An Approximation Framework for Sequencing Problems with Bipartite Structure

Aleksandar Shurbevski

An Approximation Framework for Sequencing Problems with Bipartite Structure

Aleksandar Shurbevski

Department of Applied Mathematics and Physics
Graduate School of Informatics
Kyoto University
Kyoto, Japan

September, 2014

Doctoral dissertation
submitted to the
Graduate School of Informatics, Kyoto University
in partial fulfillment of
the requirement for the degree of
DOCTOR OF ENGINEERING
(Applied Mathematics and Physics)

Preface

Optimization is a central aim that enterprises and individuals undertake to increase their efficiency, productivity, and ultimately profit. For this reason, problems arising from the strive for ever better performance and the need for optimization have been extensively studied in many areas, and continue to provide worthwhile research topics. There is an abundance of work addressing optimization problems from a variety of research communities, ranging from engineering and operations research, to computer science, applied mathematics, and even finance.

In particular, optimization challenges arising from routing and scheduling problems are highly intriguing, as they have a significant practical impact on saving resources, whether it would be manpower, time, or the ever increasingly important environmental impact. Due to the exhaustive work coming from many sources, we presently have at our disposal a great number of frameworks and tools to model and solve practical optimization problems. As great as the number of existing modeling and solution methodologies might be however, there is still a lot to be achieved by building and extending upon known results, whether we aim for a more general application which is suitable for a variety of environments and scenarios, or for a more specialized application, suited and highly tuned for a particular problem.

Driven by this objective, we approach a problem inspired by an actual application of a robotic arm in a production process. The robotic arm is used as a material handling device, charged with the task of replacing a given number of objects. It is immediately evident that such a scenario occurs in a variety of different settings where material handling devices are employed, and it is of particular interest if those devices are automated or autonomous. Aside from the obvious material handling scenarios, it will be of great interest to explore the applicability of the approaches we have taken to other problems and scenarios. We first build a specialized model for our problem, thereby arriving to a solution which captures the essence of our problem, instead of relying on more general frameworks which can hinder our insight of the truly important features. Following, gradually throughout the thesis we will introduce generalizations to our model, finally arriving to a unique solution to an existing problem. In that manner, we will savor both approaches of building upon existing knowledge men-

tioned earlier, developing a specialized solution, and generalizing a model for greater applicability. Moreover, we will introduce a unique parameter into our problem models, termed *bias factor*, thus narrowing a gap between symmetric and a much more general, but also more difficult, asymmetric versions of known problems.

The main tools we have used in the quest for a solution to our problem come from the vast field of *Combinatorial optimization*. However, as most interesting problems arising in combinatorial optimization, we will show that the problem we treat belongs to the class of *NP-complete* problems. While an efficient solution to any NP-complete problem would imply that there exists a solution to every problem of the class NP, despite intensive work on this topic, it is believed that such a solution is unlikely. This hindsight can discourage further efforts towards finding an efficient solution to a given problem, but on the other hand, can serve as an additional challenge and driving force, for it implies broad consequences of any advancement or by the uncovering of a hidden facet.

Even taking into account the high possibility that an efficient method to exactly solve an NP-complete optimization problem might never be uncovered, still the need for optimization will always persist. To this effect, we set out to build an *approximate* solution method, one that might not necessarily derive an optimal solution, but will provide us with a solution which is provably within a certain deviation bound from an optimal solution.

And this is exactly where we choose to close a circle of continuous improvement, stressing the importance of the topic undertaken in this thesis. As the need for optimization stated in the opening persists, and with the diminishing probability of ever devising efficient solution methods for practical optimization problems, improving the approximation bounds of approximate solution methods is just as important.

We believe that the work described in this thesis provides new insights into how theoretical frameworks can be efficiently employed in optimization problems arising from practical applications. It is only our hope that the work done can serve as a basis for future advancement in related topics.

Kyoto, September, 2014
Aleksandar Shurbevski

Acknowledgement

This thesis would not have been possible without the help of many others.

First and foremost, I must express my heartfelt gratitude to Professor Hiroshi Nagamochi of Kyoto University for his enthusiastic guidance, thorough discussions and persistent encouragement throughout my academic pursuit. He has selflessly provided continuous support and opportunities, yet granted me sufficient freedom and independence, which has allowed me to grow in many aspects. He commented in detail on the whole work in the manuscript, which significantly improved the accuracy of the arguments and the quality of exposition. Without his considerable help, none of this work could have been completed.

I am sincerely grateful to Professor Yoshiyuki Karuno of Kyoto Institute of Technology. Not only did he welcome me as a master student at the Laboratory of Production System Science, but in addition to providing invaluable academic support and guidance, I must observe that he also went largely out of his way to make my transition to a new environment seamless and my continued stay as pleasant as possible. It is certain that I would not have enjoyed much of the freedom and comfort during my graduate studies if it had not been for his generous support.

Due acknowledgement is in order for the members of the Discrete Mathematics Laboratory at Kyoto University. I am indeed grateful for their friendship in times of play and collegial support during times of study. I am especially grateful to Professor Liang Zhao of Kyoto University, for interesting discussions, allowing me to broaden my horizon and deepen my knowledge of recent research trends, as well as the time and patience that he has devoted to introducing me to some of the technical aspects of the work of the Discrete Mathematics Laboratory.

I wish to express my gratitude to the members of the Laboratory for Analysis of System Dependability at NEC, for an engaging and fruitful internship. They have made me feel as part of the team immediately upon my arrival, especially Fumio Machida, who in addition to introducing me to an engaging research topic, spared no time and attention for discussions on various topics and wide-ranging level of detail.

I am also thankful to Professor Yutaka Takahashi of Kyoto University, as well as Professor Yoshito Ohta of Kyoto University, for serving on my dissertation committee. They have provided many comments on earlier drafts of this thesis,

which improved the exposition style, structure and contents.

I owe large gratitude to my family, who have supported me in any endeavor in my life, this one being no exception. Every personal accomplishment is nothing but embodiment of the love and affection they have poured in me, and the values they have guided me to build. Albeit far away, I have constantly felt their presence both figuratively, in every action I take and decision I make, and often quite literary, by virtue of modern communications technology.

Last but not least, I must thank my life partner, Harifara Rabemanolontsoa, for riding out quite a few rough stretches alongside with me on this often times tumultuous journey. I am certain that the future will bring yet many challenges, and I can only hope to face them with mutual support.

Contents

List of Figures

List of Algorithms

1 Introduction

In this chapter, we describe the motivation and inspiration behind the subject treated in this thesis. The portrayal is mainly narrative and only serves as an intuitive description and background of the matter we are about to delve into. Without focusing on technical details, we exhibit certain terminology and notation to be used in the remainder of the thesis.

1.1 Using a Robotic Arm in the Production of Printed Circuit Boards

The topic of this thesis is inspired by a routing problem of a grasp-and-delivery robot used at an actual printed circuit board assembly line. The topic of this original task and references to it are kept throughout the text. We often simply use the term robot to refer to a grasp-and-delivery robot. The assembly line can process at most one printed circuit board at a time. A new (unprocessed) printed circuit board is brought into the assembly line along an automatic guided railway, and is released from there by the railway immediately after the processing had finished.

Let $\mathcal{J} = \{J_1, J_2, \ldots, J_m\}$ denote a set of m printed circuit boards to be processed at the assembly line. In practice, processing a printed circuit board involves an automatic manipulator embedding electronic parts in the board from above. Similar scenarios of using a robotic arm in production and routing issues arising thereof have been studied before, and further details can be found in the literature, e.g., [9, 10, 13, 14, 21, 32, 60]. But more importantly, we hope that the solution approaches developed in the remainder of the thesis can find their way into various other scenarios including routing problems, e.g., [40, 46, 65], even into the rapidly rising use of unmanned aerial vehicles - UAVs [45].

As a printed circuit board is processed at the assembly line, n identical pins support the printed circuit board from underneath to prevent it from overbending. Each printed circuit board comes with a unique circuit pattern, and a dedicated pin configuration, C_i, is designed for every printed circuit board J_i. A pin configuration C_i is in fact a set of locations describing where the pins should be placed

so that they do not obstruct the circuit, while providing adequate support during the processing stage.

We assume that as location sets, pin configurations C_i and C_j are disjoint for $i \neq j$, i.e., $C_i \cap C_j = \emptyset$. This implies that if after processing printed circuit board J_i, the next printed board to be processed is J_j, then the current pin configuration C_i needs to be changed to C_j during the time interval between releasing J_i and bringing J_j for processing. The assembly line employs a single grasp-and-delivery robot for changing between pin configurations.

The grasp-and-delivery robot can grasp at most one pin at a time by using its end-effector, i.e., it is a material handling device with unit capacity. Although not technically accurate, for the remainder of this thesis we refer to the position of the end-effector of the grasp-and-delivery robot as the robot's position, or location. After grasping a pin, the robot delivers the pin from its current location to some different location. We refer to the task of replacing all the pins from one pin configuration to another as a *transition*.

Each repositioning of the grasp-and-delivery robot incurs a cumulative cost. Tentatively, let us denote by $w(u, v)$ the cost incurred by the grasp-and-delivery robot by repositioning from location u to location v.

Henceforth we assume that the pins always rest on a flat surface within an area S reachable by the grasp-and-delivery robot. The bounded area S represents the domain from which pin locations may be assigned. Hence, we can define each pin configuration C_i, $i = 0, 1, \ldots, m$ as

$$C_i := \left\{ c_j^i \in S : j = 1, 2, \ldots, n \right\}. \tag{1.1.1}$$

We will denote the set of configurations by

$$\mathcal{C} := \{C_i : i = 0, 1, \ldots, m\}.$$

For the remainder of this chapter, we will proceed with an intuitive description of the optimization tasks which we will undertake in the remainder of the thesis and which arise from the use of a robotic arm for performing grasp-and-delivery type of material handling tasks in the production of printed circuit boards. The equivalent mathematical formulations shall be described after introducing essential background and related terminology in Chapter 2: Preliminaries.

1.2 A Single Transition Description

Let us focus on the mechanism of replacing the pins in a single transition. Let C_i be the current pin configuration, and C_j the desired pin configuration. As a

reminder, we assume that as location sets, configurations C_i and C_j are disjoint from each other for $i \neq j$.

At the very beginning of a transition, the grasp-and-delivery robot is placed at some location in C_i. Let us assume that this is location $c_l^i \in C_i$, for some $l \in 1, \dots n$. The grasp-and-delivery robot will proceed by grasping the pin which is currently placed at location c_l^i and deliver it to some location $c_{l'}^j \in C_j$. After the pin has been delivered, the robot will move to another location $c_p^i \in C_i$, grasp the pin at location c_p^i and deliver it to another location $c_{p'}^j \in C_j$, and proceed in this fashion until all locations in C_i have been visited, and all pins have been placed at suitable locations in C_j. At this time, the robot will come to a stop at the last location where a pin had been delivered in C_j.

By construction, $|C_i| = |C_j| = n$, $\forall i, j \in \{0, 1, \dots m\}$, which means that during a single transition each location in $C_i \cup C_j$ is visited exactly once. Moreover, the locations of C_i and C_j are visited in an alternating manner, and it is this insight which becomes one of the central topics of investigation in this thesis.

The structure of the description above can be expressed in the following perspective. We can think of the order in which locations are visited in a transition as a permutation of the respective pin configurations. Therefore, let σ_i and τ_j be permutations on the points of C_i and C_j, respectively. The permutations σ_i and τ_j are associated with the transition from pin configuration C_i to pin configuration C_j. We assumed that the location of C_i where the grasp-and-delivery robot resides just at the beginning of a transition, i.e., $\sigma_i(1)$ is known. Now, the order in which locations are visited by the grasp-and-delivery robot can be described as

$$\sigma_i(1) \to \tau_j(1) \to \sigma_i(2) \to \tau_j(2) \to \cdots \to \sigma_i(n) \to \tau_j(n). \tag{1.2.2}$$

The expression of Eq. (1.2.2) can be thought of as describing a path, which we will denote by $P_{i,j}$, and commonly refer to as a *partial transfer route*. We will extend the notation of the cost function $w(u, v)$ to assign a cost $w(P_{i,j})$. Having in mind that the grasp-and-delivery robot incurs a certain cost $w(u, v)$ with every repositioning from location u to location v, and that this cost is cumulative, we obtain the following

$$w(P_{i,j}) = \sum_{k=1}^{n} w(\sigma_i(k), \tau_j(k)) + \sum_{l=1}^{n-1} w(\tau_j(l), \sigma_i(l+1)). \tag{1.2.3}$$

One of the main objectives with which we approach this scenario is efficiently determining such permutations σ_i and τ_j, with a goal of minimizing the cost $w(P_{i,j})$ incurred by the grasp-and-delivery robot in a transition.

1.3 Planning for an Entire Processing Stage

We have intuitively described the process of performing a single transition between pin configurations. However, as stipulated in the opening of this chapter, in general we are given m printed boards to be processed at an assembly line. Additionally, a fictitious board J_0 is assigned to the pin configuration C_0, describing the locations of pins before the start of the processing stage at the assembly line. Henceforth, we shall refer to the order unprocessed printed circuit boards are brought at the assembly line as a *processing sequence*, or simply, a sequence.

1.3.1 Fixed Processing Sequence

As a first step towards a comprehensive solution to the challenge set forth by the objective of minimizing the cost incurred by the grasp-and-delivery robot, let us assume that the processing sequence of printed circuit boards is fixed. Without loss of generality, let the order in which printed circuit boards are brought in the assembly line for processing be

$$J_0 \to J_1 \to \cdots \to J_m. \tag{1.3.4}$$

We refer to the transition from pin configuration $C_{\ell-1}$ to C_ℓ, $\ell = 1, 2, \ldots, m$ as transition ℓ,

$$C_0 \xrightarrow{\text{Transition 1}} C_1 \xrightarrow{\text{Transition 2}} \cdots \xrightarrow{\text{Transition } m-1} C_{m-1} \xrightarrow{\text{Transition } m} C_m \tag{1.3.5}$$

for m transitions in total during a processing stage. For the complete routing of the grasp-and-delivery robot over a production sequence we adopt the term *transfer route*. A transfer route is a contiguous sequence of relocations of the grasp-and-delivery robot. Let Π denote a transfer route, then for the cost incurred by the grasp-and-delivery robot over the entire transfer route Π we write $L(\Pi)$.

A partial transfer route $P_{\ell-1,\ell}$ for transition ℓ can be described as

$$\sigma_{\ell-1}(1) \to \tau_\ell(1) \to \sigma_{\ell-1}(2) \to \tau_\ell(2) \to \cdots \to \sigma_{\ell-1}(n) \to \tau_\ell(n). \tag{1.3.6}$$

In the previous section, Section 1.2, it was stated that a single transition ℓ starts at location $\sigma_{\ell-1}(1) \in C_{\ell-1}$, and terminates at location $\tau_\ell(n) \in C_\ell$. Additionally, we assume that we are given the information of the grasp-and-delivery robot's location at the beginning of a processing stage, and that it is at a pin's location of C_0. In other words, $\sigma_0(1)$ is known beforehand. This assumption is not prohibitive, for there are exactly n possible candidates for the value of $\sigma_0(1)$, and without loss of generality we can assume that $\sigma_0(1) = c_1^0$ ($c_1^0 \in C_0$, Eq. (1.1.1)). For notational convenience, let $\tau_0(n) := \sigma_0(1)$. Now, given m partial transfer

routes, $P_{\ell-1,\ell}$, $\ell = 1, 2, \ldots, m$, if some partial transfer route $P_{l-1,l}$ is such that $\sigma_{l-1}(1) \neq \tau_{l-1}(n)$, then in order to maintain the contiguity of a transfer route, the grasp-and-delivery robot needs to traverse the distance between $\tau_{l-1}(n)$ and $\sigma_{l-1}(1)$ before proceeding with transferring the pins in the l^{th} transition. We call such traversals from $\tau_{l-1}(n)$ to $\sigma_{l-1}(1)$ *aligning moves*, because they can be thought of as aligning consecutive partial transfer routes. These additional relocations are cumbersome to track and account for. However, under certain reasonable assumptions (namely, that the cost function w is *metric*, see Chapter 2), we can assume that there exists an optimal transfer route Π where individual partial transfer routes satisfy the following condition

$$\sigma_\ell(1) = \tau_\ell(n), \quad \ell = 0, 1, \ldots, m-1. \tag{1.3.7}$$

Hereafter, we refer to Eq. (1.3.7) as the *consecutiveness constraint* (introduced by Karuno et al. [32]). Imposing such a constraint as consecutiveness on transfer routes can be justified by the following proposition.

Proposition 1.1. *If there exists a transfer route* Π *with cost* $L(\Pi)$ *for the grasp-and-delivery robot, such that some of the partial transfer routes do not satisfy Eq. (1.3.7) and aligning moves are employed between those partial transfer routes, then the concerned partial transfer routes can be modified to achieve contiguity without aligning moves, and moreover, without increasing the cost* $L(\Pi)$*.*

We will argue the correctness of Proposition 1.1 after we will have laid out sufficient foundations for mathematical formality in Chapter 2, Lemma 2.2.

In consequence, in the search for an optimal transfer route, we confine our investigation to transfer routes where each partial transfer route satisfies Eq. (1.3.7). Such transfer routes are said to be *feasible*, and we consider a transfer route Π as a concatenation of m individual partial transfer routes (which satisfy the consecutiveness constraint),

$$\Pi = (P_{\ell-1,\ell} : \ell = 1, 2, \ldots, m). \tag{1.3.8}$$

At this point we can define the cost $L(\Pi)$ of a feasible transfer route Π as

$$L(\Pi) = \sum_{\ell=1}^{m} w(P_{\ell-1,\ell}). \tag{1.3.9}$$

Summarizing the issue of devising a comprehensive transfer route for the grasp-and-delivery robot for an entire processing stage, in a nutshell we can define the following problem.

The Fixed-sequence Repetitive Routing Problem

Instance: A sequence $(C_\ell : \ell = 0, 1, \ldots, m)$ of m pin configurations over a surface area S, and a cost function $w(u, v)$ for $u, v \in S$.

Objective: Find a feasible transfer route Π^* such that $L(\Pi^*) \leq L(\Pi)$, for any other transfer route Π for the given instance.

From here onward, we refer to the problem of seeking out a feasible transfer route of the grasp-and-delivery robot as the Repetitive Routing Problem, or RRP for brevity. An instance of the RRP is characterized by the ordered pair $(\mathcal{C}, w)$. The description above focused on a version which we elaborated as the Fixed-sequence RRP. The reader is correct to inquire whether there are different versions of the Repetitive Routing Problem, and indeed, we will address this question in the following section.

1.3.2 Permutable Processing Sequence

So far we have introduced only a limited scope of the challenges provided by the issue of seeking out a feasible transfer route of the grasp-and-delivery robot, i.e., the Repetitive Routing Problem. A natural question arising once we are familiarized with the Fixed-sequence RRP is what would change if we were allowed to choose a processing sequence in addition to a transfer route.

Let ψ be a permutation on the interval $[0, m]$, with the condition that $\psi(0) = 0$. Now, we can index the pin configurations during a processing stage as follows

$$C_{\psi(0)} \xrightarrow{\text{Transition } 1} C_{\psi(1)} \xrightarrow{\text{Transition } 2} \cdots \xrightarrow{\text{Transition } m-1} C_{\psi(m-1)} \xrightarrow{\text{Transition } m} C_{\psi(m)}. \tag{1.3.10}$$

Since a permutation ψ with $\psi(0) = 0$ uniquely determines the order in which printed circuit boards are brought at the assembly line for processing, and the transitions which are to be performed by the grasp-and-delivery robot, we adopt the same reference (in this example ψ) to refer to both a processing sequence and a permutation on the integer interval $[0, m]$.

A transfer route with respect to a production sequence ψ is denoted by Π_ψ. Partial transfer routes with respect to the production sequence ψ are individually denoted by $P^\psi_{i,j}$, $0 \leq i \neq j \leq m$. Similarly, the permutations defining the order in which locations of S are visited in a transition are denoted by σ^ψ_i and τ^ψ_j.

(The necessity of introducing this distinction at the cost of increasing notation complexity will be justified shortly.) Now, for the cost $L(\Pi_\psi)$ we can write

$$L(\Pi_\psi) = \sum_{\ell=1}^{m} w(P^{\psi}_{\psi(\ell-1),\psi(\ell)}), \tag{1.3.11}$$

while the consecutiveness constraint becomes

$$\sigma^{\psi}_{\psi(\ell)}(1) = \tau^{\psi}_{\psi(\ell)}(n), \quad \ell = 1, 2, \ldots, m, \tag{1.3.12}$$

with the initial assumption that $\psi(0) = 0$, and $\sigma^{\psi}_{0}(1) = c^{0}_{1}$. We can now state the complete problem description incorporating the processing sequence.

The Permutable-sequence Repetitive Routing Problem

Instance: A set $\mathcal{C} := \{C_\ell : \ell = 0, 1, \ldots, m\}$ of m pin configurations over a surface area S, and a cost function $w(u, v)$ for $u, v \in S$.

Objective: Find a production sequence $\mathbf{\Psi}$ and feasible transfer route $\Pi^{*}_{\mathbf{\Psi}}$ with respect to $\mathbf{\Psi}$, such that $L(\Pi^{*}_{\mathbf{\Psi}}) \leq L(\Pi_\psi)$, for any other production sequence ψ and transfer route Π_ψ for the given instance.

But, how does introducing a permutation in the processing sequence influence the flavor of the problem? And why is it important to label a partial transfer route $P_{i,j}$ with $P^{\psi}_{i,j}$, signifying that it has been computed with respect to the production sequence ψ? In order to give an answer to these questions in an intuitive manner, let ψ be a given processing sequence, and let $\Pi^{*}_{\psi} = (P^{\psi}_{\psi(\ell-1),\psi(\ell)} : \ell = 1, 2, \ldots, m)$ be an optimal transfer route with respect to ψ, minimizing $L(\Pi^{*}_{\psi})$. Let us examine a different sequence, φ, which differs from ψ in exactly two places. Let those two places be i and j, $0 < i \neq j \leq m$, such that $\psi(i) = \varphi(j)$ and $\psi(j) = \varphi(i)$. Obviously, the permutations $\sigma^{\psi}_{\psi(i)}$, $\tau^{\psi}_{\psi(i)}$, $\sigma^{\psi}_{\psi(j)}$ and $\tau^{\psi}_{\psi(j)}$ cannot be simply substituted for the respective $\sigma^{\varphi}_{\varphi(j)}$, $\tau^{\varphi}_{\varphi(j)}$, $\sigma^{\varphi}_{\varphi(i)}$ and $\tau^{\varphi}_{\varphi(i)}$ to define the partial transfer routes $P^{\varphi}_{\varphi(i-1),\varphi(i)}$ and $P^{\varphi}_{\varphi(j-1),\varphi(j)}$. This is due to the fact that, $C_{\varphi(i)}$ and $C_{\psi(i)}$ (respectively, $C_{\varphi(j)}$ and $C_{\psi(j)}$) are different (and disjoint) location sets on the surface S, and calculating $w(P^{\varphi}_{\varphi(i-1),\varphi(i)})$ (respectively, $w(P^{\varphi}_{\varphi(j-1),\varphi(j)})$) according to Eq. (1.2.3) is almost guaranteed to produce a different result, which in turn contributes also to the value of $L(\Pi_\varphi)$, Eq. (1.3.11). Furthermore, given a partial transfer route $P^{\psi}_{\psi(\ell-1),\psi(\ell)}$ for transition ℓ, the roles of both $\sigma^{\psi}_{\psi(\ell-1)}$ and $\tau^{\psi}_{\psi(\ell)}$ are highly coupled in determining the value of $w(P^{\psi}_{\psi(\ell-1),\psi(\ell)})$, and hence, the value of $L(\Pi^{*}_{\psi})$. This reasoning is general, for any processing sequence. Therefore, a local

change in a single partial transfer route directly influences the preceding and following partial transfer routes as well, and this influence whiplashes through the entire transfer route. Finally, there is also the consecutiveness constraint which should be satisfied in a feasible transfer route.

In conclusion, even a seemingly small change in a processing sequence necessitates computing all partial transfer routes anew. So, even though $\varphi(\ell) = \psi(\ell)$ for every $\ell = 1, 2, \ldots, m$ other than i and j, the already existing $\sigma^{\psi}(\ell)$ and $\tau^{\psi}(\ell)$ cannot be re-used for $\sigma^{\varphi}(\ell)$ and $\tau^{\varphi}(\ell)$ and still result in a feasible transfer route Π^{*}_{φ} with $L(\Pi^{*}_{\varphi}) \leq L(\Pi_{\varphi})$ for any transfer route Π_{φ} with respect to the processing sequence φ.

1.4 Biased Asymmetric Cost

Another extension sprouting from the scenario at a printed circuit board processing assembly line concerns the cost function we have used to evaluate candidate solutions. Up until now we have assigned a certain cost whenever the grasp-and-delivery robot repositions from location u to location v. What comes as a natural question is whether the robot would incur the same cost by repositioning while it is grasping a pin. Does it really take the same effort to change the location of an empty material handling device (or vehicle) and when it has some payload? For brevity, we say that the robot is performing an *empty move* from u to v when the robot is repositioning from location u to location v without grasping a pin, or simply, that u-v is an empty move. Conversely, whenever the robot is repositioning while grasping a pin to be delivered from location u to location v, we say that the robot performs a *transfer move* from u to v, or that u-v is a transfer move. In order to answer the question above and develop a broader model, we introduce a *bias* term $\beta \geq 1$, and a *biased* cost function $\widetilde{w}(u, v)$ as follows

$$\widetilde{w}(u,v) = \begin{cases} w(u,v), & u\text{-}v \text{ is an empty move,} \\ \beta \cdot w(u,v), & u\text{-}v \text{ is a transfer move.} \end{cases} \tag{1.4.13}$$

Returning to the description of a partial transfer route $P_{i,j}$, Eq. (1.2.2), we can discern that the set of transfer moves is

$$\{\sigma_i(k) \to \tau_j(k) : k = 1, 2, \ldots, n\},$$

while the set of empty moves is

$$\{\tau_j(l) \to \sigma_i(l+1) : l = 1 \ldots, n-1\}.$$

In terms of the biased cost function, the biased cost of the partial transfer route $P_{i,j}$ becomes

$$\begin{aligned}\widetilde{w}(P_{i,j}) =& \sum_{k=1}^{n} \widetilde{w}(\sigma_i(k), \tau_j(k)) + \sum_{l=1}^{n-1} \widetilde{w}(\tau_j(l), \sigma_i(l+1)) \\ =& \beta \cdot \sum_{k=1}^{n} w(\sigma_i(k), \tau_j(k)) + \sum_{l=1}^{n-1} w(\tau_j(l), \sigma_i(l+1)).\end{aligned} \tag{1.4.14}$$

This result is easily extended to obtain the biased cost $L(\Pi)$ of a transfer route Π (or, for the permutable sequence case, $L(\Pi_\psi)$ of Π_ψ with respect to ψ)

$$L(\Pi) = \sum_{\ell=1}^{m} \widetilde{w}(P_{\ell-1,\ell}) \tag{1.4.15}$$

$$L(\Pi_\psi) = \sum_{\ell=1}^{m} \widetilde{w}(P^{\psi}_{\psi(\ell-1),\psi(\ell)}). \tag{1.4.16}$$

One of the most notable contributions of this thesis is improving the known approximation ratio of algorithms for building alternating Hamiltonian paths and cycles with a biased cost function. The biased cost function in turn accounts for any additional effort that the grasp-and-delivery robot needs to exert when actually grasping a pin and transporting it from location u to location v, making that cost greater than a mere repositioning from u to v. This scenario was first explored in the scope of the (Fixed-sequence) RRP by Shurbevski et al. [54], to be later generalized in a broader setting [59]. We believe that because of the fact that this extension has applications far beyond the scope of the repetitive routing problem, it merits an independent main elaboration in general terms. An entire chapter is devoted to this facet of the problem, Chapter 3: The Biased Bipartite TSP. After the general exhibition, we devote some attention to the consequences the biased asymmetric cost function extension has on the repetitive routing problem and how the cost bias can be incorporated with the rest of the solution techniques developed for the Fixed and Permutable-sequence versions of the RRP.

2 Preliminaries

The previous chapter introduced the Fixed and Permutable-sequence RRP, the biased cost function generalization, and gave us a framework in which to orient ourselves should we like to think of the results described in this thesis in a tangible and intuitive way. Before proceeding to elaborating our solution approaches, we ought to become familiar with necessary tools and terminology which are used.

2.1 Approximation Algorithms

Approximation algorithms are an efficient approach for solving notoriously difficult NP-complete optimization problems. It is a widely held opinion that NP-complete optimization problems do not admit any polynomial time algorithm that always produces an optimal solution [26]. Due to the definition of the class NP, a polynomial time algorithm to solve any NP-complete problem implies there is a polynomial time algorithm for all problems in the class NP. On the other hand, given the exuberant amount of effort put into research towards that goal, and the growing number of problems regularly recognized to be NP-complete, the possibility of such a breakthrough is diminishingly small.

However, while seeking an optimal solution to an NP-complete optimization problem can be prohibitively time and resource consuming, it is often the case that a reasonably good solution delivered almost instantly would satisfy many an optimization query. This is exactly the case where approximation algorithms are sought for. The study of approximation algorithms is a fascinating area in itself, but here we only provide basic definitions and terminology to be used in the remainder of the thesis. Further information can be found in authoritative textbooks [61, 64].

Let us assume that we are faced with a minimization problem $\mathcal{P}$. For a given instance of problem $\mathcal{P}$, let P^* be the value of an optimal solution, such that $P^* \leq P$ for any other value P of a feasible solution for the given instance. An approximation algorithm ALG is such that for any instance of $\mathcal{P}$, it can produce a feasible solution of value P' in polynomial time. We call the value

$$\alpha = \sup \left\{ \frac{P'}{P^*} : \text{over all possible instances of } \mathcal{P} \right\}, \tag{2.1.1}$$

the approximation factor of algorithm ALG, and usually say that ALG is an α-approximation algorithm. Note that Eq. (2.1.1) implies that for a given instance of problem $\mathcal{P}$, and a feasible solution P' obtained by algorithm ALG,

$$P' \leq \alpha P \tag{2.1.2}$$

holds for any value P of a feasible solution for the same instance.

2.2 Spaces, Distances and Metrics

In general $\mathbb{R}$ stands for the set of reals, $\mathbb{Z}$ for the set of integers, and we use $\mathbb{R}_+$ and $\mathbb{Z}_+$, to denote the sets of nonnegative reals and integers, respectively. Let S be a set, possibly infinite and uncountable. A distance function $d : S \times S \to \mathbb{R}$ is a mapping of ordered pairs of elements of S to a real number. The distance function d is *metric* if for all $x, y, z \in S$ the following conditions are satisfied:

a) $d(x, y) \geq 0$,

b) $d(x, y) = 0 \iff x = y$,

c) $d(x, y) = d(y, x)$,

d) $d(x, y) \geq d(x, z) + d(z, y)$.

Condition c) is commonly referred to as *symmetry*, while condition d) is known as the *triangle inequality*. These two notions play a vital role in the solution techniques developed throughout this thesis.

If d is a metric distance function over the set S, the ordered pair (S, d) is called a *metric space*. A commonly encountered metric space is the Euclidean space, where $S = \mathbb{R}^2$, and distance is defined via the ℓ_2-norm.

2.3 Graph Theory

Graph theory is said to have been initiated by Leonard Euler's approach to solve the problem of the seven bridges of Königsberg, formalized as a publication in 1736 (engaging historical notes and further information may be found in the extensive book of Moore and Mertens [41]). Since then, graph theory has grown into a well defined and versatile branch of mathematics, enjoying a variety of applications across theoretical and applied sciences. As an indispensable part of any discrete mathematics or combinatorial optimization textbook, concise introduction can be found in any related textbook (e.g., [20, 36, 50]), textbooks on algorithms (e.g., [18, 35]), those bridging the above topics (e.g., [42]), as well

as dedicated textbooks (e.g., [2, 15]). For the sake of completeness, we will revise some of the standard notation used in graph theory, but limit ourselves to exhibiting the terminology and notation essential for our exposition.

The ordered pair $G = (V, E)$ is a graph, where V is a set of vertices and E is a set of edges, i.e., a family of subsets of V with cardinality 2. Throughout this thesis, we always assume that a given graph $G = (V, E)$ is a connected undirected graph. The vertex set and the edge set of a graph G are denoted by $V(G)$ and $E(G)$, respectively. We allow for parallel edges, or think of $G = (V, E)$ as a multigraph. Thus, E is a multiset of elements in $V \times V$. We will make use of the multiset sum function which preserves element multiplicity, denoted by the symbol $\uplus$, as well as the shorthand $k \cdot E$ for $\biguplus_{i=1}^{k} E$. We use $\{u, v\}$, $u, v \in V(G)$ to reference any and all $e \in E(G)$ such that e is incident with u and v. A graph G is *complete* if for every pair of vertices $u, v \in V(G)$, $\{u, v\} \in E(G)$. For a complete graph induced by a set of vertices V, we write $G[V]$. By definition, $V(G[V]) = V$ and $E(G[V]) = V \times V$.

A subgraph G' of G is such that $V(G') \subseteq V(G)$, and $E(G') \subseteq E(G)$ with the condition that if $e = \{u, v\} \in E(G')$ then necessarily both $u \in V(G')$ and $v \in V(G')$. For $u \in V(G)$, $d_G(u)$ denotes the degree of the vertex u in the graph G, i.e., the number of edges in $E(G)$ incident with u. A graph is weighted if we are given some weight function $w : E(G) \to \mathbb{R}_+$ over the graph's edges. For any subset of edges $E' \subseteq E$, $w(E')$ denotes $\sum_{e \in E'} w(e)$. Similarly, for a subgraph G' of G, $w(G')$ denotes $\sum_{e \in E(G')} w(e)$. For our purpose all parallel edges are of equal weight, and $\forall e \in E(G)$, $e = \{u, v\}$, we equate the expressions $w(e)$ and $w(u, v)$. As a consequence, the expressions $w(u, v)$ and $w(v, u)$ are also identical for all $e = \{u, v\} \in E(G)$, and we can state that the weight function w is symmetric. If w is the edge weight function of a complete graph, and it holds that

$$w(u, v) \leq w(u, q) + w(q, v), \quad \forall q, u, v \in V(G), \tag{2.3.3}$$

then it is said that w satisfies the triangle inequality. If the edge weight w is symmetrical, satisfies the triangle inequality, and in addition it holds that

$$w(u, v) = 0 \Leftrightarrow u = v, \quad u, v \in V(G) \tag{2.3.4}$$

then the graph $G = (V, E)$ is said to be metric. This is a common case when vertices in $V(G)$ are used to represent elements from some set S, and a metric distance function d in S serves to define the edge weight w.

A *bipartite graph* $G = (B, W; E)$ is such that $V(G) = B \cup W$, $B \cap W = \emptyset$, and $E(G) \subseteq B \times W$. A bipartite graph G is complete if $E(G) = B \times W$, and *balanced* if $|B| = |W|$. A property similar to the triangle inequality can be extended over

complete bipartite graphs, into the quadrangle inequality

$$w(u,v) \leq w(u,q) + w(q,y) + w(y,v), \quad \forall u,y \in B, q,v \in W. \tag{2.3.5}$$

Let $G = (B, W; E)$ be a given bipartite graph with an edge weight function $w : E \to \mathbb{R}_+$. Let us consider a single vertex partition B, without loss of generality. Following the definition of bipartite graphs, for any two vertices $u, y \in B$, there exists no edge $\{u, y\}$ in the edge set E. However, we may construct and make use of the complete graph $G[B]$ over the vertex set B. Given a bipartite graph $G = (B, W; E)$ with an edge weight function $w : E(G) \to \mathbb{R}_+$ which satisfies the quadrangle inequality, we can extend the edge weight function over the induced graph $G[B]$ as follows

$$w(u,y) = \min_{q \in W}\{w(u,q) + w(q,y)\} \quad \forall\, u, y \in B. \tag{2.3.6}$$

Lemma 2.1. *[59] For a given complete bipartite graph $G(B, W; E)$ with a symmetric edge weight function $w : E \to \mathbb{R}_+$ satisfying the quadrangle inequality, let $G[B]$ be a complete graph over the vertex set B. The extension of w as an edge weight function of $G[B]$ of Eq. (2.3.6) is symmetric and satisfies the triangle inequality.*

Given a graph G, a *walk* in G can be thought of as a sequence of vertices $v_1, v_2, \ldots, v_k$, under the condition that $\{v_l, v_{l+1}\} \in E(G)$, $l = 1, 2, \ldots, k-1$. A *path* in G is a walk where for $1 \leq i \neq j \leq k$, $v_i \neq v_j$. A path described by the vertex sequence $v_1, v_2, \ldots, v_k$ is referred to as a path from v_1 to v_k, or a v_1-v_k path. A *cycle* is a path where $v_1 = v_k$. Paths and cycles may be thought of as subgraphs of G.

There are several more important structures which arise in graphs. Given a graph $G = (V, E)$, a subgraph G' of G is called *spanning* if $V(G') = V(G)$. The graph G is *connected* if for every pair of vertices, $u, v \in V(G)$ there exists a u-v path in G. A tree is a connected acyclic graph. A *spanning tree* T of a given connected graph $G = (V, E)$ is a connected spanning acyclic subgraph of G. A *matching* $M \subseteq E(G)$ is such that each vertex in $V(G)$ is incident with at most one edge in M. A matching $M \subseteq E(G)$ is *perfect* if each vertex in $V(G)$ is incident with exactly one edge in M. As stated so far, we think of spanning trees as subgraphs, and of matchings as edge sets. Given a graph $G = (V, E)$ with an edge weight function w, computing minimum (with respect to w) weight spanning trees and matchings are well studied combinatorial optimization problems. Algorithms running in polynomial time complexity exist and are indispensably included in classic and modern textbooks, such as [18, 20, 35, 36, 39, 50], to name but a few.

A collection $\mathcal{R} = (R_1, R_2, \ldots, R_k)$ of disjoint cycles (including cycles on 2 vertices, but no self-loops) which collectively include all vertices in $V(G)$, i.e.,

$$V(R_i) \cap V(R_j) = \emptyset,\ 1 \leq i \neq j \leq k, \tag{2.3.7}$$

$$V(\mathcal{R}) := \bigcup_{i=1}^{k} V(R_i) = V(G), \tag{2.3.8}$$

is called a *cycle cover* of the graph G. It holds

$$d_{\mathcal{R}}(v) = 2,\ \forall v \in V(\mathcal{R}). \tag{2.3.9}$$

A *Hamiltonian path* P is a connected spanning subgraph of G such that

$$d_P(v) \leq 2, \quad \forall v \in V(G). \tag{2.3.10}$$

A *Hamiltonian cycle* H is a connected spanning subgraph of G such that

$$d_H(v) = 2, \quad \forall v \in V(G). \tag{2.3.11}$$

The problem of finding a Hamiltonian cycle H of minimum $w(H)$ is commonly referred to as the traveling salesman problem (TSP). The TSP itself has enchanted mathematics and computer science researchers, enthusiasts and hobbyists for several decades now. It is a landmark problem in combinatorial optimization springing in many forms. Numerous papers and research monographs have been devoted to the TSP and its various versions, as well as solution methods. Even entire books have been devoted to the topic (e.g., [6, 17]). At this point it is worth noting that as Hamiltonian paths and cycles can be described by permutations (the order vertices in a graph are visited), so can permutations be derived from a given Hamiltonian cycle or path.

For a complete, balanced bipartite graph $G = (B, W; E)$, let $n := |B|(= |W|)$ and let σ and τ be permutations on the points of B and W, respectively. A traversal of a Hamiltonian path P in G is of the form

$$\sigma(1) \to \tau(1) \to \sigma(2) \to \cdots \to \tau(n-1) \to \sigma(n) \to \tau(n). \tag{2.3.12}$$

We term Hamiltonian paths (and cycles) in bipartite graphs *alternating*, for points in B and W appear alternately. An alternating Hamiltonian cycle H contains the additional edge $\tau(n) \to \sigma(1)$. In certain explicitly stated cases, when using an indexing device, e.g., $l = 1, 2, \ldots, n$, we allow it to wrap around, i.e.,

$$l := \begin{cases} l + n, & l \leq 0, \\ l - n, & l > n. \end{cases} \tag{2.3.13}$$

As subgraphs of G, Hamiltonian paths and cycles are undirected. However, once we settle for a way to traverse them, they assume an *orientation*. An orientation defines the direction in which each edge of a path or a cycle is traversed. This notion is essential when examining the biased generalization of the cost function of alternating Hamiltonian paths and cycles.

2.3.1 The RRP in Graph-theoretic Terminology

In the remainder of this section, we will try to re-state the terminology used to provide an intuitive description of the repetitive routing problem and its forms with graph-theoretic nomenclature, which will allow us to use the tools developed in graph theory to tackle the RRP.

First, let us revise the essential subproblem of a single transition from configuration C_i to configuration C_j. Let $G_{i,j} = (C_i, C_j; C_i \times C_j)$ be a complete bipartite graph. Following this correspondence, we refer to each C_ℓ, $(\ell = 0, 1, \ldots, m)$ as a vertex set. The vertices of $V(G_{i,j})$ are actually points from a surface S. Given a distance function d over S, we define the edge weight function $w(u, v)$, $u, v \in V(G)$, (without loss of generality, $u \in C_i$, $v \in C_j$) as $w(u, v) := d(u, v)$. Similarly, the distance function d can be extended as an edge weight function over the induced complete graphs $G[C_i]$ and $G[C_j]$. If (S, d) is a metric space, then the graph edge weight function w would be metric. Recalling Eq. (1.2.2), a transition was defined via permutations σ_i and τ_j on the vertex sets C_i and C_j, respectively, and denoted as $P_{i,j}$. Referring to Eq. (2.3.12), we see that the path $P_{i,j}$ is an alternating Hamiltonian path in the graph $G_{i,j}$. The expression for the cost $w(P_{i,j})$ from Eq. (1.2.3) retains its validity without any modification.

The Fixed-sequence Repetitive Routing Problem presents a much more challenging problem. A transfer route as a straightforward solution can indeed be thought of as a sort of a repetitive walk in a graph, but cannot be categorized as gracefully as partial transfer routes can be classified as alternating Hamiltonian paths. However, we rely on the consecutiveness constraint and the relative independence between different transitions to define a transfer route as a concatenation of m alternating Hamiltonian paths. Therefore, given $\mathcal{C} = \{C_\ell : \ell = 1, 2, \ldots, m\}$, we examine the bipartite graphs $G_{\ell-1,\ell}, \ell = 1, 2, \ldots, m$. A transfer route Π can be thought of as a concatenation of m alternating Hamiltonian paths, $P_{\ell-1,\ell}, \ell = 1, 2, \ldots, m$, under the condition that all alternating Hamiltonian paths satisfy the consecutiveness constraint, Eq. (1.3.7). At this point, we would like to support Proposition 1.1 made in Chapter 1: Introduction.

We consider an alternating Hamiltonian path $P_{\ell-1,\ell}$ for some $\ell \in [1, m]$, which violates the consecutiveness constraint, i.e., $\sigma_{\ell-1}(1) \neq \tau_{\ell-1}(n)$. For the purpose

of completeness, we introduce $\tau_0(n) := c_1^0$, in compliance with the assumption that $\sigma_0(1) = c_1^0$. Let us assume that $\sigma_{\ell-1}(\tau_{\ell-1}(n))^{-1} = k$, for some $k \in [1, n]$ ($\sigma_{\ell-1}(k) = \tau_{\ell-1}(n)$). As a consequence, we need to consider an additional edge $\tau_{\ell-1}(n)$-$\sigma_{\ell-1}(1)$ traversed in the direction from $\tau_{\ell-1}(n) \in C_{\ell-1}$ to $\sigma_{\ell-1}(1) \in C_{\ell-1}$ (a black dashed arrow in Figure 2.1 (a)). The edge $\tau_{\ell-1}(n)$-$\sigma_{\ell-1}(1)$ is necessary in order to guarantee that the alternating Hamiltonian path $P_{\ell-1,\ell}$ forms a contiguous walk with respect to the last visited vertex, $\tau_{\ell-1}(n) \in C_{\ell-1}$. We can partly modify $P_{\ell-1,\ell}$ into $P'_{\ell-1,\ell}$, respectively $\sigma_{\ell-1}$ into $\sigma'_{\ell-1}$ and τ_ℓ into τ'_ℓ as follows.

For $i = 1, 2, \ldots, n$:

$$\sigma'_{\ell-1}(i) := \begin{cases} \sigma_{\ell-1}(k-i+1), & i \leq k, \\ \sigma_{\ell-1}(i), & i > k; \end{cases} \tag{2.3.14}$$

$$\tau'_{\ell}(i) := \begin{cases} \tau_{\ell}(k-i), & i < k, \\ \tau_{\ell}(i), & i \geq k. \end{cases} \tag{2.3.15}$$

The result of the procedure described above takes the form as illustrated in Figure 2.1 (b). Most importantly, note that the set of edges traversed in the direction from the vertex set $C_{\ell-1}$ to C_ℓ has been modified from the original alternating Hamiltonian path $P_{\ell-1,\ell}$ (indicated by bold gray arrows in Figure 2.1). The consequence of the above modification procedure is expressed as Lemma 2.2.

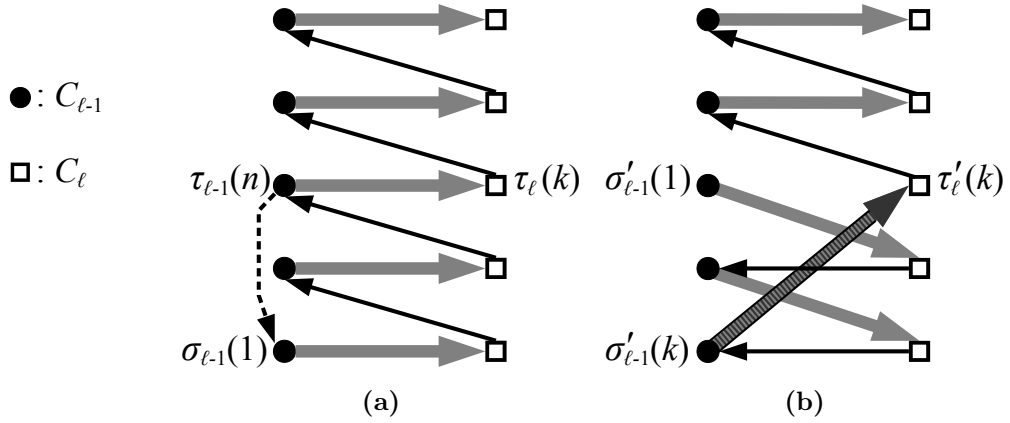

Figure 2.1. (a) An illustration of an alternating Hamiltonian path $P_{\ell-1,\ell}$ violating the consecutiveness constraint; (b) A modified alternating Hamiltonian path $P'_{\ell-1,\ell}$ which satisfies the consecutiveness constraint.

Lemma 2.2. *Given an alternating Hamiltonian path $P_{\ell-1,\ell}$, which violates the consecutiveness constraint with respect to the last visited vertex, $\tau_{\ell-1}(n) \in C_{\ell-1}$, ($\ell = 2, 3, \ldots, m$), we can modify $P_{\ell-1,\ell}$ into an alternating Hamiltonian path*

$P'_{\ell-1,\ell}$ such that the consecutiveness constraint is satisfied, i.e., $\sigma'_{\ell-1}(1) = \tau_{\ell-1}(n)$. Under the assumption that the cost function w is symmetric and satisfies the triangle inequality, it holds that

$$w(P'_{\ell-1,\ell}) \leq w(P_{\ell-1,\ell}) + w(\tau_{\ell-1}(n), \sigma_{\ell-1}(1)). \tag{2.3.16}$$

Proof. Examining the expression from Eqs. (2.3.14) and (2.3.15), we point out the following relations

$$\begin{aligned}
\sigma'_{\ell-1}(1) &= \sigma_{\ell-1}(k) = \tau_{\ell-1}(n),\\
\sigma'_{\ell-1}(k) &= \sigma_{\ell-1}(1),\\
\tau'_{\ell}(k) &= \tau_{\ell}(k).
\end{aligned}$$

There is an additional edge $\tau_{\ell-1}(n)$-$\sigma_{\ell-1}(1)$ with weight $w(\tau_{\ell-1}(n), \sigma_{\ell-1}(1))$ which needs to be traversed from the last visited vertex, $\tau_{\ell-1}(n) \in C_{\ell-1}$, to $\sigma_{\ell-1}(1)$, the starting vertex of the path $P_{\ell-1,\ell}$. Bringing to attention Figure 2.1, we notice that under the assumption of symmetry for w, meaning that the direction in which edges are traversed is irrelevant, the only difference between $P_{\ell-1,\ell}$ and $P'_{\ell-1,\ell}$ is with the edge $\sigma_{\ell-1}(k)$-$\tau_{\ell}(k)$, which is replaced by the edge $\sigma'_{\ell-1}(k)$-$\tau'_{\ell}(k)$. Notice that the newly introduced edge $\sigma'_{\ell-1}(k)$-$\tau'_{\ell}(k)$ (with respect to $\sigma_{\ell-1}$ and τ_{ℓ}, this is the same as edge $\sigma_{\ell-1}(1)$-$\tau_{\ell}(k)$) replaces the additional aligning edge $\tau_{\ell-1}(n)$-$\sigma_{\ell-1}(0)$ and an edge $\sigma_{\ell-1}(k)$-$\tau_{\ell}(k)$. Due to the triangle inequality we have

$$\underbrace{w(\sigma'_{\ell-1}(k), \tau'_{\ell}(k))}_{\text{Newly introduced edge}} \leq \underbrace{w(\tau_{\ell-1}(n), \sigma_{\ell-1}(1))}_{\text{Additional aligning edge}} + \underbrace{w(\sigma_{\ell-1}(k), \tau_{\ell}(k))}_{\text{Replaced edge}}$$

from where the claim follows. □

Lemma 2.2 justifies Proposition 1.1 from Chapter 1, and also our choice of terminology for referring to a transfer route as feasible if it satisfies the consecutiveness constraint. Unfortunately though, the assumptions behind this claim, namely, the need for symmetry of the cost function w fail when we are faced with the biased versions of the problem. We will examine certain approaches to circumvent this issue in Chapter 4.

The Permutable-sequence RRP would rely on the same approach of concatenating alternating Hamiltonian paths under the consecutiveness constraint. In this case additional care needs to be taken due to the fact that alternating Hamiltonian paths $P^{\psi}_{\psi(\ell-1),\psi(\ell)}$, $\ell = 1, 2, \ldots, m$, are computed with respect to a permutation ψ on the interval $[0, m]$ with the restriction that $\psi(0) = 0$.

2.4 Introduction of a Bias in Broader Terms

The introduction of a bias factor in the cost function further complicates matters, for the assumption of Lemma 2.2 is no longer valid. While we oblige ourselves to investigate how this issue can be resolved towards a solution to the RRP, we believe that the mere introduction of the bias factor has far-reaching implications beyond the scope which initiated this research. Therefore, with the entrance of the bias factor $\beta \geq 1$, we would like to extend our view by expressing the problem we face in broader terminology, for which graph theory has provided a more than suitable framework.

Let $G = (B, W; E)$ be a complete, balanced bipartite graph, endowed with a symmetric edge weight function $w(u, v)$ (without loss of generality, $u \in B$ and $v \in W$). For a given bias $\beta \geq 1$, let the edge cost $\widetilde{w}(u, v)$ be

$$\widetilde{w}(u,v) = \begin{cases} \beta w(u,v), & u \in B, v \in W, \\ w(u,v), & u \in W, v \in B. \end{cases} \tag{2.4.17}$$

In terms of permutations σ and τ on the vertex sets B and W, respectively, as in Eq. (2.3.12), the biased cost $L(P)$ of an alternating Hamiltonian path P in the bipartite graph G becomes

$$L(P) = \beta \sum_{i=1}^{n} w(\sigma(i), \tau(i)) + \sum_{i=1}^{n-1} w(\tau(i), \sigma(i+1)). \tag{2.4.18}$$

Similarly, the biased cost $L(H)$ of an alternating Hamiltonian cycle H in G is

$$L(H) = \beta \sum_{i=1}^{n} w(\sigma(i), \tau(i)) + \sum_{i=1}^{n} w(\tau(i), \sigma(i+1)). \tag{2.4.19}$$

At this point we formalize the problem incorporating a bias factor.

The Biased Bipartite Traveling Salesman Problem - BBTSP

Instance: A complete, balanced bipartite graph $G = (B, W; E)$, a symmetric edge weight function $w : E(G) \to \mathbb{R}_+$ which satisfies the quadrangle inequality, and a bias factor $\beta \geq 1$.

Objective: Find an alternating Hamiltonian cycle H^* in G such that $L(H^*)$ is minimized.

The Biased Bipartite TSP-path problem can be expressed equivalently in terms of an alternating Hamiltonian path P in the bipartite graph G. To simplify notation, we denote an instance of the BBTSP as the ordered triplet (G, w, β).

Using the above formulation, the BBTSP has been recently introduced by Shurbevski et al. [59]. Adopted to suit the purposes of the RRP, it is easily discerned that the BBTSP-path is equivalent to finding a partial transfer route $P_{i,j}$ for a single transition from configuration C_i to configuration C_j, by setting $B := C_i$ and $W := C_j$.

The (unbiased) bipartite TSP has been proved as an NP-complete problem [1, 37]. Furthermore, just like the TSP, it is hopeless to approximate the bipartite TSP within a constant factor in the general case, assuming that $\mathrm{P} \neq \mathrm{NP}$ [21, 49].

2.5 Matroid Theory

Matroids are versatile mathematical structures which generalize the notion of linear independence in vector spaces. Their invention is attributed to Whitney [63]. Matroids capture various concepts spawning from diverse areas in mathematics such as linear algebra, geometry, and graph theory, and have consistently found their way into influential combinatorial optimization works, e.g., [20, 36, 39, 50]. Understandably, matroids and matroid theory have been applied to a variety of problems, especially with examples in routing problems [10, 21, 40, 46], particularly in the works leading to this thesis, [30–33, 53–59], but also in other engineering domains [47, 48].

We proceed by an overview of elementary concepts which are necessary for our presentation. Further information regarding matroids and related subjects, in addition to presentations with relation to combinatorial optimization ([20, 36, 39, 50]), can be found in dedicated texts, e.g., [12, 27, 44].

Let E be a finite set, and $\mathcal{I}$ a family of subsets of E. The ordered pair $(E, \mathcal{I})$ is called an *independence system* if the subset family $\mathcal{I}$ satisfies the following conditions

i) $\emptyset \in \mathcal{I}$,

ii) if $I_1 \in \mathcal{I}$ and $I_2 \subseteq I_1$, then $I_2 \in \mathcal{I}$.

Subsets $I \subseteq E$ such that $I \in \mathcal{I}$ are called *independent*, otherwise, they are dependent sets. If in addition to the above two, the subset family $\mathcal{I}$ satisfies the condition

iii) if $I_1, I_2 \in \mathcal{I}$ and $|I_1| > |I_2|$, then there is some element $e \in I_1 - I_2$ such that $I_2 \cup \{e\} \in \mathcal{I}$,

then $\mathcal{I}$ is the family of independent sets of a *matroid*, $M = (E, \mathcal{I})$. The set E is called the *ground set* of the matroid $M = (E, \mathcal{I})$. A subset $C \subseteq E$ with $C \notin \mathcal{I}$

such that $C\backslash\{e\} \in \mathcal{I}$ for any $e \in C$ is called a *circuit* in $M = (E, \mathcal{I})$. Given $I \in \mathcal{I}$, if for some $e \in E\backslash I$, $I \cup \{e\} \notin \mathcal{I}$, then there is exactly one circuit in $I \cup \{e\}$, which we denote by $c(I, e)$, and $e \in c(I, e)$.

For a subset $I \subseteq E$, the function

$$r(I) = \max\{|J| : J \subseteq I,\ J \in \mathcal{I}\} \tag{2.5.20}$$

is called the *rank* function of the matroid $M = (E, \mathcal{I})$. Obviously $r(I) = |I|$ implies that $I \in \mathcal{I}$.

For a given matroid $M = (E, \mathcal{I})$, let $\mathcal{B} := \{B \in \mathcal{I} : B \cup \{e\} \notin \mathcal{I}, \forall e \in E - B\}$. The maximal inclusion-wise sets B are called *bases* of the matroid $M = (E, \mathcal{I})$. Condition iii) gives rise to an interesting property of matroids

$$|B_i| = |B_j|, \quad \forall B_i, B_j \in \mathcal{B}.$$

Consequently, for any $B \in \mathcal{B}$, it holds $r(B) = r(E)$.

2.5.1 Matroid Examples

There exist a variety of structures which have been identified as matroids. We will only define two matroids which are of interest to this thesis.

Let $G = (V, E)$ be a graph. Taking the edge set E to be a ground set, we define the following family of subsets over E

$$\mathcal{I}_1 := \{I \subseteq E : (V, I) \text{ is acyclic}\}. \tag{2.5.21}$$

Then, the independence system $(E, \mathcal{I}_1)$ is a matroid, which we denote as M_1. The matroid $M_1 = (E, \mathcal{I}_1)$ as defined above is known as the *graphic* matroid. The terms *circuit* in graphic matroids and *cycle* in graphs coincide.

Another example of interest to this thesis is the *partition* matroid. Let a given ground set E be partitioned into k disjoint subsets, $E = \bigcup_{i=1}^{k} E_k$. For given k non-negative integers d_i, $i = 1, 2, \ldots, k$, the subset family

$$\mathcal{I}_2 := \{I \subseteq E : |I \cap E_i| \leq d_i,\ i = 1, 2, \ldots, k\} \tag{2.5.22}$$

defines the family of independent sets of a matroid $M_2 = (E, \mathcal{I}_2)$, which is in fact, the partition matroid.

Other commonly encountered matroids include the uniform matroid, transversal matroid, paving matroid, etc.

2.5.2 Matroid Optimization

For practical purposes, we need some mechanism of recognizing whether for a given matroid $M = (E, \mathcal{I})$ and a subset $I \subseteq E$, I is independent (i.e., $I \in \mathcal{I}$) or not. This can be done by explicit enumeration of the set family $\mathcal{I}$, but since in general $|\mathcal{I}|$ may be as large as $2^{|E|}$, explicit enumeration can be impractical, to say the least. For this purpose, the concept of *oracle* has been introduced. Oracles may be given in many forms, depending on the intended application [36].

Matroids are perhaps most famed for their elegant capture of the concept of *greedy choice*, or the greedy algorithm, thereby deserving honorable mentions in otherwise mostly algorithmic textbooks [18, 35]. Let us attribute a weight $w(e)$ to all elements of the ground set E. For a subset $I \subseteq E$, let $w(I) = \sum_{e \in I} w(e)$. We introduce the problem of finding a maximum weight base B for a given matroid $M = (E, \mathcal{I})$ and a weight function w.

The Maximum Weight Matroid Base Problem

Instance: A matroid $M = (E, \mathcal{I})$ and a weight function $w : E \to \mathbb{R}$.

Objective: Find a base $B^* \in \mathcal{B}$ of the given matroid, such that $w(B^*) \geq w(B)$, for any other $B \in \mathcal{B}$.

Algorithm 1 The greedy algorithm

Input: A matroid $M = (E, \mathcal{I})$ defined by an oracle, and a weight function $w : E \to \mathbb{R}$.

Output: A base $B^* \in \mathcal{B}$ with $w(B^*) \geq w(B)$, $\forall B \in \mathcal{B}$.

1: Set $n \leftarrow |E|$;
2: Sort the elements in E such that $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_n)$;
{we have chosen to relabel the indexes for convenience}
3: Set $I \leftarrow \emptyset$;
4: **for** $i = 1$ to n **do**
5: **if** $I \cup \{e_i\} \in \mathcal{I}$ **then**
6: Set $I \leftarrow I \cup \{e_i\}$
7: **end if**
8: **end for**;
9: Set $B^* \leftarrow I$;
10: **return** B^*.

The greedy algorithm, given as Algorithm 1, allows us to solve the maximum weight matroid base problem. In fact, not only will the greedy algorithm find a base of maximum weight of all bases of the given matroid $M = (E, \mathcal{I})$, but also, at any time during the iteration of the algorithm, for a partially constructed solution I it holds that $w(I) \geq w(J)$ for any $J \in \mathcal{I}$ with $|I| = |J|$. The greedy algorithm can be used unmodified if we also seek for a minimum weight base (or an independent set of given cardinality), by setting $w(e) := -w(e)$ for all $e \in E$. Notable applications of the greedy algorithm include the single-server identical job scheduling problem [39] (interval scheduling [35]), which is the greedy algorithm applied to a transversal matroid, and the well-known Kruskal's algorithm for the minimum weight spanning tree problem, over a graphic matroid.

A more powerful application of matroid theory comes from the concept of matroid intersection. For two matroids given over the same ground set E, $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$, we define their intersection as $(E, \mathcal{I}_1 \cap \mathcal{I}_2)$. A subset $I \subseteq E$ is independent in $\mathcal{I}_1 \cap \mathcal{I}_2$ if I is independent both in $\mathcal{I}_1$ and $\mathcal{I}_2$.

The Cardinality Matroid Intersection Problem

Instance: Two matroids $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$.

Objective: Find a set $B \in \mathcal{I}_1 \cap \mathcal{I}_2$ such that $|B| \geq |J|$ for any $J \subseteq E, J \in \mathcal{I}_1 \cap \mathcal{I}_2$.

We call such a set $B \subseteq E$ a *common base* of the two matroids, despite the fact that it might not necessarily be a base of one of the matroids. Additionally, given a weight function $w : E \to \mathbb{R}$ over the elements of the ground set E, we might be interested in seeking an optimal weight common base.

The Weighted Matroid Intersection Problem

Instance: Two matroids $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$, and a weight function $w : E \to \mathbb{R}$.

Objective: Find a set $I \in \mathcal{I}_1 \cap \mathcal{I}_2$ such that $w(I) \geq w(J)$ for any $J \subseteq E, J \in \mathcal{I}_1 \cap \mathcal{I}_2$.

At this point we should bring to attention that such an $I \subseteq E$ with $I \in \mathcal{I}_1 \cap \mathcal{I}_2$ that maximizes $w(I)$ need not be a common base of the matroids M_1 and M_2, even if the weight function w is positive [50] (Volume B, pp. 710).

In general, the intersection of two matroids is not a matroid. Therefore, using the greedy algorithm for optimizing over independent sets in $\mathcal{I}_1 \cap \mathcal{I}_2$ is not guaranteed to succeed.

Several well-known and important combinatorial problems can be posed as a matroid intersection problem. For example, bipartite matching can be thought of as the intersection of two partition matroids, and branchings in directed graphs as an intersection of a graphic and a partition matroid. Knowing that representative combinatorial problems which can be given as the intersection of two special types of matroids admit efficient solution algorithms, a natural question is whether there is a general approach for matroid intersection over any types of matroids.

An empowering positive answer to the question above comes in the form a matroid intersection algorithm. Detailed descriptions on various matroid intersection algorithms can be found in [50]. We mainly rely on the very clearly and concisely given algorithm by Frank [19]. Presently it suffices to note that it is an efficient algorithm, running in polynomial number of oracle calls, and that due to the implementation of the algorithm, it iteratively augments a set $I \in \mathcal{I}_1 \cap \mathcal{I}_2$, and at all times it holds that $w(I) \geq w(J)$, $\forall J \in \mathcal{I}_1 \cap \mathcal{I}_2$, $|I| = |J|$. Therefore, this algorithm can be used as an efficient means to find a maximum weight common base of two matroids, M_1 and M_2, despite the previously mentioned fact that a common base need not be of maximum weight over independent sets in $\mathcal{I}_1 \cap \mathcal{I}_2$. By flipping the signs of the weights of all elements in E, a common base of minimum weight can be found by the same weighted matroid intersection algorithm.

In addition to the well known combinatorial optimization problems of finding bipartite matchings, branchings and arborescences [36, 39, 50], weighted matroid intersection has been efficiently utilized toward solutions to some routing problems, including, but not limited to [10, 13, 14, 21, 40, 46], as well as [30, 32, 53–59].

2.5.3 Alternating Spanning Trees

We will make use of weighted matroid intersection to optimally solve a problem which would provide a lower bound on an optimal solution to the RRP and the Bipartite TSP, thereby facilitating our quest for efficient approximation algorithms.

Let $G = (B, W; E)$ be a given complete bipartite graph, i.e., $E := B \times W$. For our purpose, we assume that the graph G is balanced, i.e., $|B| = |W|$. We define an *alternating* tree T as an acyclic subgraph of G, such that $d_T(v) \leq 2$, for all $v \in B$ (the vertex partition B is taken without loss of generality). An alternating tree T is spanning if it is a spanning subgraph of the graph G.

The Minimum Weight Alternating Spanning Tree Problem

Instance: A complete, balanced bipartite graph $G = (B, W; E)$ with an edge weight function $w : E \to \mathbb{R}$.

Objective: Find an alternating spanning tree T^* of G, such that $w(T^*) \leq w(T)$, for any other alternating spanning tree T of the graph G.

The minimum weight alternating spanning tree problem can be posed as a weighted matroid intersection problem over a graphic and a partition matroid. Given the bipartite graph $G = (B, W; E)$, let $n := |B|(= |W|)$. We define two matroids over the edge set E of G as a ground set.

$M_1 = (E, \mathcal{I}_1)$**:** $\mathcal{I}_1 := \{I \subseteq E :$ the graph $(B, W; I)$ is acyclic$\}$;

$M_2 = (E, \mathcal{I}_2)$**:** $\mathcal{I}_2 := \{I \subseteq E : |\{\{u, v\} \in I : v \in W\}| \leq 2, \forall u \in B\}$.

The former matroid, $M_1 = (E, \mathcal{I}_1)$, is straightforwardly the graphic matroid as described in Section 2.5.1, whereas the latter, $M_2 = (E, \mathcal{I}_2)$, can be thought of as a partition matroid, by partitioning the edge set E as $E = \bigcup_{u \in B} E_u$, where for each $u \in B$, $E_u := \{\{u, v\} : v \in W\}$, and all associated $d_u := 2$.

The intersection of the two matroids as defined above results in an alternating forest. Consequently, an alternating spanning tree is a common base of the two matroids, and a minimum weight alternating spanning tree can be obtained by a weighted matroid intersection algorithm.

Due to the degree limitation of vertices in one of the vertex sets of a bipartite graph, alternating spanning trees have a useful property, formalized in the following claim.

Lemma 2.3. *Given a balanced bipartite graph $G = (B, W; E)$, assume there exists an alternating spanning tree T of G such that $d_T(u) \leq 2, \forall u \in B$. Then, there exists a unique $u^* \in B$ such that $d_T(u^*) = 1$.*

Proof. Let $n := |B|(= |W|)$. By definition, the vertex sets B and W form a partition of $V(G)$, i.e., $V(G) = B \cup W$ and $B \cap W = \emptyset$. Therefore,

$$|V(G)| = |B \cup W| = |B| + |W| = 2n. \tag{2.5.23}$$

It is a well known fact that all alternating trees of a given connected graph on k vertices have exactly $k - 1$ edges (see, e.g., [18]). Also, in general spanning trees, each vertex needs to be incident with at least one edge in the spanning tree, to

maintain connectedness. Since alternating spanning trees must also be spanning trees in G, and by Eq. (2.5.23) the graph G has $2n$ vertices, we get that

$$|E(T)| = 2n - 1, \tag{2.5.24}$$

$$d_T(u) \geq 1, \forall u \in B. \tag{2.5.25}$$

Further, since for all $u, y \in B$ (respectively, $q, v \in W$), $\nexists\{u, y\} \in E$ (resp., $\nexists\{q, v\} \in E$), every edge in $E(T)$ can be accounted for by examining the incidence with vertices of either vertex partition. Therefore, it holds

$$|E(T)| = \sum_{u \in B} d_T(u). \tag{2.5.26}$$

Now, by the definition of alternating spanning trees, $d_T(u) \leq 2, \forall u \in B$, but we cannot have $d_T(u) = 2, \forall u \in B$, otherwise

$$|E(T)| = \sum_{u \in B} d_T(u) = 2n,$$

which contradicts Eq. (2.5.24), so there must be at least one vertex $u^* \in B$ with $d_T(u^*) = 1$. On the other hand, if there is another vertex $u' \in B$ with $d_T(u') = 1$, then

$$\sum_{u \in B} d_T(u) = \sum_{u \in B \setminus \{u^*, u'\}} d_T(u) + d_T(u^*) + d_T(u') = 2(n-2) + 2 = 2n - 2,$$

which again contradicts Eq. (2.5.24), and confirms the uniqueness of u^*. □

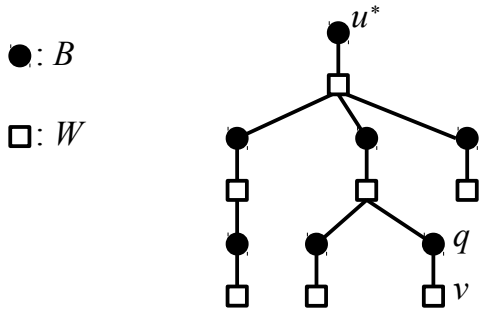

Figure 2.2. An alternating spanning tree with degree restriction on vertices in the vertex set B.

An alternating tree is illustrated in Figure 2.2. The alternating spanning tree in Figure 2.2 is not necessarily rooted at the vertex u^*, the choice of representation

is merely to bring to focus the unique $u^* \in B$ with degree one with respect to the alternating spanning tree.

A more interesting property of alternating spanning trees is that they always contain a perfect bipartite matching of the graph they span. This insight has been exploited by Shurbevski et al. [59] to derive an improved approximation factor algorithm for the BBTSP. Unfortunately, there lacks a formal proof of this claim, and we hereby would like to rectify this oversight.

Lemma 2.4. *Given a balanced bipartite graph $G = (B, W; E)$, assume there exists an alternating spanning tree T of G. The edge set of T, $E(T)$, contains a unique perfect bipartite matching of the graph G.*

Proof. Assume without loss of generality that the alternating spanning tree T is such that $d_T(u) \leq 2$, $\forall u \in B$. The above claim can be proved inductively over the number of vertices in each of the graph's partitions, $n := |B|(= |W|)$.

Case $n = 1$: The complete bipartite graph G is at the same time an alternating tree, and the single edge is the unique perfect matching of G.

Case $n > 1$: Because the alternating spanning tree contains exactly one leaf $u^* \in B$, there must exist at least one leaf $v \in W$, with a single neighbor $q \in B$ (Figure 2.2). Mark the edge $\{q, v\}$ as an edge of the matching, and remove the vertices $q \in B$ and $v \in W$. In this way, we have obtained an alternating spanning tree on a bipartite graph where each vertex set contains $n - 1$ vertices. □

Alternating spanning trees in bipartite graphs have been utilized to obtain efficient approximation algorithms [10, 13, 14, 21, 30, 32, 53–59]. All of the above works bring to attention the high computational complexity of a general weighted matroid intersection when used for computing minimum weight alternating spanning trees. For a bipartite graph $G = (B, W; E)$ with $|B| = |W| = n$, the computational complexity according to the description of a matroid intersection algorithm by Frank [19, 20] is $O(n^7)$. However, with a different implementation of a matroid intersection algorithm and a slightly more careful translation of the general matroid-related terms to the special case of alternating trees, we can obtain better bounds on the time complexity in which a minimum weight alternating tree can be computed.

Lemma 2.5. *Given a complete bipartite graph $G = (B, W; E)$ with $|B| = |W| = n$, a minimum weight alternating spanning tree T can be computed in $O(n^4)$ time complexity by a general weighted matroid intersection algorithm.*

Lemma 2.5 follows from Corollary 41.10 a, pp. 712 in Schrijver [50].

3 The Biased Bipartite TSP

We familiarized ourselves with essential concepts and methodology to proceed with the analysis and solution of the problems set forth so far. In this chapter, we develop a general algorithm for approximating alternating Hamiltonian paths and cycles in bipartite graphs, under biased conditions.

3.1 Introduction

The traveling salesman problem (TSP) is a landmark problem in combinatorial optimization (e.g., Cook [17]). Its bipartite analogue is as follows. Given a bipartite graph $G = (B, W; E)$ with an edge weight function $w : E \to \mathbb{R}_+$, find a shortest (with respect to w) alternating tour which visits every point of $B \cup W$ exactly once. We assume that the weight function w is symmetric and satisfies the quadrangle inequality (Eq. (2.3.5), the bipartite analogue of the triangle inequality). We introduced the bipartite TSP and its biased generalization in Chapter 2, Section 2.4, where it was pointed out that not only is it an NP-complete optimization problem ([1, 37]), but also finding an approximate solution with a constant factor approximation guarantee in general is at least equally challenging ([21, 49]).

The bipartite TSP has justly attracted attention due to its applicability in typical industrial settings where pick and place or grasp and delivery robots are employed with some material handling tasks [10, 13, 14, 21, 32, 60]. In addition, while for the metric case of the TSP there exist heuristics guaranteeing an approximation ratio as low $1 + \varepsilon$ ($\varepsilon > 0$, namely a polynomial time approximation scheme - PTAS) for the Euclidean metric [7, 8, 61] and factor 1.5 for any metric [16], the bipartite version of the TSP eludes these approaches. For the symmetric case, the best known approximation factor 2 has been independently reported by Chalasani et al. [14] and Frank et al. [21]. Extensive comparison of current heuristics and computational results have been reported by Baltz and Srivastav [10]. To the best of our knowledge, Shurbevski et al. [54] gave the first account examining the presence of a bias factor, and at the same time, demonstrated a constant, 16/7-factor approximation algorithm. The previously reported approximation ratio of 16/7 has been achieved by a composite heuristic.

Recently, Shurbevski et al. [59] have presented a novel heuristic procedure for building Hamiltonian cycles in bipartite graphs. We elaborate on this heuristic procedure, and show that for the biased case it is an approximation algorithm with an approximation ratio of

$$1 + \frac{1+\lambda}{\beta+\lambda} \tag{3.1.1}$$

where λ is a positive real parameter which depends on the problem instance and cannot be known upfront. On one hand, the above expression is bounded by a constant 2 for any positive real λ and $\beta \geq 1$ and thereby the proposed algorithm has a constant factor approximation ratio, improving the one from [54]. On the other hand, for a fixed λ, the above expression approaches 1 as β grows larger.

The presented approach by itself does not rely on approximating the metric TSP, however, it can be used as part of a composite heuristic to achieve an approximation ratio of

$$1 + \frac{2}{\zeta + \beta(2-\zeta)}, \tag{3.1.2}$$

where $1 < \zeta \leq 2$ is an approximation ratio for the metric TSP. The expression from Eq. (3.1.2) is also bounded above by a constant 2, but it is not dependent on an instance-specific parameter, and has a clear relationship with the bias β for a fixed $\zeta < 2$.

We focus exclusively on the version of the BBTSP where the edge weight function w is symmetric and satisfies the quadrangle inequality. We settle for this assumption because it has been shown [1, 21, 37, 49] that the bipartite TSP is not only NP-hard to solve, but also that in the general case, there is no constant factor approximation algorithm running in polynomial time complexity, under the assumption that $\mathrm{P} \neq \mathrm{NP}$.

3.2 Building Blocks

In this section we will exhibit some of the known lower bounds on the value of an optimal solution for the BBTSP, as well as add a few new insights into their correlations. The presented lower bounds are structures well known in combinatorial optimization, and will serve as building blocks for a new procedure for constructing alternating Hamiltonian cycles in bipartite graphs.

3.2.1 Known Lower Bounds of the BBTSP

We present some of the observations made in [54] concerning the lower bounds of an optimal solution for the BBTSP. Our analysis mainly concerns two combina-

torial structures in bipartite graphs; perfect matchings, and alternating spanning trees, which were appropriately defined in Chapter 2, Preliminaries. We will just briefly review their definitions.

Let $G = (B, W; E)$ be a (weighted) complete, balanced bipartite graph with an edge weight function $w : E(G) \to \mathbb{R}_+$. The edge weight function w is assumed to be symmetric and satisfying the quadrangle inequality (Eq. (2.3.5)). A perfect matching $M \subseteq E(G)$ is such that there is exactly one edge in M incident with any $u \in V(G)$. An alternating spanning tree T is a connected acyclic spanning subgraph of G such that

$$d_T(u) \leq 2, \quad \forall u \in B. \tag{3.2.3}$$

Both perfect matchings and alternating spanning trees are well studied combinatorial structures, e.g., [36, 50], and there exist polynomial time algorithms for computing perfect matchings and alternating spanning trees (of minimum weight) in bipartite graphs. Henceforth, let M^* denote a perfect matching in G of minimum weight $w(M^*)$, and T^* an alternating spanning tree with minimum weight $w(T^*)$.

Given an instance of the BBTSP, let H^* be an optimal solution, which minimizes the biased cost $L(H^*)$, as in Eq. (2.4.19). Analogously, for an instance of BBTSP-path, let P^* be a solution which minimizes $L(P^*)$, as in Eq. (2.4.18). The edges of $E(H^*)$ can be decomposed into two disjoint perfect matchings, $\overrightarrow{H^*}$ and $\overleftarrow{H^*}$, as in Figure 3.1 (a). In the case of an alternating Hamiltonian path P^*, we can see that it is composed of two disjoint edge sets $\overrightarrow{P^*}$ and $\overleftarrow{P^*}$, where $\overrightarrow{P^*}$ is a perfect bipartite matching, as in Figure 3.1 (b). Without loss of generality, we assume that H^* (respectively, P^*) is to be traversed as indicated by arrows in Figure 3.1 (a) (resp., Figure 3.1 (b)), and $\overrightarrow{H^*}$ (resp., $\overrightarrow{P^*}$) solely accounts for the bias term. The biased costs $L(H^*)$ and $L(P^*)$ are given by

$$L(H^*) = \beta w(\overrightarrow{H^*}) + w(\overleftarrow{H^*}), \tag{3.2.4}$$

$$L(P^*) = \beta w(\overrightarrow{P^*}) + w(\overleftarrow{P^*}). \tag{3.2.5}$$

It surely holds

$$w(M^*) \leq w(\overrightarrow{H^*}) \leq w(\overleftarrow{H^*}), \tag{3.2.6}$$

$$w(M^*) \leq w(\overrightarrow{P^*}). \tag{3.2.7}$$

Concerning alternating spanning trees in G, note that $w(T^*)$ is a lower bound on the weight of any alternating Hamiltonian cycle or path disregarding the bias

factor, i.e.,

$$w(T^*) \leq w(\overrightarrow{H^*}) + w(\overleftarrow{H^*}), \tag{3.2.8}$$
$$w(T^*) \leq w(\overrightarrow{P^*}) + w(\overleftarrow{P^*}). \tag{3.2.9}$$

At this point we would like to point out that the pairs of equations Eq. (3.2.6) and Eq. (3.2.7), also Eq. (3.2.8) and Eq. (3.2.9) are virtually identical for the case of both alternating Hamiltonian cycles and alternating Hamiltonian paths. Due to this fortunate circumstance, the complete analysis exhibited for alternating Hamiltonian cycles holds for alternating Hamiltonian paths as well.

Next we observe the graph $G[B]$ over the vertex set B. The vertex set B is chosen without loss of generality, and the same conclusions would hold for the complete graph over the vertex set W. An alternating Hamiltonian cycle in G does in fact visit each vertex in B exactly once, and can be shortcut to a Hamiltonian cycle of $G[B]$. We will use the extended w from Eq. (2.3.6) in relation with $G[B]$. For an optimal alternating Hamiltonian cycle H^* (respectively, an optimal alternating Hamiltonian path P^*), let $\hat{C}$ (resp., $\check{S}$) be the resulting shortcut, as given in Figure 3.1 (a) (resp., Figure 3.1 (b)). Due to Eq. (2.3.6), or in the metric case Eq. (2.3.3), we have

$$w(\hat{C}) \leq w(\overrightarrow{H^*}) + w(\overleftarrow{H^*}), \tag{3.2.10}$$
$$w(\check{S}) \leq w(\overrightarrow{P^*}) + w(\overleftarrow{P^*}). \tag{3.2.11}$$

Consequently, for optimal Hamiltonian cycle C^* and path S^* in $G[B]$ it holds

$$w(C^*) \leq w(\hat{C}) \leq w(\overrightarrow{H^*}) + w(\overleftarrow{H^*}), \tag{3.2.12}$$
$$w(S^*) \leq w(\check{S}) \leq w(\overrightarrow{P^*}) + w(\overleftarrow{P^*}). \tag{3.2.13}$$

For a minimum cost spanning tree T_B^* in the graph $G[B]$, from the established fact that

$$w(T_B^*) \leq w(C^*), \tag{3.2.14}$$
$$w(T_B^*) \leq w(S^*), \tag{3.2.15}$$

it follows that

$$w(T_B^*) \leq w(\hat{C}) \leq w(\overrightarrow{H^*}) + w(\overleftarrow{H^*}), \tag{3.2.16}$$
$$w(T_B^*) \leq w(\check{S}) \leq w(\overrightarrow{P^*}) + w(\overleftarrow{P^*}). \tag{3.2.17}$$

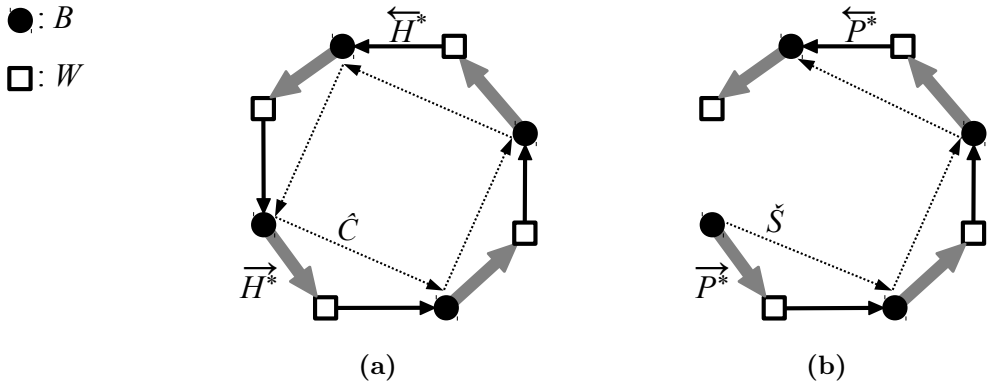

Figure 3.1. (a) A minimum cost alternating Hamiltonian cycle H^* of G. The subsets of edges $\overrightarrow{H^*}$ (bold gray arrows) and $\overleftarrow{H^*}$ (slender black arrows) form two disjoint perfect bipartite matchings. The shortcut $\hat{C}$ on $G[B]$ is given in dashed lines; (b) A minimum cost alternating Hamiltonian path P^* of G, where the set of edges $\overrightarrow{P^*}$ is given in bold gray arrows, $\overleftarrow{P^*}$ in slender black arrows, and the shortcut path $\check{S}$ on $G[B]$ is given in dashed lines.

3.2.2 Further Observations

We would like to bring to attention an observation with respect to the structures presented above, alternating spanning trees and perfect bipartite matchings. Let M^* and T^* be a minimum weight perfect bipartite matching and a minimum weight alternating spanning tree in a given bipartite graph G, respectively. Owing to its special structure any alternating spanning tree in G contains a unique perfect bipartite matching (Chapter 2, Lemma 2.4). Therefore, let $T^M \subseteq E(T^*)$ denote the edge set forming the perfect bipartite matching, and $T^\top$ the remaining edges of the alternating tree, i.e., $T^\top = E(T^*) \backslash T^M$. It simply holds

$$w(T^*) = w(T^M) + w(T^\top). \tag{3.2.18}$$

We present our view of the structure of an optimal alternating Hamiltonian path H^*, with $L(H^*) = \beta w(\overrightarrow{H^*}) + w(\overleftarrow{H^*})$, and an optimal alternating Hamiltonian path P^*, with $L(P^*) = \beta w(\overrightarrow{P^*}) + w(\overleftarrow{P^*})$, (see Eqs. (3.2.4), and (3.2.5), respectively). We introduce the parameters $\lambda, \kappa \in \mathbb{R}_+$ as

$$\lambda = \frac{w(\overleftarrow{H^*})}{w(\overrightarrow{H^*})}, \quad 1 \leq \lambda < \infty, \tag{3.2.19}$$

$$\kappa = \frac{w(\overleftarrow{P^*})}{w(\overrightarrow{P^*})}, \quad 0 \leq \kappa < \infty. \tag{3.2.20}$$

Then, for the cost of H^* we can write

$$L(H^*) = (\beta + \lambda) w(\overrightarrow{H^*}), \tag{3.2.21}$$

and for the cost of P^*

$$L(P^*) = (\beta + \kappa)w(\overrightarrow{P^*}). \tag{3.2.22}$$

For a given instance of the BBTSP and BBTSP-path, respectively, the value of the parameters λ and κ cannot be known without solving the instance exactly. However, for the purpose of our exposition, it suffices that $\lambda, \kappa \in \mathbb{R}_+$.

3.3 Previously Reported Heuristic Procedures for Building Alternating Hamiltonian Cycles

We digress for a moment to review some of the previously reported heuristics for the BTSP, and their behavior with respect to the bias factor $\beta \geq 1$.

Perhaps the most intuitive heuristic for the BTSP would be the *SWAP* heuristic, given in Algorithms 2 and 3. It has been reported that the *SWAP*-heuristic has an approximation of $1 + \zeta$ for the unbiased case [4, 10], and we confirm that this approximation factor remains valid for any $\beta \geq 1$. The correctness and validity of the $SWAP_\zeta$ procedure is argued in more detail in, e.g., [4, 10, 54].

Algorithm 2 Heuristic procedure $SWAP_\zeta$

Input: A complete, balanced bipartite graph $G = (B, W; E)$, and a symmetric edge weight function $w : E \to \mathbb{R}$ satisfying the quadrangle inequality.

Output: An alternating Hamiltonian cycle H_{SWAP_ζ}.

1: Find a minimum cost perfect bipartite matching M^* in $G = (B, W; E)$;
2: Build a ζ-approximate Hamiltonian cycle C' in $G[B]$;
3: Make an Eulerian multigraph $\mathcal{E}_{SWAP_\zeta} = (V(\mathcal{E}_{SWAP_\zeta}), E(\mathcal{E}_{SWAP_\zeta}))$, where $V(\mathcal{E}_{SWAP_\zeta}) = V(G)$ and $E(\mathcal{E}_{SWAP_\zeta}) = E(C') \uplus 2 \cdot M^*$;
4: Appropriately shortcut an Eulerian walk in $\mathcal{E}_{SWAP_\zeta}$ to get an alternating Hamiltonian cycle H_{SWAP_ζ} in G, preserving one copy of M^*;
5: **return** H_{SWAP_ζ}.

Lemma 3.6. *For a given instance (G, w, β) of the BBTSP, let H^* be an alternating Hamiltonian cycle in the bipartite graph G minimizing $L(H^*)$. Let $\overrightarrow{H^*} \subseteq E(H^*)$ be the set of edges traversed in the direction from the vertex set B to the vertex set W and let λ ($\lambda \geq 1$) be some real value which parameterizes $L(H^*)$ as $L(H^*) = (\beta + \lambda)w(\overrightarrow{H^*})$. Then, for the alternating Hamiltonian cycle H_{SWAP_ζ} computed by the $SWAP_\zeta$ procedure it holds*

$$L(H_{SWAP_\zeta}) \leq \left(1 + \frac{1 + \zeta + (\zeta - 1)\lambda}{\beta + \lambda}\right) L(H^*). \tag{3.3.23}$$

Proof. Following the construction steps of Algorithm 2 we get that,

$$\begin{aligned} L(H_{SWAP_\zeta}) \leq & \zeta w(C^*) + (\beta+1)w(M^*) \\ \leq & \zeta(w(\overrightarrow{H^*}) + w(\overleftarrow{H^*})) + (\beta+1)w(\overrightarrow{H^*}) \\ = & \zeta(1+\lambda)w(\overrightarrow{H^*}) + (\beta+1)w(\overrightarrow{H^*}) \\ = & (\beta+\zeta\lambda+\zeta+1)w(\overrightarrow{H^*}) \\ = & \left(\frac{\beta+\zeta\lambda+\zeta+1}{\lambda+\beta}\right) L(H^*). \end{aligned} \tag{3.3.24}$$

Note that the above result implies that for any $\beta \geq 1$ and $\lambda \geq 1$, it holds

$$L(H_{SWAP_\zeta}) \leq (1+\zeta)L(H^*). \tag{3.3.25}$$

□

Algorithm 3 Heuristic procedure $SWAP_2$

Input: A complete, balanced bipartite graph $G = (B, W; E)$, and a symmetric edge weight function $w : E \to \mathbb{R}$ satisfying the quadrangle inequality.

Output: An alternating Hamiltonian cycle H_{SWAP_2}.

1: Find a minimum cost perfect matching M^* in $G = (B, W; E)$;
2: Build a 2-approximate Hamiltonian cycle C' in $G[B]$;
3: {Make use of the double spanning tree approximation [61, 62]}
4: Make an Eulerian multigraph $\mathcal{E}_{SWAP_2} = (V(\mathcal{E}_{SWAP_2}), E(\mathcal{E}_{SWAP_2}))$, where $V(\mathcal{E}_{SWAP_2}) = V(G)$ and $E(\mathcal{E}_{SWAP_\zeta}) = E(C') \uplus 2 \cdot M^*$;
5: Appropriately shortcut an Eulerian walk in $\mathcal{E}_{SWAP_2}$ to get an alternating Hamiltonian cycle H_{SWAP_2} in G, preserving one copy of M^*;
6: **return** H_{SWAP_2}.

We choose to explicitly state the procedure adopting $\zeta = 2$, or the $SWAP_2$ procedure, given as Algorithm 3. The main reason lies in line 3, the way we choose to build C'. Namely, many procedures for building a ζ-approximate shortcut cycle C' (refer to Figure 3.1 (a)) do not have an explicit relationship with $\check{S}$ (Figure 3.1 (b)). However, the choice of using the *double around the tree* heuristic ([61, 62, 64]) enables us to state the following useful claim.

Lemma 3.7. *For a given instance (G, w, β) of the BBTSP, let H^* be an alternating Hamiltonian cycle in the bipartite graph G minimizing $L(H^*)$ and let P^* be an alternating Hamiltonian path in G minimizing $L(P^*)$. Let $\overrightarrow{H^*} \subseteq E(H^*)$ be the set of edges traversed in the direction from the vertex set B to the vertex*

set W in the alternating Hamiltonian cycle H^*. Analogously, let $\overrightarrow{P^*} \subseteq E(P^*)$ be the set of edges traversed in the direction from the vertex set B to the vertex set W in the alternating Hamiltonian path P^*. Consequently, for some real values λ ($\lambda \geq 1$) and κ ($\kappa \geq 0$) the optimal values $L(H^*)$ and $L(P^*)$ can be parameterized as $L(H^*) = (\beta + \lambda)w(\overrightarrow{H^*})$, respectively, $L(P^*) = (\beta + \kappa)w(\overrightarrow{P^*})$. Then, for the alternating Hamiltonian cycle H_{SWAP_2} computed by the $SWAP_2$ procedure it holds

$$L(H_{SWAP_2}) \leq \left(1 + \frac{3+\lambda}{\beta+\lambda}\right) L(H^*), \tag{3.3.26}$$

$$L(H_{SWAP_2}) \leq \left(1 + \frac{3+\kappa}{\beta+\kappa}\right) L(P^*). \tag{3.3.27}$$

Proof. The parameters λ ($\lambda \geq 1$) and κ ($\kappa \geq 0$) parameterize the optimal solutions H^* and P^* as in Eqs. (3.2.19) and (3.2.20), respectively. Comparing to the $SWAP_\zeta$ procedure, the only difference with $SWAP_2$ is the way of approximating a Hamiltonian cycle C' over the complete graph $G[B]$. The twice around the tree heuristic builds a minimum weight spanning tree T_B^* in the complete graph $G[B]$, and doubles every edge in T_B^* to get an Eulerian graph from which C' can be recovered (for more details, see [61, 62, 64]). Since it holds that $w(T_B^*) \leq S^*$ for a shortest Hamiltonian path S^* in $G[B]$ (Figure 3.1 (b)), we have

$$w(C') \leq 2w(T_B^*) \leq 2w(S^*), \tag{3.3.28}$$

$$w(C') \leq 2w(T_B^*) \leq 2w(C^*). \tag{3.3.29}$$

Nevertheless, it holds

$$w(C') \leq 2(w(\overrightarrow{H^*}) + w(\overleftarrow{H^*})), \tag{3.3.30}$$

$$w(C') \leq 2(w(\overrightarrow{P^*}) + w(\overleftarrow{P^*})). \tag{3.3.31}$$

The remainder of the analysis is the same as for the $SWAP_\zeta$ procedure. We have

$$L(H_{SWAP_2}) \leq w(C') + (\beta + 1)w(M^*). \tag{3.3.32}$$

From where, substituting Eqs. (3.2.6) and (3.3.30) we get

$$\begin{aligned} L(H_{SWAP_2}) &\leq 2(w(\overrightarrow{H^*}) + w(\overleftarrow{H^*})) + (\beta + 1)w(M^*) \\ &\leq (\beta + 2\lambda + 3)w(\overrightarrow{H^*}), \end{aligned} \tag{3.3.33}$$

which divided by the λ-parameterized value from Eq. (3.2.19) becomes

$$\begin{aligned}\frac{L(H_{SWAP_2})}{L(H^*)} \leq & \frac{(\beta+2\lambda+3)w(\overrightarrow{H^*})}{(\beta+\lambda)w(\overrightarrow{H^*})} \\ = & 1+\frac{3+\lambda}{\beta+\lambda}\end{aligned} \tag{3.3.34}$$

and the fact that the parameter $\lambda \geq 1$ leads to

$$\frac{L(H_{SWAP_2})}{L(H^*)} \leq 3 \implies L(H_{SWAP_2}) \leq 3L(H^*). \tag{3.3.35}$$

The analysis for the expression from Eq. (3.3.27) evolves identically, except that the optimal value $L(P^*)$ is parameterized by a parameter $\kappa \geq 0$, and therefore

$$\frac{L(H_{SWAP_2})}{L(P^*)} \leq 1+\frac{3+\kappa}{\beta+\kappa} \implies L(H_{SWAP_2}) \leq 4L(P^*). \tag{3.3.36}$$

□

But, having such a simple heuristic as doubling a spanning tree prompts the question why should we settle for anything less in the bipartite case. For the unbiased version ($\beta = 1$), the first improvement over the $SWAP_\zeta$ procedure comes by the virtue of alternating spanning trees, as a factor 2-approximation exactly in a double around the tree fashion [14, 21]. For the sake of completeness, we exhibit the approach relying on alternating spanning trees as Algorithm 4.

Algorithm 4 Heuristic procedure *TREE*

Input: A complete, balanced bipartite graph $G = (B, W; E)$, and a symmetric edge weight function $w : E \to \mathbb{R}$ satisfying the quadrangle inequality.

Output: An alternating Hamiltonian cycle H_{TREE}.

1: Find a minimum weight alternating spanning tree T^* in $G = (B, W; E)$;
2: Let T^* be rooted at the unique $u^* \in B$, with $d_{T^*}(u^*) = 1$;
3: Starting from u^* perform a depth-first traversal, skipping visited vertices;
4: Let H_{TREE} be the result of the depth-first traversal;
5: **return** H_{TREE}.

The correctness of the *TREE* procedure in the sense that it always returns an alternating Hamiltonian path H_{TREE} has been argued extensively [13, 14, 21, 30, 60], as has also the following claim.

Lemma 3.8. *Given an instance $(G, w, \beta = 1)$ of the BBTSP, let H_{TREE} be the result of the TREE procedure. Let H^* be an alternating Hamiltonian cycle and P^**

be an alternating Hamiltonian path, minimizing $L(H^*)$ and $L(P^*)$, respectively, for the given instance. Then it holds

$$L(H_{TREE}) \leq 2L(H^*), \tag{3.3.37}$$

$$L(H_{TREE}) \leq 2L(P^*). \tag{3.3.38}$$

However, with the introduction of the bias, the above claim becomes

Lemma 3.9. *Given an instance (G, w, β) of the BBTSP ($\beta \geq 1$), let H_{TREE} be the result of the TREE procedure. Let H^* be an alternating Hamiltonian cycle and P^* be an alternating Hamiltonian path, minimizing $L(H^*)$ and $L(P^*)$, respectively, for the given instance. Then it holds*

$$L(H_{TREE}) \leq (\beta + 1)L(H^*), \tag{3.3.39}$$

$$L(H_{TREE}) \leq (\beta + 1)L(P^*). \tag{3.3.40}$$

Noticing that there is an obvious trade-off between expressions given with Eq. (3.3.23) and Eq. (3.3.39) over the range $1 \leq \beta \leq \zeta$, Shurbevski et al. [54] proposed adopting the concept of a composite heuristic. The procedure is described as Algorithm 5. Note, we follow the terminology of Langston [38] when using the term composite heuristic. Similar exploitation of different approaches in order to derive an improved approximation guarantee has been described for similar routing problems [11, 22, 28].

Algorithm 5 Heuristic procedure *QCOMP*

Input: A complete, balanced bipartite graph $G = (B, W; E)$, and a symmetric edge weight function $w : E \to \mathbb{R}$ satisfying the quadrangle inequality.

Output: An alternating Hamiltonian cycle H_{QCOMP}.

1: Compute H_{SWAP_ζ} by the $SWAP_\zeta$ procedure;
2: Compute H_{TREE} by the *TREE* procedure;
3: Set $H_{QCOMP} \leftarrow \text{argmin}\{L(H_{SWAP_\zeta}), L(H_{TREE})\}$;
4: **return** H_{QCOMP}.

Lemma 3.10. *Given an instance (G, w, β) of the BBTSP ($\beta \geq 1$), let H_{QCOMP} be the result of the QCOMP procedure, and let H^* be an optimal solution to the given instance, minimizing $L(H^*)$. Then,*

$$L(H_{QCOMP}) \leq \frac{(\beta + 1)^2}{\beta(\beta + 1) - \zeta(\beta - 1)} L(H^*). \tag{3.3.41}$$

Proof. The proof to the above claim follows by equating the right-hand side expressions from Eq. (3.3.24) and

$$L(H_{TREE}) \leq \frac{(\beta+1)(\lambda+1)}{\beta+\lambda} L(H^*),$$

in the sense that

$$L(H_{QCOMP}) \leq L(H^*) \cdot \min\left\{ \frac{(\beta+1)(\lambda+1)}{\beta+\lambda}, 1 + \frac{\zeta+1+(\zeta-1)\lambda}{\beta+\lambda} \right\}. \quad (3.3.42)$$

Equality of the above expressions is achieved for $\lambda = \frac{\zeta}{\beta+1-\zeta}$. □

Note, the ratio of 16/7 reported in [52, 54] had been achieved for $\zeta = 2$, $\beta = 5/3$.

3.4 An Improved Heuristic Procedure for Building Alternating Hamiltonian Cycles

In this section we present a procedure for building an alternating Hamiltonian cycle in a given complete, balanced bipartite graph $G = (B, W; E)$. We show that if the graph G is endowed with a positive symmetric edge weight function w which satisfies the quadrangle inequality, this procedure can be used as an approximation algorithm for the BBTSP. The procedure for building an alternating Hamiltonian cycle does not rely on approximating the metric TSP.

3.4.1 Construction

Let $G = (B, W; E)$, be a complete, balanced bipartite graph and let $n := |B|(= |W|)$. Let $w : E(G) \to \mathbb{R}_+$ be a symmetric edge weight function satisfying the quadrangle inequality. Let M^* and T^* be a perfect matching and an alternating spanning tree in G of minimum $w(M^*)$ and $w(T^*)$, respectively.

We bring to attention the union of M^* and T^*. As observed in Section 2.5.3, Lemma 2.4, and repeated in Section 3.2.2, the alternating spanning tree T^* contains a unique perfect matching, T^M. We observe the union of T^M and M^*. By the definition of (perfect) matchings (Section 2.3), each vertex in $V(G) = B \cup W$ is incident with exactly one edge in either of T^M and M^*. Consequently, each vertex in $V(G)$ is incident with exactly two edges in $T^M \cup M^*$. Straightforwardly it can be deduced that $(V(G), T^M \cup M^*)$ forms a cycle cover of G. Let there be $k \leq n$ individual cycles, which we will denote by $\mathcal{R} := \{R_i : i = 1, 2, \ldots, k\}$. We can think of elements of $\mathcal{R}$ as nodes, and define a graph $G_{\mathcal{R}} = (V(G_{\mathcal{R}}), E(G_{\mathcal{R}}))$,

where $V(G_{\mathcal{R}}) = \mathcal{R}$. For brevity, for a subset E' of $E(G)$, we will use E' in terms of $E(G_{\mathcal{R}})$ to denote that

$$E(G_{\mathcal{R}}) = \left\{\{i,j\} : \exists\{u,v\} \in E',\, u \in V(R_i) \wedge v \in V(R_j)\right\}, \quad 1 \leq i,j \leq k. \tag{3.4.43}$$

Since T^* is an alternating spanning tree, which means that all vertices in $V(G)$ are connected, therefore the individual cycles R_i must be connected with each other as well, i.e., the graph $G_{\mathcal{R}} = (\mathcal{R}, T^{\top})$ is connected. We can choose an inclusion wise minimal $T^{\perp} \subseteq T^{\top}$, such that the graph $T_{\mathcal{R}} = (\mathcal{R}, T^{\perp})$ remains connected, i.e., $T_{\mathcal{R}}$ is a spanning tree of $G_{\mathcal{R}}$, as in Figure 3.2 (a). The multigraph $\mathcal{E}_{2APX}$ over the vertex set $V(G) = B \cup W$ in Figure 3.2 (b), has as its edge set a multiset sum of M^*, T^M and two copies of $T^{\perp}$. We need to show that this structure can be used to obtain a valid alternating cycle. As a first step, we will elaborate that there is an Eulerian walk.

Lemma 3.11. *The multigraph $\mathcal{E}_{2APX}$ is Eulerian.*

Proof. We need to show that $\mathcal{E}_{2APX}$ is connected, and every vertex has even degree with respect to $\mathcal{E}_{2APX}$. Connectedness follows from the fact that we sought the structure $T_{\mathcal{R}} = (\mathcal{R}, T^{\perp})$ to be a spanning tree, where $\mathcal{R}$ is a cycle cover of the vertex set $V(G) = B \cup W$. Every vertex in $V(G)$ is of degree 2 with respect to the cycle cover $\mathcal{R}$. Finally, we have added two copies of $T^{\perp}$, and from there the claim follows. □

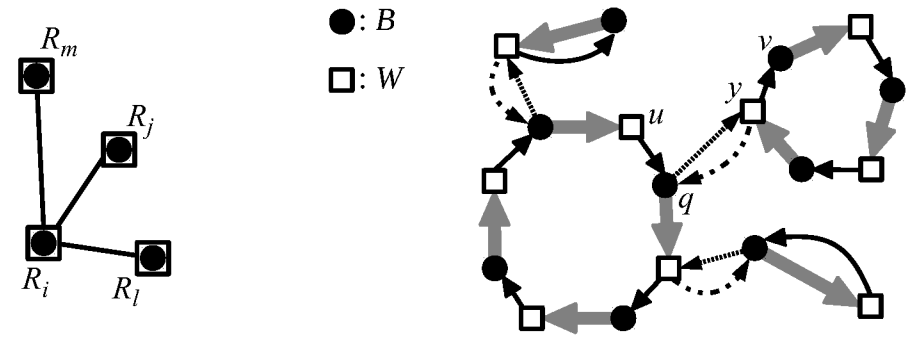

Figure 3.2. (a) A representation of $T_{\mathcal{R}} = (\mathcal{R}, T^{\perp})$. Nodes of $\mathcal{R}$ (■) are individual cycles over $V(G) = B \cup W$; (b) The resulting multigraph $\mathcal{E}_{2APX}$, arrows added to aid the image of traversing. The perfect matching M^* is given in bold gray lines, T^M in slender black, and the two copies of $T^{\perp}$ in dashed lines.

Next we show how $\mathcal{E}_{2APX}$ can be shortcut to give an alternating Hamiltonian cycle.

Lemma 3.12. *The Eulerian graph $\mathcal{E}_{2APX}$ can always be shortcut to an alternating Hamiltonian cycle H_{2APX}, preserving the edges from M^*.*

Proof. We will prove this claim by induction over the number k of cycles in the cycle cover $\mathcal{R}$.

Case $k = 1$: Trivial, this is H_{2APX};

Case $k > 1$: Start from the observation that $T^{\perp}$ is bipartite. Therefore there must exist a certain $q \in B$ joined to some $y \in W$ by an edge $\{q, y\} \in T^{\perp}$. Let $q \in R_i$ and $y \in R_j$. Now, let $u \in W$ (also $u \in R_i$) such that $\{u, q\} \in T^M$, and let $v \in B$ (also $v \in R_j$), such that $\{y, v\} \in T^M$ (Figure 3.2 (b)). We shortcut $\{\{u, q\}, \{q, y\}, \{y, v\}\}$ by $\{u, v\}$, thus merging the two cycles R_i and R_j and decreasing the number of cycles by one.

Note, all of the shortcut edges, $\{u, q\}$, $\{q, y\}$ and $\{y, v\}$ belong to T (either in $T^{\perp} \subseteq T^{\top}$ or T^M). Thus, edges in M^* are preserved and unmodified. Lastly, due to the quadrangle inequality from Eq. (2.3.5), this shortcutting will not increase the total weight $w(\mathcal{E}_{2APX})$. □

Algorithm 6 Heuristic procedure *2APX*

Input: A complete, balanced bipartite graph $G = (B, W; E)$, and a symmetric edge weight function $w : E \to \mathbb{R}$ satisfying the quadrangle inequality.

Output: An alternating Hamiltonian cycle H_{2APX}.

1: Compute a minimum weight perfect matching M^* and a minimum weight alternating spanning tree T^* in G;
2: Let $\mathcal{R} := \{R_i : i = 1, 2, \ldots, k\}$ be the cycle cover of G given by $(V(G), M^* \cup T^M$;
3: Choose an inclusion-wise minimal $T^{\perp} \subseteq T^{\top}$ such that $T_{\mathcal{R}} = (\mathcal{R}, T^{\perp})$ is a spanning tree;
4: Construct a multigraph $\mathcal{E}_{2APX} = (V(\mathcal{E}_{2APX}), E(\mathcal{E}_{2APX}))$, where $V(\mathcal{E}_{2APX}) = V(G)$, and $E(\mathcal{E}_{2APX}) = M^* \uplus T^M \uplus 2 \cdot T^{\perp}$ (Figure 3.2 (b));
5: Shortcut an Eulerian walk of $\mathcal{E}_{2APX}$ to an alternating Hamiltonian cycle H_{2APX}, preserving the edges from M^*;
6: **return** H_{2APX}.

We term the procedure for constructing alternating Hamiltonian cycles, which we have exhibited so far, as *2APX*. A concise summary of the construction procedure *2APX* is given as Algorithm 6.

An approximate alternating Hamiltonian path P_{2APX} can be obtained by discarding any arc of H_{2APX} not in M^*. The discarded arc does not need to

have the greatest weight among all arcs in $H_{2APX} \backslash M^*$. The procedure that utilizes the construction process of the procedure *2APX* to obtain an alternating Hamiltonian path is termed *2APX-path*, and summarized as Algorithm 7.

Algorithm 7 Heuristic procedure *2APX-path*

Input: Same as the *2APX* procedure.
Output: An alternating Hamiltonian path P_{2APX}.
1: Run the *2APX* procedure to get H_{2APX};
2: Choose any edge $\{u, v\} \notin M^*$ from H_{2APX};
3: Set $P_{2APX} \leftarrow H_{2APX} \backslash \{u, v\}$;
4: **return** P_{2APX}.

3.4.2 Approximation Ratio

Next, we investigate the applicability of the *2APX* procedure as an approximation algorithm.

Theorem 3.1. *For a given instance (G, w, β) of the metric BBTSP, let H^* be an alternating Hamiltonian cycle of minimal cost $L(H^*)$. Let $\overrightarrow{H^*} \subseteq E(H^*)$ be the edge set that is traversed in the direction from B to W. Consequently, the value $L(H^*)$ can be parameterized by some $\lambda \in \mathbb{R}_+$, $\lambda \geq 1$, as $L(H^*) = (\beta + \lambda) w(\overrightarrow{H^*})$. Then, for H_{2APX} as the result from the 2APX procedure it holds*

$$L(H_{2APX}) \leq \left(1 + \frac{1+\lambda}{\beta+\lambda}\right) L(H^*). \tag{3.4.44}$$

Proof. In order to derive an upper bound of the cost $L(H_{2APX})$, we will retrace the steps from the construction process, and recall some of the bounds presented in Section 3.2, especially Section 3.2.2.

First, recall that we chose a $T^\perp \subseteq T^\top$, implying that $w(T^\perp) \leq w(T^\top)$. It readily follows (see Eq. (3.2.18))

$$w(T^\perp) \leq w(T^*) - w(T^M). \tag{3.4.45}$$

Let us partition $E(H_{2APX})$ into two disjoint matchings, $\overrightarrow{H}_{2APX}$ and $\overleftarrow{H}_{2APX}$, in such a way that $\overrightarrow{H}_{2APX} = M^*$ and $\overleftarrow{H}_{2APX}$ is a shortcut through $T^M \uplus 2 \cdot T^\perp$, as in Lemma 3.12. We choose a traversal orientation such that exactly the edges of $\overrightarrow{H}_{2APX}$ are traversed in the direction from B to W. From the bias factor β of

Eqs. (2.4.17), (2.4.19) and (3.2.4), we have

$$\begin{aligned} L(H_{2APX}) &= \beta w(\overrightarrow{H}_{2APX}) + w(\overleftarrow{H}_{2APX}) \\ &\leq \beta w(M^*) + w(T^M) + 2w(T^{\perp}). \end{aligned} \tag{3.4.46}$$

Recall the partition of a minimum cost alternating spanning tree from Eq. (3.2.18) and the related bound from Eq. (3.4.45) and substitute them in Eq. (3.4.46). From this, and the fact that $w(M^*) \leq w(T^M)$, we get

$$\begin{aligned} L(H_{2APX}) &\leq \beta w(M^*) + 2w(T^*) - w(T^M) \\ &\leq 2w(T^*) + (\beta - 1)w(M^*). \end{aligned} \tag{3.4.47}$$

Next we substitute for M^* and T^* the bounds given with Eqs. (3.2.6) and (3.2.8) to obtain

$$\begin{aligned} L(H_{2APX}) \leq & 2(w(\overrightarrow{H^*}) + w(\overleftarrow{H^*})) + (\beta - 1)w(\overrightarrow{H^*}) \\ = & 2(1 + \lambda)w(\overrightarrow{H^*}) + (\beta - 1)w(\overrightarrow{H^*}) \\ = & (\beta + 2\lambda + 1)w(\overrightarrow{H^*}). \end{aligned} \tag{3.4.48}$$

Finally, following Eq. (3.2.21), the expression above can be stated as

$$L(H_{2APX}) \leq \frac{\beta + 2\lambda + 1}{\beta + \lambda} L(H^*), \tag{3.4.49}$$

which leads to the claim. □

Theorem 3.1 gives the result announced in the Introduction, Eq. (3.1.1)

$$\frac{L(H_{2APX})}{L(H^*)} \leq 1 + \frac{1 + \lambda}{\beta + \lambda}.$$

The result from Theorem 3.1 and the definition of an approximation ratio of Eq. (2.1.1) give the following result

$$\alpha_{2APX} \leq 2,$$

which holds true for any $\beta \geq 1$ and $\lambda \in \mathbb{R}_+$. However, Eq. (3.1.1) provides us with further insight of the behavior of $L(H_{2APX})$ for increasing values of β, and some fixed value of λ.

The case of alternating Hamiltonian paths has an essentially identical analysis and therefore we shall just state it as a result, omitting the formal proof, as it largely follows the proof of Theorem 3.1.

Theorem 3.2. *For a given instance (G, w, β) of the metric BBTSP-path, let P^* be an alternating Hamiltonian path of minimal cost $L(P^*)$. Let the edge set $\overrightarrow{P^*} \subseteq E(P^*)$ be traversed in the direction from B to W, so that the value $L(P^*)$ is parameterized by some $\kappa \in \mathbb{R}_+$, $\kappa \geq 0$, as $L(P^*) = (\beta + \kappa)w(\overrightarrow{P^*})$. Then, for H_{2APX} as the result from the 2APX procedure it holds*

$$L(H_{2APX}) \leq \left(1 + \frac{1+\kappa}{\beta+\kappa}\right) L(P^*). \tag{3.4.50}$$

3.4.3 As Part of a Composite Heuristic

As the previously reported approximation ratio of 16/7 described in [54] relies on a composite heuristic, i.e., on a trade-off between two different procedures for building an alternating Hamiltonian path, we investigate a similar approach. For that purpose, we make use of the $SWAP_\zeta$ procedure, which was introduced and analyzed in Section 3.3. The $SWAP_\zeta$ procedure has been described as a heuristic method for the swapping problem [4], and adopted to the bipartite TSP [10]. Briefly described, it is given as Algorithm 2.

For the purpose of arriving to a suitable expression for a composite heuristic relying on the *2APX* and $SWAP_\zeta$ procedures, we revise the bounds on $L(H_{SWAP_\zeta})$. As a reminder, let C^* be an optimal Hamiltonian cycle on the complete graph induced by one of the vertex partitions, taken to be $G[B]$ without loss of generality. Analogous to Eq. (3.2.12), for a ζ-approximate C' of an optimal C^* we get

$$w(C') \leq \zeta w(C^*) \leq \zeta \left(w(\overrightarrow{H^*}) + w(\overleftarrow{H^*}) \right). \tag{3.4.51}$$

Since we can shortcut an Eulerian walk in $\mathcal{E}_{SWAP_\zeta}$ to obtain H_{SWAP_ζ} in such a way that one copy of M^* is preserved, we can orient the traversal of H_{SWAP_ζ} so that exactly the edges in M^* are traversed in the direction from B to W. Following Eqs. (3.2.6), (3.2.21) and (3.4.51)

$$\begin{aligned} L(H_{SWAP_\zeta}) \leq & \zeta(1+\lambda) w(\overrightarrow{H^*}) + (\beta+1) w(M^*) \\ \leq & \frac{\zeta(1+\lambda) + \beta + 1}{\beta + \lambda} L(H^*). \end{aligned} \tag{3.4.52}$$

Since from Lemma 2.1 we have that the extension of w over the edges of $G[B]$ is symmetric and satisfies the triangle inequality, we can use, e.g., Christofides' heuristic [16] to build a C' with $\zeta = 3/2$.

We propose a simple procedure which will compute both H_{2APX} and H_{SWAP_ζ} according to their respective construction procedures, and choose the one of lower

cost. Let us term this procedure $COMP$ and the resulting alternating Hamiltonian cycle H_{COMP}. From Eqs. (3.4.44) and (3.4.52) we get

$$\begin{aligned} L(H_{COMP}) &\leq \min\left\{\frac{\beta+2\lambda+1}{\beta+\lambda}L(H^*), \frac{\zeta(1+\lambda)+\beta+1}{\beta+\lambda}L(H^*)\right\} \\ &\leq \left(1+\frac{2}{\zeta+\beta(2-\zeta)}\right)L(H^*). \end{aligned} \tag{3.4.53}$$

The trade-off in Eq. (3.1.2) is achieved for $\lambda = \frac{\zeta}{2-\zeta}$ and therefore it only makes sense to be called when $\zeta < 2$.

Corollary 3.1. *The metric BBTSP can be approximated within ratio*

$$\alpha_{COMP} = 1 + \frac{2}{\zeta+\beta(2-\zeta)}, \tag{3.4.54}$$

where ζ is an approximation ratio for the metric TSP over the same metric.

As an added benefit, we have arrived to the expression of Eq. (3.4.54), which is not dependent on a hidden instance-specific parameter, such as λ.

The above conclusion does generalize to the case of alternating Hamiltonian paths as well. The same arguments concerning the relations between the Hamiltonian cycles $\hat{C}$, C^* and C' in the graph $G[B]$, can be employed for the relations between the Hamiltonian paths in the induced graph $G[B]$: $\check{S}$, S^* and an approximate Hamiltonian path S'. With a slight modification, the well known Christofides' heuristic [16] can be applied to build an approximate S', retaining $\zeta = 3/2$ [29]. Note that we have arbitrarily chosen to elaborate with regards to shortcut paths and cycles on the induced graph $G[B]$, but the same relations are valid with regards to the induced graph $G[W]$.

3.4.4 Computational Complexity

Without much deliberation we will state that all procedures undertaken to obtain an alternating Hamiltonian cycle have well known polynomial time implementations. An excellent source of information concerning the presented combinatorial structures as well as their algorithmic implementations can be found in [36, 50], as well as [20]. We will just state that even with the improved analysis of the application of a matroid intersection algorithm for computing a minimum weight alternating spanning trees (Chapter 2, Lemma 2.5), it still remains a bottleneck procedure and determines the time complexity of computing an approximate alternating Hamiltonian cycle. As a consequence, we can state the following

Theorem 3.3. *The biased bipartite traveling salesman problem with a symmetric edge weight function satisfying the quadrangle inequality and a bias $\beta \geq 1$ can be approximated within a factor $\alpha \leq 2$, in polynomial time complexity.*

3.5 Concluding Remarks

We examined the biased bipartite TSP (BBTSP) as a generalization of the symmetric bipartite TSP with quadrangle inequality by introducing a bias term $\beta \geq 1$, which introduces asymmetry in the cost of alternating Hamiltonian paths and cycles. This generalization had been introduced as a means to better capture some features of industrial material handling scenarios, but we hope it will find applicability in a much broader range of optimization problems.

We presented a heuristic for building alternating Hamiltonian paths and cycles in complete bipartite graphs. After analyzing the performance of the heuristic as an approximation algorithm, we obtained a constant factor 2 which is valid for both alternating Hamiltonian paths and alternating Hamiltonian cycles, and showed that this approximation ratio holds for any value of the bias $\beta \geq 1$. We also analyzed the performance of the proposed procedure for building alternating Hamiltonian cycles as part of a composite heuristic, and derived an approximation ratio which benefits from both a better approximation for the metric TSP and an increased value for the bias β.

It is a standing question whether the constant bound 2 of the approximation ratio presented in this chapter can be further improved by some algorithms similar to existing approaches for the standard metric TSP [62].

4 Repetitive Routing With a Fixed Processing Sequence

In the previous chapter we presented and analyzed approximation algorithms for the Biased Bipartite TSP and TSP-path problems. Next, we aim to apply the approximation algorithm solution to the practical problem which inspired this research. We will gradually work our way through the intricacies arising from adopting our theoretical solution to a specific domain. During this process, we will face several challenges and we will propose methods to efficiently resolve them.

4.1 Introduction

In this chapter we return to the Fixed-sequence Repetitive Routing Problem as it was described in Chapter 1. In order to retain generality, we confine our exposition to graph-theoretic results, but for the sake of continuity, we briefly revise the correspondence of the terms associated with the concrete RRP and general graph-theoretic terminology, as detailed in Section 2.3.1. In the context of the RRP, we observe a transition from configuration C_i to configuration C_j. We can set-up a BBTSP-path instance by setting $B := C_i$, $W := C_j$, and adopting the distance $d(u, v), u, v \in S$ over locations u and v in S as an edge weight function in the complete bipartite graph $G = (B, W; E)$. An alternating Hamiltonian path P in the bipartite graph G is exactly a partial transfer route $P_{i,j}$.

Let $H'_{i,j}$ denote an approximate alternating Hamiltonian cycle derived by the procedure *2APX* for the BBTSP instance defined by the bipartite graph with vertex sets C_i and C_j. Then, without loss of generality, $H'_{i,j}$ can be expressed as

$$\sigma'_i(1) \to \tau'_j(1) \to \sigma'_i(2) \to \tau'_j(2) \to \cdots \to \sigma'_i(n) \to \tau'_j(n) \to \sigma'_i(1).$$

For convenience we introduce a *shift* function for a permutation σ as

$$\sigma := \sigma \gg k \quad \Leftrightarrow \quad \sigma(l) := \sigma(l + k), \quad l = 1, 2, \ldots, n, \tag{4.1.1}$$

where the indexing device l is allowed to wrap around, Eq. (2.3.13).

The alternating Hamiltonian cycle $H'_{i,j}$ can be converted to an alternating Hamiltonian path $P'_{i,j}$ by simply discarding any edge $\tau'_j(k) \to \sigma'_i(k+1)$, $k \in [1, n]$,

where for convenience, the indexing device k is allowed to wrap around, as in Eq. (2.3.13). The discarded edge $\tau'_j(k) \to \sigma'_i(k+1)$ is traversed in the direction from vertex $\tau'_j(k) \in C_j$ to vertex $\sigma'_i(k+1) \in C_i$. After removing the edge $\tau'_j(k) \to \sigma'_i(k+1)$ from the alternating Hamiltonian cycle $H'_{i,j}$, we shift the permutations by k positions:

$$\sigma'_i := \sigma'_i \gg k \tag{4.1.2}$$

$$\tau'_j := \tau'_j \gg k. \tag{4.1.3}$$

In effect, we have actually relabeled the permutations σ'_i and τ'_j in such a way that the resulting alternating path $P'_{i,j}$ can be described as

$$\sigma'_i(1) \to \tau'_j(1) \to \sigma'_i(2) \to \tau'_j(2) \to \cdots \to \sigma'_i(n) \to \tau'_j(n).$$

As we have seen from Theorem 3.2, Eq. (3.4.50), we can use alternating Hamiltonian cycles $H'_{\ell-1,\ell}$, $\ell = 1, 2, \ldots, m$, resulting from the *2APX* procedure to obtain partial transfer routes $P'_{\ell-1,\ell}$, $\ell = 1, 2, \ldots, m$, for all transitions. The remaining question is how to choose which edge $\tau'_\ell(l) \to \sigma'_{\ell-1}(l+1)$ to discard from each of the alternating Hamiltonian paths $H'_{\ell-1,\ell}$ in order to extract the alternating Hamiltonian paths (partial transfer routes) $P'_{\ell-1,\ell}$, $\ell = 1, 2, \ldots, m$.

The consecutiveness constraint (Eq. (1.3.7)) gives us a convenient guideline for doing so. As a reminder, we assume that for reasons relevant to the factual Repetitive Routing Problem, we wish to fix a vertex $c_l^0 \in C_0$ as a starting point of the alternating Hamiltonian path $P_{0,1}$. Without loss of generality, assume that this vertex is c_1^0. Obviously, the edge $\tau_1(n) \to \sigma_0(1)$ is to be discarded, thereby the vertex $\tau_1(n) \in C_1$ is determined as the terminal of the first alternating Hamiltonian path, $P_{0,1}$. We proceed through all remaining transitions in the same fashion. For notational convenience, we make use of an artificial point $\tau_0(n) := c_1^0$. The simple procedure for obtaining a complete transfer route where each alternating Hamiltonian path satisfies the consecutiveness constraint is given as Algorithm 8.

As a final step, the m alternating Hamiltonian paths $P'_{\ell-1,\ell}$, $\ell = 1, 2, \ldots, m$ define a contiguous transfer route $\Pi' := (P'_{\ell-1,\ell} : \ell = 1, 2, \ldots, m)$. The concatenation procedure of Algorithm 8 obviously runs in polynomial time. (With some care, it can even be implemented to run in time $O(m)$, linear in the number of transitions.)

Similar to Eq. (3.2.20), we introduce a parameter characterizing the cost $L(\Pi^*)$ of an optimal transfer route $\Pi^* = (P_{\ell-1,\ell} : \ell = 1, 2, \ldots, m)$ as follows

$$K = \frac{\sum_{\ell=1}^{m} w(\overleftarrow{P}_{\ell-1,\ell})}{\sum_{\ell=1}^{m} w(\overrightarrow{P}_{\ell-1,\ell})}, \quad 0 \leq K < \infty, \tag{4.1.4}$$

Algorithm 8 A straightforward concatenation procedure

Input: m approximate alternating Hamiltonian cycles $H'_{\ell-1,\ell}$, $\ell = 1, 2, \ldots, m$ computed by the *2APX* procedure.

Output: An approximate transfer route $\Pi' = (P'_{\ell-1,\ell} : \ell = 1, 2, \ldots, m)$, where the m partial transfer routes $P'_{\ell-1,\ell}$ satisfy the consecutiveness constraint, Eq. (1.3.7).

1: Let $\tau_0(n) \leftarrow c_1^0$; {Assumed without loss of generality}
2: **for** $\ell \leftarrow 1$ to m **do**
3: Set $k \leftarrow \sigma'_{\ell-1}(\tau_{\ell-1}(n))^{-1}$;
4: $\sigma'_{\ell-1} \leftarrow \sigma'_{\ell-1} \gg (k-1)$;
5: $\tau'_{\ell} \leftarrow \tau'_{\ell} \gg (k-1)$; {Shift the permutations $\sigma'_{\ell-1}$ and τ'_{ℓ} by $k-1$ positions}
6: $P'_{\ell-1,\ell} \leftarrow \big(\sigma'_{\ell-1}(1) \to \tau'_{\ell}(1) \to \sigma'_{\ell-1}(2) \to \tau'_{\ell}(2) \to \cdots \to \sigma'_{\ell-1}(n) \to \tau'_{\ell}(n)\big)$
7: **end for**;
8: Set $\Pi' \leftarrow (P'_{\ell-1,\ell} : \ell = 1, 2, \ldots, m)$;
9: **return** Π'.

where $\overrightarrow{P}_{\ell-1,\ell}$ denotes the set of edges which are traversed in a direction influenced by the bias factor (see Eqs. (1.4.13), (2.4.18)) and $\overleftarrow{P}_{\ell-1,\ell}$ denotes the set of edges traversed in an unbiased direction, respectively, in each transition $\ell = 1, 2, \ldots, m$. Then, for the value of the cost $L(\Pi^*)$ we can write

$$L(\Pi^*) = (\beta + K) \sum_{\ell=1}^{m} w(\overrightarrow{P}_{\ell-1,\ell}). \tag{4.1.5}$$

Theorem 4.4. *Given an instance of the Fixed-sequence Repetitive Routing Problem, let a positive real parameter K characterize the value of an optimal solution $L(\Pi^*)$ as in Eq. (4.1.5). We can build a transfer route $\Pi' = (P'_{\ell-1,\ell} : \ell = 1, 2, \ldots, m)$ in polynomial time, such that*

$$L(\Pi') \leq \left(1 + \frac{1+K}{\beta+K}\right) L(\Pi^*).$$

Proof. We will make use of the result of Theorem 3.2 to obtain approximate alternating Hamiltonian cycles $H'_{\ell-1,\ell}$, for $\ell = 1, 2, \ldots, m$, and of Algorithm 8 to extract a feasible transfer route Π'. Let each alternating Hamiltonian path $P_{\ell-1,\ell}$, $\ell = 1, 2, \ldots, m$, in the optimal solution Π^* be parameterized by a positive real parameter $\kappa_{\ell-1,\ell}$ as $\widetilde{w}(P_{\ell-1,\ell}) = (\beta + \kappa_{\ell-1,\ell}) w(\overrightarrow{P}_{\ell-1,\ell})$, where $\kappa_{\ell-1,\ell}$ is defined as

$$\kappa_{\ell-1,\ell} = \frac{w(\overleftarrow{P}_{\ell-1,\ell})}{w(\overrightarrow{P}_{\ell-1,\ell})}, \quad 0 \leq \kappa_{\ell-1,\ell} < \infty, \quad \ell = 1, 2, \ldots, m,$$

and $\overrightarrow{P}_{\ell-1,\ell}$ and $\overleftarrow{P}_{\ell-1,\ell}$ are the sets of edges traversed in the direction from the vertex set $C_{\ell-1}$ to C_ℓ and in the direction from the vertex set C_ℓ to the vertex set $C_{\ell-1}$ respectively. As a consequence of Theorem 3.2, we have

$$\widetilde{w}(H'_{\ell-1,\ell}) \leq \left(1 + \frac{1+\kappa_{\ell-1,\ell}}{\beta+\kappa_{\ell-1,\ell}}\right) \widetilde{w}(P_{\ell-1,\ell}), \tag{4.1.6}$$

and as the cost function w is metric, thereby $w(\tau'_j(l) \to \sigma'_i(l+1)) \geq 0$,

$$\widetilde{w}(P'_{\ell-1,\ell}) \leq \left(1 + \frac{1+\kappa_{\ell-1,\ell}}{\beta+\kappa_{\ell-1,\ell}}\right) \widetilde{w}(P_{\ell-1,\ell}), \tag{4.1.7}$$

holds for any choice of an edge $\tau'_j(l) \to \sigma'_i(l+1)$ to be discarded. Therefore, we can write

$$\begin{aligned}
L(\Pi') =& \sum_{\ell=1}^{m} \widetilde{w}(P'_{\ell-1,\ell}) \\
\leq& \sum_{\ell=1}^{m} \left(1 + \frac{1+\kappa_{\ell-1,\ell}}{\beta+\kappa_{\ell-1,\ell}}\right) \widetilde{w}(P) \\
=& \sum_{\ell=1}^{m} (\beta + 2\kappa_{\ell-1,\ell} + 1)\, w(\overrightarrow{P}_{\ell-1,\ell}) \\
=& (\beta+1) \sum_{\ell=1}^{m} w(\overrightarrow{P}_{\ell-1,\ell}) + 2 \sum_{\ell=1}^{m} \kappa_{\ell-1,\ell} w(\overrightarrow{P}_{\ell-1,\ell}) \\
=& (\beta+1) \sum_{\ell=1}^{m} w(\overrightarrow{P}_{\ell-1,\ell}) + 2 \sum_{\ell=1}^{m} w(\overleftarrow{P}_{\ell-1,\ell}) \\
=& (\beta + 2K + 1) \sum_{\ell=1}^{m} w(\overrightarrow{P}_{\ell-1,\ell}) \\
=& \left(\frac{\beta+2K+1}{\beta+K}\right) \sum_{\ell=1}^{m} \widetilde{w}(P_{\ell-1,\ell}) \\
=& \left(\frac{\beta+2K+1}{\beta+K}\right) L(\Pi^*).
\end{aligned} \tag{4.1.8}$$

□

4.2 Optimal Concatenation by Dynamic Programming

We have just provided a polynomial time approximation approach for the Fixed-sequence RRP. However, we are able to do slightly better than the straightforward concatenation to satisfy the consecutiveness constraint. We examine two different approaches for the biased and the unbiased version or the RRP separately. In the two cases we rely on slightly different ideas to possibly achieve a better concatenation of given partial transfer routes.

4.2.1 Unbiased Case

In the case of $\beta = 1$, we are free to choose the traversal orientation of an alternating Hamiltonian cycle $H'_{i,j}$, without influencing the cost $w(H'_{i,j})$. Therefore, the traversal of $H'_{i,j}$ can take two forms

Clockwise: $\sigma_i^{\rightarrow}(1) \rightarrow \tau_j^{\rightarrow}(1) \rightarrow \sigma_i^{\rightarrow}(2) \rightarrow \tau_j^{\rightarrow}(2) \rightarrow \cdots \rightarrow \sigma_i^{\rightarrow}(n-1) \rightarrow \tau_j^{\rightarrow}(n) \rightarrow \sigma_i^{\rightarrow}(1)$;

Counterclockwise: $\sigma_i^{\leftarrow}(1) \rightarrow \tau_j^{\leftarrow}(1) \rightarrow \sigma_i^{\leftarrow}(2) \rightarrow \tau_j^{\leftarrow}(2) \rightarrow \cdots \rightarrow \sigma_i^{\leftarrow}(n-1) \rightarrow \tau_j^{\leftarrow}(n) \rightarrow \sigma_i^{\leftarrow}(1)$.

The counterclockwise traversal is simply an inverted version of the clockwise traversal, i.e., for $l = 1, 2, \ldots, n$

$$\sigma_i^{\leftarrow}(l) = \begin{cases} \sigma_i^{\rightarrow}(l), & l = 1, \\ \sigma_i^{\rightarrow}(n-l+2), & l \neq 1, \end{cases} \tag{4.2.9}$$

$$\tau_j^{\leftarrow}(l) = \tau_j^{\rightarrow}(n-l+1). \tag{4.2.10}$$

This feature, in spite of the consecutiveness constraint, allows for some freedom in choosing the way to concatenate the candidate partial routes. This was first noticed by Shurbevski et al. [53], where initial computational results are reported. Subsequently, further computational experiments were performed, and reported in [52, 54, 55].

Optimal concatenation is performed by a dynamic programming procedure, given as Algorithm 9. During the dynamic programming procedure, we need to keep track of partial solutions. Namely, the algorithm proceeds in stages $\ell = 1, 2, \ldots, m$, i.e., over the m transitions. While the permutations $\sigma_{\ell-1}^{\rightarrow}$ and $\tau_{\ell}^{\rightarrow}$ describing the traversal of an alternating Hamiltonian cycle $H'_{\ell-1,\ell}$ are immutable with regards to the shift operation, we need to have explicit bookkeeping for the case of alternating Hamiltonian paths. Therefore, for each stage ℓ of the dynamic programming procedure, we introduce the candidate solution permutations $\sigma_{\ell-1}^{j}$ and τ_{ℓ}^{j}, $j = 1, 2, \ldots, n$. We will use $w(\sigma_{\ell-1}^{j}, \tau_{\ell}^{j})$ to denote the cost of an alternating Hamiltonian path described by $\sigma_{\ell-1}^{j}$ and τ_{ℓ}^{j}. In addition, we introduce the value function $z_\ell(j)$, evaluating the fitness of the candidate solution represented by $\sigma_{\ell-1}^{j}$ and τ_{ℓ}^{j} at stage ℓ. For generality in our notation, we introduce n artificial variables $\tau_0^j(n)$ as

$$\tau_0^j(n) := c_j^0, \quad j = 1, 2, \ldots, n. \tag{4.2.11}$$

as well as initial values of the value function $z_\ell(j)$

$$z_\ell(j) = \begin{cases} 0, & \ell = 0,\ j = 1, \\ \infty, & \text{otherwise}, \end{cases} \qquad \ell = 0, 1, \ldots, m;\ j = 1, 2, \ldots, n. \tag{4.2.12}$$

The initial values that are set to the value function $z_0(j)$ as in Eq. (4.2.12) prohibitively punish solutions where the alternating Hamiltonian path $P_{0,1}$ does not start from location $c_1^0 \in C_0$. By setting all the $z_0(j) = 0$, we can possibly get an even better solution without any extra computational cost, provided we are granted the freedom to choose a starting location in this way. Finally, at each stage ℓ of the algorithm we will keep a *predecessor* reference, $pred_\ell(j)$ to facilitate backtracking at the end of the algorithm and extracting the optimal solution.

With some care, the dynamic programming procedure described as Algorithm 9 can be implemented to run in $O(mn)$ time ([52, 55]). It does not have influence on the theoretical approximation factor for the Fixed-sequence RRP, but might offer better results at little additional computational cost.

4.2.2 Biased Case

In the biased case, $\beta > 1$, the traversal orientation of an alternating Hamiltonian cycle $H'_{i,j}$ determined in the *2APX* procedure cannot be changed without influencing the biased cost $L(H'_{i,j})$. However, we might still exploit an interesting feature arising from the introduction of the bias, namely, we may relax the consecutiveness constraint. Relaxing the consecutiveness constraint is justifiable, because, as can be discerned from Figure 2.1 (a) and (b), modifying a route which violates the consecutiveness constraint calls for a change of direction in which some of the edges are traversed. This in turn, is not a freedom we can allow ourselves when a bias factor is concerned, for then the assumption of symmetry does not hold. Therefore, it is not inconceivable that optimal transfer routes where partial transfer routes violate the consecutiveness constraint might exist. Let $\widetilde{\Pi}$ be a transfer route with a relaxed consecutiveness constraint. Still, for $\widetilde{\Pi}$ to be contiguous, it should be of the form

$$\begin{aligned}\widetilde{\Pi} = &\overbrace{\sigma_0(1) \to \tau_1(1) \to \sigma_0(2) \to \cdots \to \sigma_0(n) \to \tau_1(n)}^{P_{0,1}} \xrightarrow[\text{edge}]{\text{Aligning}} \overbrace{\sigma_1(1) \to}^{P_{1,2}} \cdots \\ &\overbrace{\cdots \to \tau_{\ell-1}(n)}^{P_{\ell-2,\ell-1}} \xrightarrow[\text{edge}]{\text{Aligning}} \overbrace{\sigma_{\ell-1}(1) \to \tau_\ell(1) \to \cdots \to \sigma_{\ell-1}(n) \to \tau_\ell(n)}^{P_{\ell-1,\ell}},\end{aligned} \tag{4.2.13}$$

where we call the edges of the form $\tau_\ell(n) \to \sigma_\ell(1)$, $\ell = 1, 2, \ldots, m$, *aligning* edges, because they can be thought of as aligning alternating Hamiltonian paths $P_{\ell-1,\ell}$ and $P_{\ell,\ell+1}$ in the case where the consecutiveness constraint of Eq. (1.3.7) is violated, i.e., $\sigma_\ell(1) \neq \tau_\ell(n)$. Given an ordered tuple of alternating Hamiltonian paths and aligning edges, a transfer route $\widetilde{\Pi}$ with relaxed consecutiveness

constraint can be written as

$$\widetilde{\Pi} = (P_{0,1}; (\tau_1(n), \sigma_1(1)); P_{1,2}; (\tau_2(n), \sigma_2(1)); \ldots; (\tau_{n-1}(n), \sigma_{n-1}(1)); P_{n-1,n}). \tag{4.2.14}$$

Thus, given m approximate alternating Hamiltonian cycles $H'_{\ell-1,\ell}$ obtained by the *2APX* procedure, we aim to align them to form a feasible transfer route, as opposed to concatenating them. Of course, a concatenation can also be thought of as an alignment which satisfies the consecutiveness constraint.

We proceed by exhibiting a dynamic programing alignment procedure, similar to the concatenation procedure of Algorithm 9. For notational convenience, we introduce n additional variables, $\tau_0(j)$, $j = 1, 2, \ldots, n$ as

$$\tau_0(j) := c_1^0, \quad j = 1, 2, \ldots, n. \tag{4.2.15}$$

Note, if we were at liberty to choose a starting vertex from C_0, we can possibly set $\tau_0(j) := c_j^0$, $j = 1, 2, \ldots, n$, and gain a slight advantage at virtually no additional computational cost.

During the course of the alignment procedure, we make use of value functions $z_\ell(j)$, $\ell = 0, 1, \ldots, m$, $j = 1, 2, \ldots, n$ setting their initial values as in Eq. (4.2.12). In addition, appropriate bookkeeping devices and variables are used. The dynamic programming optimal alignment procedure is given as Algorithm 10. To the best of our knowledge, this is the first account of the idea of aligning, as opposed to concatenating partial transfer routes. The dynamic programming procedure for concatenating partial transfer routes when $\beta = 1$ and aligning them when $\beta > 1$ are quite similar. Although there are no reported implementations of Algorithm 10, we note that with care, an implementation running in $O(mn^2)$ time is possible.

4.3 Concluding Remarks

In this chapter, we applied the solution of the *2APX* procedure developed in Chapter 3 to the Fixed-sequence RRP. We examined the issues arising by the need to properly modify candidates for partial transfer routes as given by the *2APX* procedure to suit our need for a transfer route as a feasible solution to the Fixed-sequence RRP. To this end, we have introduced the concepts of concatenation, and provided Algorithm 8 as a method of extracting a feasible transfer route from a pool of candidates for partial transfer routes. Theorem 4.4 stated that adopting the *2APX* procedure (Algorithm 6, Chapter 3) and the straightforward

concatenation method of Algorithm 8 results in a polynomial-time approximation algorithm for the Fixed-sequence RRP, with approximation guarantee 2.

We examined the problem more deeply, and additionally introduced the concept of alignment as opposed to concatenation, and described dynamic programming procedures which given a pool of candidates for partial transfer routes return an optimal way of concatenating/aligning them (Algorithm 9 and 10, respectively). Even though these procedures do not improve the theoretically provable approximation ratio for the Fixed-sequence RRP, we hope to gain advantage of improved practical results, as reported in [52–55].

Algorithm 9 Optimal concatenation using dynamic programming for $\beta = 1$

Input: m approximate alternating Hamiltonian cycles $H'_{\ell-1,\ell}$, $\ell = 1, 2, \ldots, m$ computed by the *2APX* procedure.

Output: An approximate transfer route $\Pi' = (P'_{\ell-1,\ell} : \ell = 1, 2, \ldots, m)$, where the m partial transfer routes $P'_{\ell-1,\ell}$ satisfy the consecutiveness constraint, Eq. (1.3.7).

for $\ell \leftarrow 1$ to m **do**
  for $j \leftarrow 1$ to n **do**
    Set $k \leftarrow \sigma'_{\ell-1}(\tau^j_{\ell-1}(n))^{-1}$;
    Set $(\sigma^j_{\ell-1}, \tau^j_\ell) \leftarrow \operatorname{argmin} \begin{cases} w(\sigma^{\rightarrow}_{\ell-1} \gg (k-1), \tau^{\rightarrow}_\ell \gg (k-1)), \\ w(\sigma^{\leftarrow}_{\ell-1} \gg (k-1), \tau^{\leftarrow}_\ell \gg (k-1)); \end{cases}$
    if $z_\ell(j) > z_{\ell-1}(k) + w(\sigma^j_{\ell-1}, \tau^j_\ell)$ **then**
      Set $z_\ell(j) \leftarrow z_{\ell-1}(k) + w(\sigma^j_{\ell-1}, \tau^j_\ell)$;
      Set $pred_\ell(j) \leftarrow k$
    end if
  end for
end for;
{Backtracking:}
Set $k \leftarrow \operatorname{argmin}_{1 \le j \le n} z_m(j)$;
for $l \leftarrow m - 1$ downto 1 **do**
  $post_l(pred_{l+1}(k)) \leftarrow k$;
  $k \leftarrow pred_{l+1}(k)$
end for;
Set $k \leftarrow 1$;
for $\ell \leftarrow 1$ to m **do**
  Set $\sigma'_{\ell-1} \leftarrow \sigma^k_{\ell-1}$;
  Set $\tau'_\ell \leftarrow \tau^k_\ell$;
  $P'_{\ell-1,\ell} \leftarrow \big(\sigma'_{\ell-1}(1) \to \tau'_\ell(1) \to \sigma'_{\ell-1}(2) \to \tau'_\ell(2) \to \cdots \to \sigma'_{\ell-1}(n) \to \tau'_\ell(n)\big)$;
  Set $k \leftarrow post_\ell(k)$
end for;
Set $\Pi' \leftarrow (P'_{\ell-1,\ell} : \ell = 1, 2, \ldots, m)$;
return Π'.

Algorithm 10 Optimal alignment using dynamic programming for $\beta > 1$

Input: m approximate alternating Hamiltonian cycles $H'_{\ell-1,\ell}$, $\ell = 1, 2, \ldots, m$ computed by the *2APX* procedure.

Output: An aligned approximate transfer route $\widetilde{\Pi} = (P'_{0,1}; (\tau'_0(n), \sigma'_1(1)); \ldots; (\tau'_{n-1}(n), \sigma'_{n-1}(1)); P'_{n-1,n})$.

for $\ell \leftarrow 1$ to m **do**
  for $k \leftarrow$ to n **do**
    for $j \leftarrow 1$ to n **do**
      Set $\sigma^j_{\ell-1} \leftarrow \sigma'_{\ell-1} \gg (j-1)$;
      Set $\tau^j_\ell \leftarrow \tau'_\ell \gg (j-1)$;
      if $z_\ell(j) > z_{\ell-1}(k) + w(\sigma^j_{\ell-1}, \tau^j_\ell) + w(\tau^k_{\ell-1}(n), \sigma^j_{\ell-1}(1))$ **then**
        Set $z_\ell(j) \leftarrow z_{\ell-1}(k) + w(\sigma^j_{\ell-1}, \tau^j_\ell) + w(\tau^k_{\ell-1}(n), \sigma^j_{\ell-1}(1))$;
        Set $pred_\ell(j) \leftarrow k$
      end if
    end for
  end for
end for;
{Backtracking:}
Set $k \leftarrow \operatorname{argmin}_{1 \le j \le n} z_m(j)$;
for $l \leftarrow m-1$ downto 1 **do**
  $post_l(pred_{l+1}(k)) \leftarrow k$;
  $k \leftarrow pred_{l+1}(k)$
end for;
Set $k \leftarrow 1$;
for $\ell \leftarrow 1$ to m **do**
  Set $\sigma'_{\ell-1} \leftarrow \sigma^k_{\ell-1}$;
  Set $\tau'_\ell \leftarrow \tau^k_\ell$;
  $P'_{\ell-1,\ell} \leftarrow \big(\sigma'_{\ell-1}(1) \to \tau'_\ell(1) \to \sigma'_{\ell-1}(2) \to \tau'_\ell(2) \to \cdots \to \sigma'_{\ell-1}(n) \to \tau'_\ell(n)\big)$;
  Set $k \leftarrow post_\ell(k)$
end for;
Set $\widetilde{\Pi} \leftarrow (P'_{0,1}; (\tau'_0(n), \sigma'_1(1)); \ldots; (\tau'_{n-1}(n), \sigma'_{n-1}(1)); P'_{n-1,n})$;
return $\widetilde{\Pi}$.

5 Routing With a Permutable Processing Sequence

In the preceding chapter we saw how a pool of candidates for alternating Hamiltonian paths can be aligned and concatenated to obtain a feasible transfer route. The entire description evolved around aligning the alternating Hamiltonian paths in a given order. In this chapter, we will try to perturb the order in which alternating Hamiltonian paths are put together to form a transfer route.

5.1 Introduction

We have defined the Permutable-sequence RRP in Chapter 1: Introduction, and have been consistently elaborating techniques which would ultimately lead to an approximation algorithm for the Permutable-sequence RRP. As explained previously, in spite of the fact that we have chosen to approach the problem on two levels, finding a sequence ψ (permutation on the integer interval $[0, m]$ under the condition $\psi(0) = 0$) and a feasible transfer route Π_ψ with respect to ψ, these two facets are not as easily separable in the quest of minimizing $L(\Pi_\psi)$, Eq. (1.3.11).

The Permutable-sequence generalization of the unbiased version of the RRP has been examined by Karuno et al. [31], where a factor 8 approximation algorithm has been presented. Subsequently, for the unbiased version ($\beta = 1$), improved ratio approximation algorithms have been reported [33, 58], with approximation ratios of 6 and 3, respectively. To the best of our knowledge, no reports of algorithms of any kind exist in the literature for the biased version of the Permutable-sequence RRP.

Known approaches leverage the fact that just as a Hamiltonian path (or cycle) can be described by a permutation, so can a permutation be obtained through finding a Hamiltonian path (or cycle) in an appropriately defined graph. To this effect, we introduce the concept of a *universal graph*, which will be used as a scaffold around which we will build approximate solutions for the Permutable-sequence RRP, but predominantly, to determine a provably good sequence ψ which can be used in tandem with the approximate methods developed thus far for the Fixed-sequence RRP.

5.2 Sequencing Subproblem

Until this point, we have prepared the scene for the introduction of the universal graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as the complete graph $\mathcal{G}[\mathcal{C}]$, induced by the vertex sets in $\mathcal{C}$, i.e.,

$$\mathcal{V} := \{C_l : l = 0, 1, \ldots, m\}, \quad \mathcal{E} := \mathcal{V} \times \mathcal{V}.$$

In terms of the universal graph $\mathcal{G}$, we will refer to individual vertex sets C_l as *nodes*. Evidently, a Hamiltonian path ψ in $\mathcal{G}$ starting at node $C_0 \in \mathcal{V}$ can be described as a permutation ψ on the integers of the interval $[0, m]$ under the condition that $\psi(0) = 0$, and can be used to uniquely determine a processing sequence.

The main question which arises following the above insight is how to quantify the appropriateness of a Hamiltonian path ψ in the universal graph $\mathcal{G}$ as a transfer route sequence. In order to answer this question, let us assume that edges in the universal graph $\mathcal{G}$ are endowed with some non-negative edge weight function $\omega : \mathcal{E} \to \mathbb{R}_+$. Ensuing, the weight $\omega(\psi)$ of a permutation ψ becomes

$$\omega(\psi) = \sum_{l=1}^{m} \omega(C_{\psi(l-1)}, C_{\psi(l)}). \tag{5.2.1}$$

Now, let $P^*_{i,j}$ be an alternating Hamiltonian path in the bipartite graph $G_{i,j} = (C_i, C_j; C_i \times C_j)$, defined over the vertex sets C_i to C_j, which minimizes $\widetilde{w}(P^*_{i,j})$ (Eq. (1.4.14)), and let $\Pi^*_\psi = (P^\psi_{\psi(\ell-1),\psi(\ell)} : \ell = 1, 2, \ldots, m)$ be a feasible transfer route minimizing the value of $L(\Pi^*_\psi) = \sum_{\ell=1}^{m} \widetilde{w}(P^\psi_{\psi(\ell-1),\psi(\ell)})$ over the given sequence ψ. By definition, it holds that

$$\widetilde{w}(P^*_{\psi(\ell-1),\psi(\ell)}) \leq \widetilde{w}(P^\psi_{\psi(\ell-1),\psi(\ell)}). \tag{5.2.2}$$

By choosing values for $\omega : \mathcal{E} \to \mathbb{R}_+$ such that

$$\omega(C_i, C_j) \leq \widetilde{w}(P^*_{i,j}), \tag{5.2.3}$$

it immediately follows that

$$\omega(\psi) \leq L(\Pi^*_\psi). \tag{5.2.4}$$

Finally, for a Hamiltonian path ψ^*, which is optimal with respect to the edge weight function ω in the universal graph $\mathcal{G}$, in the sense that

$$\omega(\psi^*) \leq \omega(\psi),$$

for any other Hamiltonian path ψ in the universal graph $\mathcal{G}$ (starting from C_0), we can attest that

$$\omega(\psi^*) \leq L(\Pi^*_\psi) \tag{5.2.5}$$

holds for any processing sequence ψ and a transfer route Π^*_ψ which is optimal with respect to the processing sequence ψ.

The above observations face an unfavorable fact, namely, we have already stated that not only is finding an exact optimal solution an NP-complete optimization problem even for alternating Hamiltonian paths, but also that optimizing individual alternating Hamiltonian paths does not necessarily lead to a global optimal solution for the Permutable-sequence RRP.

However, in previous chapters it was demonstrated how approximate alternating Hamiltonian paths can be obtained by heuristic methods. We have presented a few different methods for building Hamiltonian cycles with a constant factor approximation guarantee with respect to lower bounds of an alternating Hamiltonian path. Nevertheless, it is still plausible that those methods may be refined, or even new ones developed in the future. Therefore, for greater generality, let us assume that we can build an approximate alternating Hamiltonian cycle $H'_{i,j}$ such that

$$\widetilde{w}(H'_{i,j}) \leq \rho \widetilde{w}(P^*_{i,j}), \quad \rho \geq 1 \tag{5.2.6}$$

for an alternating Hamiltonian path $P^*_{i,j}$ in the bipartite graph $G_{i,j}$ which minimizes its cost $\widetilde{w}(P^*_{i,j})$. As described in Chapter 4: Fixed Sequence Repetitive Routing, a collection of alternating Hamiltonian cycles can be thought of as a pool of candidate solutions for alternating Hamiltonian paths. In order to eventually obtain a feasible transfer route Π', alternating Hamiltonian paths could be extracted from a given pool of candidates, as detailed in Algorithms 8, 9 and 10. After stating the above reference as a reminder to results from previous chapters, we state the following theorem.

Theorem 5.5. *Given an instance $(\mathcal{C}, w, \beta)$ of the Permutable-sequence RRP, let $\mathcal{G} = (\mathcal{C}, \mathcal{C} \times \mathcal{C})$ be the corresponding universal graph. Let the edge weight function ω in the universal graph $\mathcal{G}$ be chosen such that for any pair of vertex sets C_i and C_j defining a bipartite graph $G_{i,j} = (C_i, C_j; C_i \times C_j)$, it is possible to build, in polynomial time, an alternating Hamiltonian cycle $H'_{i,j}$ and*

$$\widetilde{w}(H'_{i,j}) \leq \omega(C_i, C_j) \leq \rho \widetilde{w}(P^*_{i,j}), \tag{5.2.7}$$

holds for any alternating Hamiltonian path $P^*_{i,j}$ which minimizes $\widetilde{w}(P^*_{i,j})$ and some $\rho \geq 1$. Let ψ^* be a Hamiltonian path in $\mathcal{G}$ starting from C_0, such that

$$\omega(\psi^*) \leq \omega(\psi) \tag{5.2.8}$$

for any other Hamiltonian path ψ in $\mathcal{G}$ starting at C_0. If we can compute a γ-approximate Hamiltonian path ψ' starting at C_0 in $\mathcal{G}$ such that

$$\omega(\psi') \leq \gamma \omega(\psi^*),$$

then we can build in polynomial time a transfer route $\Pi'_{\psi'}$ *with respect to the sequence* ψ' *such that*

$$L(\Pi'_{\psi'}) \leq \gamma\rho \cdot L(\Pi^*_{\mathbf{\Psi}}), \tag{5.2.9}$$

holds for an optimal solution $\Pi^*_{\mathbf{\Psi}} = (P^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)} : \ell = 1, 2, \ldots, m)$ *to the instance of the Permutable-sequence RRP.*

Proof. By the proposition, for each $\ell = 1, 2, \ldots, m$, we can obtain an alternating Hamiltonian cycle $H'_{\psi'(\ell-1),\psi'(\ell)}$ with

$$\widetilde{w}(H'_{\psi'(\ell-1),\psi'(\ell)}) \leq \omega(C_{\psi'(\ell-1)}, C_{\psi'(\ell)}).$$

From Algorithm 8, we can obtain alternating Hamiltonian paths $P'_{\psi'(\ell-1),\psi'(\ell)}$ satisfying the consecutiveness constraint, and therefore a feasible transfer route $\Pi'_{\psi'} := (P'_{\psi'(\ell-1),\psi'(\ell)} : \ell = 1, 2, \ldots, m)$. Note that all alternating Hamiltonian paths $P'_{\psi'(\ell-1),\psi'(\ell)}$ obtained in this manner satisfy

$$\widetilde{w}(P'_{\psi'(\ell-1),\psi'(\ell)}) \leq \widetilde{w}(H'_{\psi'(\ell-1),\psi'(\ell)}).$$

Consequently, for the length of the approximate transfer route $\Pi'_{\psi'}$ we can write

$$\begin{aligned} L(\Pi'_{\psi'}) &= \sum_{\ell=1}^{m} \tilde{w}(P'_{\psi'(\ell-1),\psi'(\ell)}) \\ &\leq \sum_{\ell=1}^{m} \tilde{w}(H'_{\psi'(\ell-1),\psi'(\ell)}) \\ &\leq \sum_{\ell=1}^{m} \omega(C_{\psi'(\ell-1)}, C_{\psi'(\ell)}) \\ &= \omega(\psi'). \end{aligned} \tag{5.2.10}$$

On the other hand, given an optimal sequence ψ^* as a Hamiltonian path in the universal graph $\mathcal{G}$ minimizing $\omega(\psi^*)$, for the cost of the sequence ψ' we have

$$\begin{aligned} \omega(\psi') &\leq \gamma\omega(\psi^*) \leq \gamma\omega(\mathbf{\Psi}) \\ &= \gamma \sum_{\ell=1}^{m} \omega(C_{\mathbf{\Psi}(\ell-1)}, C_{\mathbf{\Psi}(\ell)}) \\ &\leq \gamma\rho \sum_{\ell=1}^{m} \widetilde{w}(P^*_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)}) \\ &\leq \gamma\rho \sum_{\ell=1}^{m} \widetilde{w}(P^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)}) \\ &= \gamma\rho L(\Pi^*_{\mathbf{\Psi}}). \end{aligned} \tag{5.2.11}$$

Finally, we compare the expressions from Eqs. (5.2.10) and (5.2.11) to verify the claim. □

Unfortunately, even computing such an optimal Hamiltonian path ψ^* as in Theorem 5.5 is itself an NP-complete problem. Even more so, it is in general NP-hard to find an approximate sequence ψ such that

$$\omega(\psi) \leq \gamma\omega(\psi^*), \tag{5.2.12}$$

would hold, for any positive constant γ ([49]). Such an approximation is possible only if the edge weight function ω in the universal graph $\mathcal{G}$ satisfies certain conditions. However, under the assumption that the edge weight function ω is symmetric and satisfies the triangle inequality, a modification of the well-known Christofides heuristic [16] for finding a Hamiltonian path starting at a designated vertex has been given by Hoogeveen [29]. This method can be used to achieve a value of $\gamma = 1.5$.

We devote the remainder of the chapter to investigating how the approaches for building alternating Hamiltonian cycles and paths as described in Chapter 3 can be adopted to provide values for the edge weight ω in the universal graph $\mathcal{G}$.

5.3 Assigning Edge Weights in the Universal Graph

The two main conditions that the edge weight ω in the universal graph $\mathcal{G}$ needs to satisfy in order for the result of Theorem 5.5 to benefit from a constant value for the parameter γ, are symmetry and the triangle inequality.

Early reports of heuristic solutions for the sequencing problem [31, 33], giving respectively approximation ratios of 8 ($\rho = 4$, $\gamma = 2$) and 6 ($\rho = 4$, $\gamma = 1.5$) adopted an upper bound on the cost of candidate solutions as an edge weight in the universal graph. The upper bounds are in turn weights (with respect to a metric w) of the building blocks used to build an approximate candidate solution for an alternating Hamiltonian path. The heuristic method used is the $SWAP_2$ procedure, given as Algorithm 3 in Chapter 3. The accompanying Lemma 3.7 provides a value for $\rho \leq 4$, Eq. (3.3.27).

Let $M^*_{i,j}$ denote a perfect bipartite matching of minimum $w(M^*_{i,j})$ in the bipartite graph $G_{i,j} = (C_i, C_j; C_i \times C_j)$. Similarly, let T^*_i and T^*_j be minimum weight (w.r.t. the metric weight function w) spanning trees in the induced complete graphs $G[C_i]$ and $G[C_j]$, respectively.

We set the edge weight ω in the universal graph $\mathcal{G}$ as

$$\omega(C_i, C_j) := (\beta + 1)w(M^*_{i,j}) + w(T^*_i) + w(T^*_j). \tag{5.3.13}$$

By the fact that the edge weight function w adopted in the bipartite graph $G_{i,j}$ is symmetric, so will be the edge weights given by ω in the universal graph $\mathcal{G}$. The main challenge left is to show that the expression from Eq. (5.3.13) satisfies the triangle inequality. With this in mind, we state the following.

Lemma 5.13. *For an instance $(\mathcal{C}, w, \beta)$ of the RRP, let $\omega(C_i, C_j)$ be given as*

$$\omega(C_i, C_j) = (\beta + 1)w(M^*_{i,j}) + w(T^*_i) + w(T^*_j).$$

Then, adopted as an edge weight function for the universal graph $\mathcal{G} := G[\mathcal{C}]$, ω is symmetric and satisfies the triangle inequality.

Proof. Symmetry of ω immediately follows from the symmetry of the edge weight function w. Therefore we proceed by examining if the inequality

$$\omega(C_i, C_k) + \omega(C_k, C_j) \geq \omega(C_i, C_j)$$

holds for every triplet C_i, C_j and C_k.

Let us consider two bipartite graphs, $G_{i,k}$ and $G_{k,j}$ over three different configurations, C_i, C_j and C_k, and perfect bipartite matchings $M_{i,k}$ and $M_{k,j}$. Every vertex $c^i_x \in C_i$ is incident with a unique vertex $c^k_y \in C_k$ by an edge in the matching $M_{i,k}$. The vertex $c^k_y \in C_k$ in turn is incident with a unique $c^j_z \in C_j$ by an edge in $M_{k,j}$. We can replace the matching edges $\{c^i_x, c^k_y\}$ and $\{c^k_y, c^j_z\}$ with an edge $\{c^i_x, c^j_z\}$, and by the assumption of a metric edge weight function w, it inevitably holds that $w(c^i_x, c^j_z) \leq w(c^i_x, c^k_y) + w(c^k_y, c^j_z)$. Repeating this process for every triplet of points thus related by the matchings $M_{i,k}$ and $M_{k,j}$, we can derive a perfect bipartite matching $M'_{i,j}$, as in Figure 5.1, with $w(M'_{i,j}) \leq w(M_{i,k}) + w(M_{k,j})$. Naturally, for a minimum cost perfect bipartite matching $M^*_{i,j}$ it holds

$$w(M^*_{i,j}) \leq w(M'_{i,j}).$$

In conclusion, we can write

$$\begin{aligned}
\omega(C_i, C_k) + \omega(C_k, C_j) =& (\beta + 1)w(M^*_{i,k}) + w(T^*_i) + w(T^*_k) \\
&+ (\beta + 1)w(M^*_{j,k}) + w(T^*_k) + w(T^*_j) \\
\geq& (\beta + 1)(w(M^*_{i,k}) + w(M^*_{k,j})) + w(T^*_i) + w(T^*_j) \\
\geq& (\beta + 1)w(M'_{i,j}) + w(T^*_i) + w(T^*_j) \\
\geq& (\beta + 1)w(M^*_{i,j}) + w(T^*_i) + w(T^*_j) \\
=& \omega(C_i, C_j). \qquad (5.3.14)
\end{aligned}$$

□

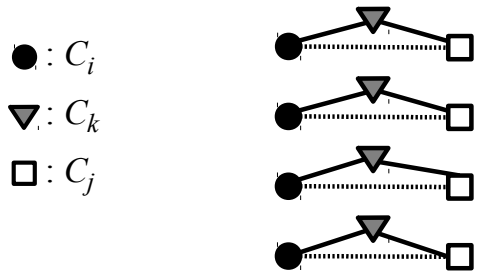

Figure 5.1. Two perfect bipartite matchings $M_{i,k}$ and $M_{k,j}$ such that the vertex set C_k is covered by both of them, can be shortcut to a perfect bipartite matching $M'_{i,j}$.

In a close relation with the result from Lemma 3.7, we state the following.

Lemma 5.14. *Let the value $\omega(C_i, C_j)$ be set as in Eq. (5.3.13). For any given alternating Hamiltonian path $P_{i,j}$, let the bias factor β influence the weight of the edges traversed in the direction from the vertex set C_i to C_j and let this edge set be denoted as $\overrightarrow{P}_{i,j}$; while the set of edges traversed in the direction from C_j to C_i is denoted as $\overleftarrow{P}_{i,j}$. Following, the parameter*

$$\kappa_{i,j} := \frac{w(\overleftarrow{P}_{i,j})}{w(\overrightarrow{P}_{i,j})}, \quad 0 \leq \kappa_{i,j} < \infty,$$

parameterizes the value of the biased path cost $\widetilde{w}(P_{i,j})$ as

$$\widetilde{w}(P_{i,j}) = (\beta + \kappa_{i,j}) w(\overrightarrow{P}_{i,j}).$$

Then, it holds

$$\omega(C_i, C_j) \leq \left(1 + \frac{3 + \kappa_{i,j}}{\beta + \kappa_{i,j}}\right) \widetilde{w}(P_{i,j}). \tag{5.3.15}$$

Proof. We begin by reviewing the lower bounds used in the proof of Lemma 3.7. Briefly, given an alternating Hamiltonian path $P_{i,j}$ whose cost $\widetilde{w}(P_{i,j})$ can be written as

$$\widetilde{w}(P_{i,j}) = \beta w(\overrightarrow{P}_{i,j}) + w(\overleftarrow{P}_{i,j}) = (\beta + \kappa_{i,j}) w(\overrightarrow{P}_{i,j}),$$

for a real parameter $0 \leq \kappa_{i,j} < \infty$, in Chapter 3 we have demonstrated that

$$w(T_i^*) \leq w(\overrightarrow{P}_{i,j}) + w(\overleftarrow{P}_{i,j}) = (1 + \kappa_{i,j}) w(\overrightarrow{P}_{i,j}), \tag{5.3.16}$$

$$w(T_j^*) \leq w(\overrightarrow{P}_{i,j}) + w(\overleftarrow{P}_{i,j}) = (1 + \kappa_{i,j}) w(\overrightarrow{P}_{i,j}), \tag{5.3.17}$$

$$w(M_{i,j}^*) \leq w(\overrightarrow{P}_{i,j}). \tag{5.3.18}$$

It straightforwardly follows that

$$\begin{aligned}\omega(C_i, C_j) &= w(T_i^*) + w(T_j^*) + (\beta+1)w(M_{i,j}^*) \\ &\leq 2(w(\overrightarrow{P}_{i,j}) + w(\overleftarrow{P}_{i,j})) + (\beta+1)w(\overrightarrow{P}_{i,j}) \\ &= (\beta + 2\kappa_{i,j} + 3)w(\overrightarrow{P}_{i,j}) \\ &= \frac{\beta + 2\kappa_{i,j} + 3}{\beta + \kappa_{i,j}}\widetilde{w}(P_{i,j}). \end{aligned} \tag{5.3.19}$$

□

At this point it is worthwhile to be reminded of the result of Algorithm 3 and Lemma 3.7, and that for the given edge weight function $\omega(C_i, C_j)$ in the universal graph, it is indeed possible to build an alternating Hamiltonian cycle $H'_{i,j}$ in polynomial time, such that

$$\widetilde{w}(H'_{i,j}) \leq \omega(C_i, C_j) \leq \frac{\beta + 2\kappa_{i,j} + 3}{\beta + \kappa_{i,j}}\widetilde{w}(P_{i,j}), \tag{5.3.20}$$

and that this is valid for any alternating Hamiltonian path $P_{i,j}$ and accompanying parameter $\kappa_{i,j}$.

Next, we formalize the parameter K_ψ (analogous to Eq. (4.1.4)), which for a given sequence ψ parameterizes the value $L(\Pi_\psi^*)$ of a transfer route Π_ψ^*, which is optimal with respect to ψ, as

$$K_\psi := \frac{\sum_{\ell=1}^{m} w(\overleftarrow{P^\psi}_{\psi(\ell-1),\psi(\ell)})}{\sum_{\ell=1}^{m} w(\overrightarrow{P^\psi}_{\psi(\ell-1),\psi(\ell)})}, \quad 0 \leq K_\psi < \infty, \tag{5.3.21}$$

such that for the cost $L(\Pi_\psi^*)$ we can write (see Eq. (4.1.5))

$$\begin{aligned} L(\Pi_\psi^*) &= \beta \sum_{\ell=1}^{m} w(\overrightarrow{P^\psi}_{\psi(\ell-1),\psi(\ell)}) + \sum_{\ell=1}^{m} w(\overleftarrow{P^\psi}_{\psi(\ell-1),\psi(\ell)}) \\ &= (\beta + K_\psi) \sum_{\ell=1}^{m} w(\overrightarrow{P^\psi}_{\psi(\ell-1),\psi(\ell)}). \end{aligned} \tag{5.3.22}$$

Theorem 5.6. *For a given instance $(\mathcal{C}, w, \beta)$ of the Permutable-sequence RRP, let $\Pi_{\mathbf{\Psi}}^*$ denote a global optimal solution minimizing the cost $L(\Pi_{\mathbf{\Psi}}^*)$. In polynomial time, we can compute an approximate sequence ψ and a feasible transfer route Π'_ψ with respect to ψ, such that*

$$L(\Pi'_\psi) \leq \frac{3}{2}\left(1 + \frac{3 + K_{\mathbf{\Psi}}}{\beta + K_{\mathbf{\Psi}}}\right) L(\Pi_{\mathbf{\Psi}}^*). \tag{5.3.23}$$

Proof. Let $\mathcal{G}$ be the universal graph appropriately defined for the given instance $(\mathcal{C}, w, \beta)$. For each pair of vertex sets $C_i, C_j \in \mathcal{C}$ as nodes in the universal graph, the edge cost function $\omega(C_i, C_j)$ is set as in Eq. (5.3.13),

$$\omega(C_i, C_j) := w(T_i^*) + w(T_j^*) + (\beta + 1)w(M_{i,j}^*), \quad 0 \leq i \neq j \leq m$$

and as a consequence of Lemma 5.13, a Hamiltonian walk in the universal graph $\mathcal{G}$, i.e., a sequence ψ such that

$$\omega(\psi) \leq \frac{3}{2}\omega(\mathbf{\Psi}) \tag{5.3.24}$$

can be computed in polynomial time [29]. For the alternating Hamiltonian paths $P^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)}$, $\ell = 1, 2, \ldots, m$ constituting the transfer route $\Pi^*_{\mathbf{\Psi}}$, which is optimal with respect to the sequence $\mathbf{\Psi}$, we adopt the parameters

$$\kappa^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)} := \frac{w(\overleftarrow{P^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)}})}{w(\overrightarrow{P^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)}})}, \quad 0 \leq \kappa^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)} < \infty,$$

so that the individual costs $\widetilde{w}(P^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)})$ can be parameterized as

$$\begin{aligned}\widetilde{w}(P^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)}) &= \beta w(\overrightarrow{P^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)}}) + w(\overleftarrow{P^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)}}) \\ &= (\beta + \kappa^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)}) w(\overrightarrow{P^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)}}),\end{aligned}$$

where the $\overrightarrow{P^{\mathbf{\Psi}}_{\mathbf{\Psi}(\ell-1),\mathbf{\Psi}(\ell)}}$ denote the sets of edges traversed in the direction which is influenced by the bias factor β. For the cost $L(\Pi'_\psi)$ of the transfer route Π'_ψ we can write

$$\begin{aligned}L(\Pi'_\psi) &= \sum_{\ell=1}^{m} \widetilde{w}(P'_{\psi(\ell-1),\psi(\ell)}) \\ &\leq \sum_{\ell=1}^{m} \widetilde{w}(H'_{\psi(\ell-1),\psi(\ell)}) \\ &\leq \sum_{\ell=1}^{m} \omega(C_{\psi(\ell-1)}, C_{\psi(\ell)}) \\ &= \omega(\psi),\end{aligned} \tag{5.3.25}$$

where substituting from Eq. (5.3.24) and following Lemma 5.14, Eq. (5.3.15) we

obtain

$$\begin{aligned}
L(\Pi'_\psi) &\leq \omega(\psi) \leq \frac{3}{2}\omega(\boldsymbol{\Psi}) \\
&= \frac{3}{2}\sum_{\ell=1}^{m}\omega(C_{\boldsymbol{\Psi}(\ell-1)}, C_{\boldsymbol{\Psi}(\ell)}) \\
&\leq \frac{3}{2}\sum_{\ell=1}^{m}\left(1+\frac{3+\kappa^{\boldsymbol{\Psi}}_{\boldsymbol{\Psi}(\ell-1),\boldsymbol{\Psi}(\ell)}}{\beta+\kappa^{\boldsymbol{\Psi}}_{\boldsymbol{\Psi}(\ell-1),\boldsymbol{\Psi}(\ell)}}\right)\widetilde{w}(P^{\boldsymbol{\Psi}}_{\boldsymbol{\Psi}(\ell-1),\boldsymbol{\Psi}(\ell)}) \\
&= \frac{3}{2}\sum_{\ell=1}^{m}\left(\beta+2\kappa^{\boldsymbol{\Psi}}_{\boldsymbol{\Psi}(\ell-1),\boldsymbol{\Psi}(\ell)}+3\right)w(\overrightarrow{P^{\boldsymbol{\Psi}}_{\boldsymbol{\Psi}(\ell-1),\boldsymbol{\Psi}(\ell)}}) \\
&= \frac{3}{2}\left((\beta+3)\sum_{\ell=1}^{m}w(\overrightarrow{P^{\boldsymbol{\Psi}}_{\boldsymbol{\Psi}(\ell-1),\boldsymbol{\Psi}(\ell)}})+2\sum_{\ell=1}^{m}w(\overleftarrow{P^{\boldsymbol{\Psi}}_{\boldsymbol{\Psi}(\ell-1),\boldsymbol{\Psi}(\ell)}})\right) \\
&= \frac{3}{2}\left(\beta+2K_{\boldsymbol{\Psi}}+3\right)\sum_{\ell=1}^{m}w(\overrightarrow{P^{\boldsymbol{\Psi}}_{\boldsymbol{\Psi}(\ell-1),\boldsymbol{\Psi}(\ell)}}) \\
&= \frac{3}{2}\left(\frac{\beta+2K_{\boldsymbol{\Psi}}+3}{\beta+K_{\boldsymbol{\Psi}}}\right)\sum_{\ell=1}^{m}\widetilde{w}(P^{\boldsymbol{\Psi}}_{\boldsymbol{\Psi}(\ell-1),\boldsymbol{\Psi}(\ell)}) \\
&= \frac{3}{2}\left(\frac{\beta+2K_{\boldsymbol{\Psi}}+3}{\beta+K_{\boldsymbol{\Psi}}}\right)L(\Pi^*_{\boldsymbol{\Psi}}). \qquad (5.3.26)
\end{aligned}$$

□

In conclusion, the expression from Eq. (5.3.23) is bounded above by 6 for any positive real value of the parameter $K_{\boldsymbol{\Psi}}$, implying that

$$L(\Pi'_\psi) \leq 6L(\Pi^*_{\boldsymbol{\Psi}}) \qquad (5.3.27)$$

for any value of the bias $\beta \geq 1$.

5.3.1 Improved Guarantees for the Unbiased Case

The *2APX* procedure described as Algorithm 6 in Chapter 3 and the result of Theorem 3.1 hinted to the possibility that a value for $\rho = 2$ might be attainable. This would bring the provable approximation guarantee for the Permutable-sequence RRP from 6, as given in Theorem 5.6 to a more favorable value of 3. Alas, adopting the result or the building blocks from the *2APX* procedure faces an obstacle, for we do not have a way to show that such an edge weight for the universal graph satisfies the triangle inequality. Without such a property as the triangle inequality, we are unable to use any of the tools for approximating Hamiltonian paths for deriving a processing sequence.

Nevertheless, for the limited unbiased case (i.e., the bias $\beta = 1$) of the RRP, we can rely on a nice property of alternating Hamiltonian paths presented in [58].

Before proceeding, let us exhibit certain claims necessary for a clear presentation of our idea.

We introduce the graph operation of splitting, which replaces two adjacent edges $\{u, y\}$ and $\{y, v\}$ by a new edge $\{u, v\}$, thus shortcutting the vertex y. Given a graph $G = (V, E)$, and a vertex $v \in V$ of even degree $d_G(v) = 2k$, $k \in \mathbb{Z}_+$, by the operation of splitting the vertex v we designate the aggregated operations of splitting all pairs of edges incident with v and removing v from the vertex set V. It is not difficult to discern that for $k \geq 1$, there are $\prod_{i=0}^{k-1}(2i+1)$ different ways to split the vertex v.

Lemma 5.15 ([58]). *Let $G = (V, E)$ be an Eulerian graph, with a distinguished vertex $v \in V$ of degree 4. Among the three different ways to split v there are at least two ways in which the graph obtained by splitting v remains connected, and consequently, Eulerian.*

Proof. Let q, r, s, and t be the four neighbors of the vertex v in G. All vertex degrees $d_G(q)$, $d_G(r)$, $d_G(s)$ and $d_G(t)$ are even (consequence of G being an Eulerian graph). Let G' denote one of the three possible resulting graphs obtained as described above. Without loss of generality, let the resulting new edges be $\{q, r\}$ and $\{s, t\}$. Assume that G' is disconnected. Then, G' has two components G'_1 and G'_2. Let the component G'_1 contain vertices q and r, as well as the newly obtained edge $\{q, r\}$, and component G'_2 contains the vertices s and t and the edge $\{s, t\}$, as in Figure 5.2 (b). We see that G'_h $(h = 1, 2)$ must remain connected even if we remove the newly formed edges, since otherwise there would be components which would have only one odd degree vertex. Thereby, we have a contradiction to the fact that the number of vertices with odd degree in a graph is even. Therefore the remaining two ways to split the vertex v must result in a connected graph, for they will connect the odd degree vertices of G'_1 to those of G'_2. Moreover, since the splitting of v does not change the degree of any other vertex, in both cases the newly obtained graphs by splitting the vertex v remain Eulerian. □

Lemma 5.15 enables us to employ a neat trick with the alternating Hamiltonian cycles from our pool of candidate solutions for alternating Hamiltonian paths. Namely, let us consider two alternating Hamiltonian cycles, $H'_{i,k}$ and $H'_{k,j}$, in the bipartite graphs $G_{i,k}$ and $G_{k,j}$, respectively. We observe the union of $H'_{i,k}$ and $H'_{k,j}$, as illustrated in Figure 5.3 (a). Now, we can iteratively call on the vertex detachment result of Lemma 5.15, for every vertex in C_k. As a result, we will obtain a connected graph in which the degree of all vertices in G_i and G_j is 2, and that is exactly an alternating Hamiltonian path $H''_{i,j}$, as in Figure 5.3 (b).

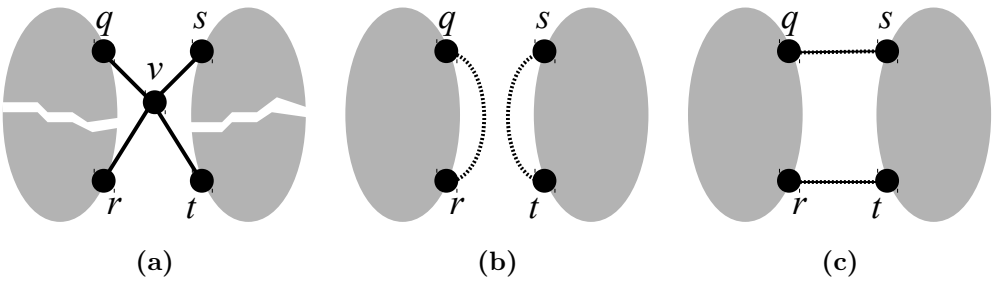

Figure 5.2. (a) Every vertex in an Eulerian graph $G = (V, E)$ has even degree, and the distinguished vertex $v \in V$ is of degree $d_G(v) = 4$; (b) The only way of detaching v resulting in a disconnected graph with components G'_1 containing vertices q and r, and G'_2 containing s and t; (c) Another way to split the vertex V, resulting in a connected graph G''; the only remaining way is by adding the edges $\{q, t\}$ and $\{r, s\}$, again resulting in a connected graph.

Due to the triangle inequality which holds for the edge weight function w, with each vertex detachment and shortcutting through a pair of edges, the total weight does not increase, therefore

$$w(H''_{i,j}) \leq w(H'_{i,k}) + w(H'_{k,j}). \tag{5.3.28}$$

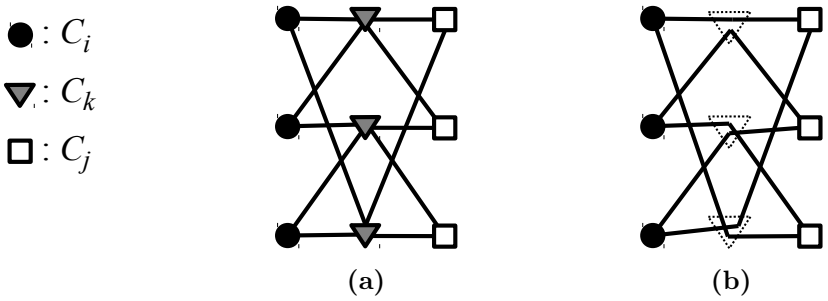

Figure 5.3. (a) The union of two alternating Hamiltonian cycles $H'_{i,k}$ and $H'_{k,j}$ with a common vertex set C_k; (b) The vertex detachment procedure described in [58] which facilitates the use of the metric closure of the universal graph $\mathcal{G}$.

Finally, the expression from Eq. (5.3.28) brings us to a powerful technique which we can use to satisfy the triangle inequality when assigning edge weights for the universal graph $\mathcal{G}$.

Let $\{H_{i,j} : 0 \leq i \neq j \leq m\}$ be a pool of candidate solutions from which we can extract feasible alternating Hamiltonian paths (the alternating Hamiltonian cycles may be obtained by, e.g., the *2APX* procedure). We initially set edge weights $\omega(C_i, C_j) := w(H'_{i,j})$ for every edge $\{C_i, C_j\} \in \mathcal{E}$ of the universal graph.

Next, we find the metric closure of $\mathcal{G}$, and adopt $\omega^*(C_i, C_j)$, the length of a shortest C_i-C_j path with respect to ω in $\mathcal{G}$.

Case $\omega(C_i, C_j) = \omega^*(C_i, C_j)$**:** Assign $H''_{i,j} := H'_{i,j}$;

Case $\omega(C_i, C_j) \geq \omega^*(C_i, C_j)$**:** Let $C_i, C_{k_1}, C_{k_2}, \ldots, C_{k_h}, C_j$ be a C_i-C_j path which is shortest with respect to ω in the universal graph $\mathcal{G}$, where $h \leq m-1$. We iteratively apply the vertex detachment procedure over consecutive vertex sets C_{k_y}, $y = 1, 2, \ldots, h$, to finally obtain $H''_{i,j}$ from the list of alternating Hamiltonian cycles $H'_{i,k_1}, H'_{k_1,k_2}, \ldots, H'_{k_h,j}$.

In the end, we can use the edge weight ω^* to find a $\gamma = 1.5$-approximate Hamiltonian path ψ in the universal graph starting from C_0, and adopting ψ as the desired sequence, use a concatenation procedure (e.g., Algorithm 9) with the pool of candidate solutions $\{H_{\psi(\ell-1),\psi(\ell)} : \ell = 1, 2, \ldots, m\}$ to obtain a feasible transfer route Π'_{ψ}. Following Theorem 3.1 which gives a value for $\rho = 2$, and Theorem 5.5, we can conclude that

Lemma 5.16 ([58])**.** *The Permutable-sequence RRP with a metric weight function w and no bias ($\beta = 1$) can be approximated with a constant factor guarantee* 3 *in polynomial time.*

Before concluding this section, we would just give a brief comment as to the reason why the *2APX* procedure cannot be used in a similar fashion as just described to provide a value of $\rho = 2$ for any value of the bias $\beta \geq 1$ in the Permutable-sequence version of the RRP. Mainly, to achieve the approximation ratio of 2, in the *2APX* procedure we assigned traversal direction for the alternating Hamiltonian paths, such that exactly the edges of a minimum weight perfect bipartite matching are additionally weighed by the bias factor. In the case $\beta = 1$ this can be ignored, but for any $\beta > 1$, the already assigned orientation (direction) in which edges should be traversed limits the ways in which we can detach a vertex. Therefore, we cannot rely on the vertex detachment result of Lemma 5.15. An illustrative example is given in Figure 5.4.

5.4 Concluding Remarks

In this chapter we investigated the most general level of the family of problems around which this thesis revolved, the Permutable-sequence RRP, and we discussed both biased ($\beta \geq 1$) and unbiased ($\beta = 1$) versions of the problem. We developed a general framework which uses two levels of approximation, on a sequencing level, with approximation guarantee γ, and on a routing level, with

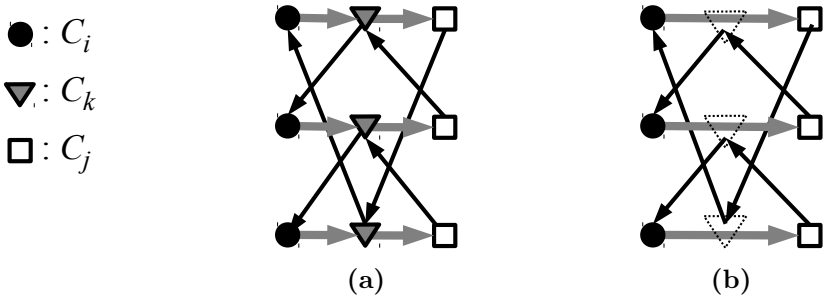

Figure 5.4. (a) The union of two alternating Hamiltonian paths $H'_{i,k}$ and $H'_{k,j}$ with determined traversal orientation; (b) A case where the detachment procedure described in [58] fails in a biased setting with $\beta \geq 1$. Observe that if the detachment needs to satisfy traversal directions, it might result in a graph with disconnected components.

an approximation guarantee ρ, and showed how under certain assumptions we can leverage approximation approaches independently, for a joint approximation ratio of $\rho \cdot \gamma$ for the Permutable-sequence RRP.

In addition, we examined the issues arising from utilizing the approaches elaborated in previous chapters, and obtained a constant factor 6 ($\rho = 4$, $\gamma = 1.5$) approximation guarantee for the general biased ($\beta \geq 1$) version of the Permutable-sequence RRP, and a constant factor 3 ($\rho = 2$, $\gamma = 1.5$) approximation guarantee for the special, unbiased ($\beta = 1$) case of the Permutable-sequence RRP.

6 Conclusion

This thesis revolved around a problem inspired by an application of a grasp-and-delivery robot at an assembly line for printed circuit boards.

In short, we are asked to find a routing of the grasp-and-delivery robot in order to minimize a certain cumulative cost incurred by the robot's motion. As an additional degree of freedom, we are also enabled to choose the processing order in which printed circuit boards arrive at the assembly line. Both the routing and the sequencing problem have a high degree of mutual impact, and cannot be optimized independently.

We have seen that even at a basic level, finding an optimal routing is a challenging NP-hard optimization problem, and every layer of generalization only contributes to the computational intractability of finding an optimal solution. Therefore, we have set out to build polynomial-time algorithms which provide approximate solutions.

During the course of pursuit for solution methods for the Repetitive Routing Problem we have stumbled upon a new way of approximating a problem with applications reaching far beyond the scope of this thesis, the Bipartite TSP and its biased generalization, elaborated in Chapter 3.

We have incorporated the solution to the biased generalization of the bipartite TSP in a solution method that for a determined processing sequence uses dynamic programming to align partial solutions in the pursuit for an even better approximate solution to the RRP.

Lastly, we examined how we can incorporate our solution methods developed thus far into a general framework which would address the sequencing and the routing problems at the same time. We proposed a sort of multi-level heuristic approach which uses the bounds obtained at one level to sustain the approximation guarantees derived at the next. At this stage we faced a problem with satisfying certain necessary conditions, and had to introduce additional tools which would help us towards a general solution for all levels of the RRP with a permutable processing sequence.

We hope that the tools exhibited and conclusions drawn in this thesis will prove useful in future research and advancement in the fields of routing, scheduling, approximation algorithms, and related optimization fields.

Bibliography

[1] T. Akiyama, T. Nishizeki, and N. Saito. NP-completeness of the Hamiltonian cycle problem for bipartite graphs. *Journal of Information Processing*, 3(2): 73–76, 1980.

[2] J. M. Aldous and R. J. Wilson. *Graphs and Applications: An Introductory Approach*. Springer, 2000.

[3] H.-C. An, R. Kleinberg, and D. B. Shmoys. Improving Christofides' algorithm for the s-t path TSP. In *Proceedings of the 44th symposium on Theory of Computing*, STOC '12, pages 875–886, New York, NY, USA, 2012.

[4] S. Anily and R. Hassin. The swapping problem. *Networks*, 22(4):419–433, 1992.

[5] S. Anily and G. Mosheiov. The traveling salesman problem with delivery and backhauls. *Operations Research Letters*, 16(1):11–18, 1994.

[6] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook. *The Traveling Salesman Problem: A Computational Study (Princeton Series in Applied Mathematics)*. Princeton University Press, 2007.

[7] S. Arora. Polynomial time approximation schemes for euclidean TSP and other geometric problems. In *Proceedings of the 37th IEEE Symposium on Foundations of Computer Science (FOCS96)*, pages 2–11, 1996.

[8] S. Arora. Nearly linear time approximation schemes for euclidean TSP and other geometric problems. In *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science (FOCS97)*, pages 554–563, IEEE, 1997.

[9] M. J. Atallah and S. R. Kosaraju. Efficient solutions to some transportation problems with applications to minimizing robot arm travel. *SIAM Journal on Computing*, 17(5):849–869, 1988.

[10] A. Baltz and A. Srivastav. Approximation algorithms for the Euclidean bipartite TSP. *Operations Research Letters*, 33(4):403 – 410, 2005.

[11] X. Bao and Z. Liu. An improved approximation algorithm for the clustered traveling salesman problem. *Information Processing Letters*, 112(23):908–910, 2012.

[12] R. E. Bixby and W. H. Cunningham. Matroid optimization and algorithms, handbook of combinatorics (vol. 1), 1990. Technical Report 90-15, Rice University.

[13] P. Chalasani and R. Motwani. Approximating capacitated routing and delivery problems. *SIAM Journal on Computing*, 28(6):2133–2149, 1999.

[14] P. Chalasani, R. Motwani, and A. Rao. Algorithms for robot grasp and delivery. In *Proceedings of 2nd International Workshop on Algorithmic Foundations of Robotics*, pages 347–362, 1996.

[15] G. Chartrand and L. Lesniak. *Graphs and Digraphs. 1996.* Wadsworth & Brooks/Cole, Monterey, CA, 1996.

[16] N. Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical Report 388, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, 1976.

[17] W. J. Cook. *In Pursuit of the Traveling Salesman: Mathematics at the Limits of Computation.* Princeton University Press, 2012.

[18] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms.* The MIT Press, 2001.

[19] A. Frank. A weighed matroid intersection algorithm. *Journal of Algorithms*, 2:328–336, 1981.

[20] A. Frank. *Connections in Combinatorial Optimization.* Oxford Lecture Series in Mathematics and Its Applications. OUP Oxford, 2011.

[21] A. Frank, E. Triesch, B. Korte, and J. Vygen. On the bipartite traveling salesman problem. Technical Report 98866-OR, University of Bonn, 1998.

[22] G. N. Frederickson, M. S. Hecht, and C. E. Kim. Approximation algorithms for some routing problems. In *17th Annual Symposium on Foundations of Computer Science*, pages 216–227, IEEE, 1976.

[23] A. M. Frieze, G. Galbiati, and F. Maffioli. On the worst-case performance of some algorithms for the asymmetric traveling salesman problem. *Networks*, 12(1):23–39, 1982.

[24] T. Fukunaga and H. Nagamochi. Some theorems on detachments preserving local-edge-connectivity. *Electronic Notes in Discrete Mathematics*, 24:173–180, 2006.

[25] T. Fukunaga and H. Nagamochi. Eulerian detachments with local edge-connectivity. *Discrete Applied Mathematics*, 157(4):691–698, 2009.

[26] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

[27] G. Gordon and J. McNulty. *Matroids: A Geometric Introduction*. Cambridge University Press, 2012.

[28] N. Guttmann-Beck, R. Hassin, S. Khuller, and B. Raghavachari. Approximation algorithms with bounded performance guarantees for the clustered traveling salesman problem. *Algorithmica*, 28(4):422–437, 2000.

[29] J. Hoogeveen. Analysis of Christofides' heuristic: Some paths are more difficult than cycles. *Operations Research Letters*, 10:291–295, 1991.

[30] Y. Karuno, H. Nagamochi, and A. Shurbevski. Approximation algorithms for a cyclic routing problem of grasp-and-delivery robots. In *Proceedings of Joint 5th International Conference on Soft Computing and Intelligent Systems and 11th International Symposium on Advanced Intelligent Systems (SCIS & ISIS 2010)*, pages 94–99, 2010.

[31] Y. Karuno, H. Nagamochi, and A. Shurbevski. Approximating cyclic routing problems of grasp-and-delivery robots in production of printed circuit boards. In *Proceedings of International Symposium on Scheduling 2011 (ISS 2011), JSME No. 11-205*, pages 247–252, 2011.

[32] Y. Karuno, H. Nagamochi, and A. Shurbevski. An approximation algorithm with factor two for a repetitive routing problem of grasp-and-delivery robots. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 15(8):1103–1108, 2011.

[33] Y. Karuno, H. Nagamochi, and A. Shurbevski. Constant factor approximation algorithms for repetitive routing problems of grasp-and-delivery robots in production of printed circuit boards. *Journal of the Operations Research Society of Japan*, 55(3):181–191, 2012.

[34] N. Katoh and T. Yano. An approximation algorithm for the pickup and delivery vehicle routing problem on trees. *Discrete Applied Mathematics*, 154(16):2335–2349, 2006.

[35] J. Kleinberg and E. Tardos. *Algorithm Design*. Pearson Education India, 2006.

[36] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms, 3rd ed.* Springer, 2005.

[37] M. Krishnamoorthy. An NP-hard problem in bipartite graphs. *SIGACT News*, 7:26–26, 1975.

[38] M. A. Langston. A study of composite heuristic algorithms. *The Journal of the Operational Research Society*, 38(6):539–544, 1987.

[39] E. L. Lawler. *Combinatorial Optimization: Networks and Matroids*. Dover, 2001.

[40] W. Malik, S. Rathinam, and S. Darbha. An approximation algorithm for a symmetric generalized multiple depot, multiple travelling salesman problem. *Operations Research Letters*, 35(6):747–753, 2007.

[41] C. Moore and S. Mertens. *The Nature of Computation*. Oxford University Press, 2011.

[42] H. Nagamochi and T. Ibaraki. *Algorithmic aspects of graph connectivity*. Number 123. Cambridge University Press, New York, 2008.

[43] C. S. J. Nash-Williams. Connected detachments of graphs and generalized Euler trails. *Journal of the London Mathematical Society*, 2(1):17–29, 1985.

[44] J. G. Oxley. *Matroid Theory*. Oxford University Press, 1992.

[45] S. Rathinam and R. Sengupta. Lower and upper bounds for a multiple depot UAV routing problem. In *45th IEEE Conference on Decision and Control*, pages 5287–5292, IEEE, 2006.

[46] S. Rathinam and R. Sengupta. 3/2-approximationn algorithm for two variants of a 2-depot Hamiltonian path problem. *Operations Research Letters*, 38(1):63–68, 2010.

[47] A. Recski. Matroids-the engineers' revenge. In *Research Trends in Combinatorial Optimization*, pages 387–398. Springer Berlin Heidelberg, 2009.

[48] A. Recski. Engineering applications of matroids-a survey. In *Matroid Theory and its Applications*, pages 300–321. Springer Berlin Heidelberg, 2011.

[49] S. Sahni and T. Gonzalez. P-complete approximation problems. *Journal of the ACM*, 23(3):555–565, 1976.

[50] A. Schrijver. *Combinatorial Optimization - Polyhedra and Efficiency*. Springer, 2003.

[51] A. Sebö. Eight-fifth approximation for the path TSP. In M. X. Goemans and J. R. Correa, editors, *IPCO*, volume 7801 of *Lecture Notes in Computer Science*, pages 362–374. Springer, 2013.

[52] A. Shurbevski. Approximating repetitive routing problems of grasp-and-delivery robots. Master's thesis, Kyoto Institute of Technology, 2012.

[53] A. Shurbevski, Y. Karuno, and H. Nagamochi. Improved implementation of an approximation algorithm with factor two for a cyclic routing problem of grasp-and-delivery robots. In *Proceedings of International Symposium on Scheduling 2011 (ISS 2011), JSME No. 11-205*, pages 235–240, 2011.

[54] A. Shurbevski, H. Nagamochi, and Y. Karuno. Heuristics for a repetitive routing problem of a single grasp-and-delivery robot with an asymmetric edge cost function. In *10th International Conference of the Society for Electronics, Telecommunications, Automatics and Informatics (ETAI 2011)*, CD-ROM Proceedings, paper A1-1, 2011.

[55] A. Shurbevski, Y. Karuno, and H. Nagamochi. A dynamic programming based improvement heuristic for a repetitive routing problem of grasp-and-delivery robots. *Journal of Advanced Mechanical Design, Systems, and Manufacturing, Special Issue on ISS 2011: Advanced Production Scheduling*, 6 (5):611–621, 2012.

[56] A. Shurbevski, H. Nagamochi, and Y. Karuno. Improved approximation ratio algorithms for grasp-and-delivery robot routing problems. In *The 15th Japan-Korea Joint Workshop on Algorithms and Computation (WAAC 2012)*, pages 9–16, 2012.

[57] A. Shurbevski, H. Nagamochi, and Y. Karuno. The repetitive routing problem revisited. In *11th International Conference of the Society for Electronics, Telecommunications, Automatics and Informatics (ETAI 2013)*, CD-ROM Proceedings, paper A1-2, 2013.

[58] A. Shurbevski, H. Nagamochi, and Y. Karuno. Better approximation algorithms for grasp-and-delivery robot routing problems. *IEICE Transactions on Information and Systems*, E96-D(3):450–456, 2013.

[59] A. Shurbevski, H. Nagamochi, and Y. Karuno. Approximating the bipartite TSP and its biased generalization. In S. P. Pal and K. Sadakane, editors, *WALCOM 2014*, volume 8344 of *Lecture Notes in Computer Science*, pages 56–67, Springer International Publishing, 2014.

[60] A. Srivastav, H. Schroeter, and C. Michel. Approximation algorithms for pick-and-place robots. *Annals of Operations Research*, 107(3):321–338, 2001.

[61] V. V. Vazirani. *Approximation Algorithms*. Springer, 2001.

[62] J. Vygen. New approximation algorithms for the TSP. *Optima: Mathematical Optimization Society Newsletter*, 90:1–12, 2012.

[63] H. Whitney. On the abstract properties of linear dependence. *American Journal of Mathematics*, 57(3):509–533, 1935.

[64] D. P. Williamson and D. B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.

[65] A. Zavichi, K. Madani, P. Xanthopoulos, and A. A. Oloufa. TSP-based model for on-site material handling operations with tower cranes. In *30th International Symposium on Automation and Robotics in the Construction, Mining and Petroleum Industries (ISARC 2013)*, pages 1288–1295, 2013.

List of the Author's Work

Journal Publications

[1] Aleksandar Shurbevski, Yoshiyuki Karuno, and Hiroshi Nagamochi. A dynamic programming based improvement heuristic for a repetitive routing problem of grasp-and-delivery robots. *Journal of Advanced Mechanical Design, Systems, and Manufacturing, Special Issue on ISS 2011: Advanced Production Scheduling*, 6(5):611–621, 2012. doi:10.1299/jamdsm.6.611

Note: Part of Chapter 4 (edited August, 2014), including some of the results presented in Section 4.2.1, are based on results which originally appear in this article.

[2] Yoshiyuki Karuno, Hiroshi Nagamochi, and Aleksandar Shurbevski. Constant factor approximation algorithms for repetitive routing problems of grasp-and-delivery robots in production of printed circuit boards. *Journal of the Operations Research Society of Japan*, 55(3):181–191, 2012.

Note: Part of Chapter 5 (edited August, 2014), including some of the results presented in Section 5.3, are based on results which originally appear in this article.

[3] Aleksandar Shurbevski, Hiroshi Nagamochi, and Yoshiyuki Karuno. Better approximation algorithms for grasp-and-delivery robot routing problems. *IEICE Transactions on Information and Systems*, E96-D(3):450–456, 2013. doi:10.1587/transinf.E96.D.450

Note: Part of Chapter 5 (edited August, 2014), including some of the results presented in Section 5.3.1, are based on results which originally appear in this article.

Refereed Conference Proceedings

[4] Aleksandar Shurbevski, Hiroshi Nagamochi, and Yoshiyuki Karuno. Approximating the bipartite TSP and its biased generalization. In Sudebkumar Prasant Pal and Kunihiko Sadakane, editors, *WALCOM 2014*, volume 8344 of *Lecture Notes in Computer Science*, pages 56–67, Springer International Publishing, 2014. doi:10.1007/978-3-319-04657-0_8

Note: Part of Chapter 3 (edited August, 2014), including some of the results presented in Section 3.4, are based on results which originally appear in this article.

[5] Aleksandar Shurbevski, Hiroshi Nagamochi, and Yoshiyuki Karuno. The repetitive routing problem revisited. In *11th International Conference of the Society for Electronics, Telecommunications, Automatics and Informatics (ETAI 2013)*, CD-ROM Proceedings, paper A1-2, 2013.
Note: Part of Chapter 3 (edited August, 2014), including some of the results presented in Section 3.4, are based on results which originally appear in this article.

Conference Proceedings without Peer Review

[6] Aleksandar Shurbevski, Hiroshi Nagamochi, and Yoshiyuki Karuno. Improved approximation ratio algorithms for grasp-and-delivery robot routing problems. In *The 15th Japan-Korea Joint Workshop on Algorithms and Computation (WAAC 2012)*, pages 9–16, 2012.
Note: Part of Chapter 5 (edited August, 2014), including some of the results presented in Section 5.3.1, are based on results which originally appear in this article.