set(range(0,n))
195                     y_candidates.remove(x)
196                     while (not checkedge):
197                         checkedge = False
198                         y = random.choice(list(y_candidates))
199                         if (not G.has_edge(x,y)):
200                             checkedge = True
201                         else:
202                             y_candidates.remove(y)
203                     if owner:
204                         G.add_edge(x,y,owner=x)
205                     else:
206                         owners = [x,y]
207                         randowner = random.sample(owners,1)[0]
208                         G.add_edge(x,y,owner=randowner)
209                     edgenum = edgenum + 1
210         return G
211
212 def main():
213     random.seed()
214     range_min = int(argv[1])
215     range_max = int(argv[2])
216     minedge = int(argv[3])
217     table = []
218     a = 0.0
219     for i in range(0,range_max+1):
220         table.append([0,0])
221     tests = 10000
222     for j in range(1,1001):
223         for nodes in range(range_min,range_max+1,5):
224             maxsteps = table[nodes][0]
225             edges = minedge * nodes
226             count = 0
227             stepsum = table[nodes][1]
```

```
228             while count < 10:
229                 count = count + 1
230                 gr = graphgen(nodes,edges,minedge)
231                 conv = randomconverge(gr,a)
232                 stepsum = stepsum + conv[3]
233                 if conv[3] > maxsteps:
234                     maxsteps = conv[3]
235                     table[nodes][0] = maxsteps
236             table[nodes][1] = stepsum
237         plotpointsmax = []
238         for i in range(range_min,range_max+1,5):
239             plotpointsmax.append([i,table[i][0]])
240         name = 'plotlist5step_id_mc_'+str(minedge)+'__'+str(range_min)
241                     +'_'+str(range_max)+'_'+str(tests)+'_max.txt'
242         out_file = open(name,"w")
243         pickle.dump(plotpointsmax,out_file)
244         out_file.close()
245         plotpointsavg = []
246         for i in range(range_min,range_max+1,5):
247             plotpointsavg.append([i,table[i][1]/(j*count)])
248         name = 'plotlist5step_id_mc_'+str(minedge)+'__'+str(range_min)
249                     +'_'+str(range_max)+'_'+str(tests)+'_avg.txt'
250         out_file = open(name,"w")
251         pickle.dump(plotpointsavg,out_file)
252         out_file.close()
253
254 main()
```

The MAX-version is simulated simply be returning the value dist in the function calc_cost(G,v,alpha). For employing the random move policy, we use the following modification of the function randomconverge(G,alpha).

Listing A.2: Modification for random cost policy

```
1  def randomconverge(G,alpha):
2      random.seed()
3      stable = False
4      steps = 0
5      diam_start = nx.diameter(G)
6      while not stable:
7          stable = True
8          nodeset = set(G.nodes())
9          while nodeset:
10             i = random.sample(nodeset,1)[0]
11             test = perform_local_search_step(G,i,alpha)
12             if not test[0]:
13                 stable = False
14                 new_graph = test[1]
15                 nodeset = set()
16             else:
17                 nodeset.remove(i)
18         if stable:
19             diam = nx.diameter(G)
20             return G,diam_start,diam,steps
21         else:
22             steps = steps + 1
23             G = new_graph
```

A.2. Python Code for the Empirical Study of the GBG

We give the code for the SUM-GBG employing the max cost policy. The initial network is randomly generated such that it is connected and has exactly m edges.

Listing A.3: Code for SUM GBG with max cost policy

```
from sys import stdin, stderr, argv, stdout
from collections import deque
import networkx as nx
import random
import math
import pickle

def calc_cost(G,v,alpha):
    s = 0
    dist = 1
    q1 = deque()
    q1.append(v)
    q2 = deque()
    unmarked = set(G.nodes())
    while q1 or q2:
        if q1:
            while q1:
                u = q1.pop()
                if u in unmarked:
                    unmarked.remove(u)
                for w in G.neighbors_iter(u):
                    if w in unmarked:
                        q2.append(w)
                        unmarked.remove(w)
                        s = s+dist
            if unmarked:
                dist = dist+1
        if q2:
            while q2:
                u = q2.pop()
                if u in unmarked:
                    unmarked.remove(u)
                for w in G.neighbors_iter(u):
                    if w in unmarked:
                        q1.append(w)
                        unmarked.remove(w)
                        s = s+dist
            if unmarked:
                dist = dist+1
    if unmarked:        #if network disconnected
        return 99999999
    else:
        ownedges = 0
        for u in G.neighbors_iter(v):
            if G[v][u]['owner'] == v:
                ownedges = ownedges + 1
        return s+(alpha*ownedges)

def check_best_deletion(G,v,alpha,costbefore):
    decrease = 0
    bad_neighbor = -1
    bad_list = []
    for u in G.neighbors_iter(v):
        if G[v][u]['owner'] == v:
            G.remove_edge(v,u)
```

```
56              newcost = calc_cost(G,v,alpha)
57              G.add_edge(v,u,owner=v)
58              costdiff = costbefore - newcost
59              if costdiff > 0:
60                  if costdiff > decrease:
61                      decrease = costdiff
62                      bad_list = []
63                      bad_list.append(u)
64                  elif costdiff == decrease:
65                      bad_list.append(u)
66      if bad_list:
67          bad_neighbor = random.sample(bad_list,1)[0]
68      return decrease,bad_neighbor
69
70  def check_best_swap(G,v,alpha,costbefore):
71      decrease = 0
72      bad_neighbor = -1
73      new_neighbor = -1
74      change_list = []
75      N = set(G.neighbors(v))
76      Q = set()  #set of neighbors to which v owns an edge
77      for u in N:
78          if G[v][u]['owner'] == v:
79              Q.add(u)
80      R = set(G.nodes()) - N
81      R.remove(v) #set of non-neighbors of v
82      for u in Q:
83          for w in R:
84              G.remove_edge(v,u)
85              G.add_edge(v,w,owner=v)
86              newcost = calc_cost(G,v,alpha)
87              G.remove_edge(v,w)
88              G.add_edge(v,u,owner=v)
89              costdiff = costbefore - newcost
90              if costdiff > 0:
91                  if costdiff > decrease:
92                      decrease = costdiff
93                      change_list = []
94                      change = (u,w)
95                      change_list.append(change)
96                  elif costdiff == decrease:
97                      change = (u,w)
98                      change_list.append(change)
99      if change_list:
100         changepair = random.sample(change_list,1)[0]
101         bad_neighbor = changepair[0]
102         new_neighbor = changepair[1]
103     return decrease,bad_neighbor,new_neighbor
104
105 def check_best_addition(G,v,alpha,costbefore):
106     decrease = 0
107     new_neighbor = -1
108     buy_list = []
109     N = set(G.neighbors(v))
110     R = set(G.nodes()) - N
111     R.remove(v) #set of non-neighbors of v
112     for w in R:
113         G.add_edge(v,w,owner=v)
114         newcost = calc_cost(G,v,alpha)
115         G.remove_edge(v,w)
116         costdiff = costbefore - newcost
117         if costdiff > 0:
118             if costdiff > decrease:
119                 decrease = costdiff
120                 buy_list = []
```

```
121                 buy_list.append(w)
122             elif costdiff == decrease:
123                 buy_list.append(w)
124     if buy_list:
125         new_neighbor = random.sample(buy_list,1)[0]
126     return decrease,new_neighbor
127
128 def count_own_edges(G,v):
129     count = 0
130     for u in G.neighbors_iter(v):
131         if G[v][u]['owner'] == v:
132             count = count + 1
133     return count
134
135 def perform_local_search_step(G,v,alpha):
136     changed = False
137     H = G.copy()
138     costbefore = calc_cost(H,v,alpha)
139     own_edges = count_own_edges(H,v)
140     if own_edges > 0:
141         check_del = check_best_deletion(H,v,alpha,costbefore)
142         check_swap = check_best_swap(H,v,alpha,costbefore)
143     else:
144         check_del = [0]
145         check_swap = [0]
146     check_add = check_best_addition(H,v,alpha,costbefore)
147     decs = [check_del[0],check_swap[0],check_add[0]]
148     maxdec = max(decs)
149     if maxdec == 0:
150         improving = False
151     else:
152         changed = True
153         if check_del[0] == maxdec: #deletion is preferred over swap and add
154             H.remove_edge(v,check_del[1])
155         elif check_swap[0] == maxdec:  #swap is preferred over add
156             H.remove_edge(v,check_swap[1])
157             H.add_edge(v,check_swap[2],owner=v)
158         else:
159             H.add_edge(v,check_add[1],owner=v)
160     if not changed:
161         return True,G
162     else:
163         return False,H
164
165 def check_costs(G,alpha):
166     cost_list = []
167     for v in G.nodes_iter():
168         cost_v = calc_cost(G,v,alpha)
169         cost_list.append((v,cost_v))
170     return sorted(cost_list, key=lambda entry:entry[1])
171
172 def randomconverge(G,alpha):
173     random.seed()
174     stable = False
175     steps = 0
176     diam_start = nx.diameter(G)
177     while not stable:
178         node_list = check_costs(G,alpha)
179         stable = True
180         while node_list:
181             i = node_list.pop()[0]
182             test = perform_local_search_step(G,i,alpha)
183             if not test[0]:
184                 stable = False
185                 new_graph = test[1]
```

```
186                 break
187         if stable:
188             diam = nx.diameter(G)
189             return G,diam_start,diam,steps
190         else:
191             steps = steps + 1
192             G = new_graph
193 
194 def graphgen(n,m): #generates random connected graph with m edges
195         random.seed()
196         G = nx.empty_graph(n)
197         connected = set()
198         remaining = set(range(0,n))
199         start = random.randint(0,n-1)
200         connected.add(start)
201         remaining.remove(start)
202         for i in range(0,n-1):
203             u_list = random.sample(remaining,1)
204             u = u_list[0]
205             v_list = random.sample(connected,1)
206             v = v_list[0]
207             remaining.remove(u)
208             connected.add(u)
209             owners = [u,v]
210             randowner = random.sample(owners,1)[0]
211             G.add_edge(u,v,owner=randowner)
212         for j in range(0,m-n+1):
213             checknodes = False
214             checkedge = False
215             while (not checkedge) and (not checknodes):
216                 checknodes = False
217                 checkedge = False
218                 x = random.randint(0,n-1)
219                 y = random.randint(0,n-1)
220                 if x!=y:
221                     checknodes = True
222                 if not G.has_edge(x,y) and checknodes:
223                     owners = [x,y]
224                     randowner = random.sample(owners,1)[0]
225                     G.add_edge(x,y,owner=randowner)
226                     checkedge = True
227         return G
228 
229 def main():
230     random.seed()
231     range_min = int(argv[1])
232     range_max = int(argv[2])
233     minedge = int(argv[3])
234     alphafactor = float(argv[4])
235     table = []
236     for i in range(0,range_max+1):
237         table.append([0,0])
238     tests = 5000
239     for j in range(1,501):
240         for nodes in range(range_min,range_max+1,10):
241             maxsteps = table[nodes][0]
242             edges = minedge * nodes
243             count = 0
244             stepsum = table[nodes][1]
245             a = alphafactor * nodes
246             while count < 10:
247                 count = count + 1
248                 gr = graphgen(nodes,edges)
249                 conv = randomconverge(gr,a)
250                 m = conv[0].number_of_edges()
```

```
251                     stepsum = stepsum + conv[3]
252                     if conv[3] > maxsteps:
253                         maxsteps = conv[3]
254                         table[nodes][0] = maxsteps
255                 table[nodes][1] = stepsum
256             plotpointsmax = []
257             for i in range(range_min,range_max+1,10):
258                 plotpointsmax.append([i,table[i][0]])
259             name = 'plotlist10step_mc_'+str(minedge)+'_'+str(alphafactor)
260                +'__'+str(range_min)+'_'+str(range_max)+'_'+str(tests)+'_max.txt'
261             out_file = open(name,"w")
262             pickle.dump(plotpointsmax,out_file)
263             out_file.close()
264             plotpointsavg = []
265             for i in range(range_min,range_max+1,10):
266                 plotpointsavg.append([i,table[i][1]/(j*count)])
267             name = 'plotlist10step_mc_'+str(minedge)+'_'+str(alphafactor)
268                +'__'+str(range_min)+'_'+str(range_max)+'_'+str(tests)+'_avg.txt'
269             out_file = open(name,"w")
270             pickle.dump(plotpointsavg,out_file)
271             out_file.close()
272
273 main()
```

The MAX-version is simulated by returning dist+(alpha∗ownedges) in the function calc_costs(G,v,alpha). Moreover the random policy is simulated with the same modification of the function randomconverge(G,alpha) as in the simulation of the ASG.

Erklärung

Hiermit erkläre ich,

- dass ich die vorliegende Dissertationsschrift **On Selfish Network Creation** selbstständig und ohne unerlaubte Hilfe angefertigt habe,
- ich mich nicht bereits anderwärts um einen Doktorgrad beworben habe oder einen solchen besitze, und
- mir die Promotionsordnung der Mathematisch-Naturwissenschaftlichen Fakultät II der Humboldt-Universität zu Berlin bekannt ist, gemäß amtlichem Mitteilungsblatt Nr. 34/2006.

Berlin, den ....