FernUniversität Hagen
Fakultät für Mathematik und Informatik

Bachelorarbeit im Fach Mathematik

Balancing Circular Stations of a Self Service Bike Hire System

Bearbeitungsbeginn 17. Juni 2015

Dr. Dipl.-Wirt.-Ing. David A. Wuttke
Matrikelnummer: 8673942

1. Prüfer: Prof. Dr. Winfried Hochstättler

Contents

List of Figures

1 Introduction

Self-service bike hire systems gain increasing attention in both theory and practice. Consider for instance the service "call a bike", offered by the Deutsche Bahn AG subsidiary DB Rent GmbH. Customers of *call a bike* can hire bikes, pick them up at a station of their choice and return them at another station of their choice. This enables customers to reach their short-distance destinations on-demand often even faster than through public transportation. Hiring bikes is thus considered a promising solution to the last-mile-transportation problem [8], a sustainable means for urban mobility, and socially equitable [16]. As a consequence, many major cities world wide provide self-service bike hire systems, for instance Vélib in Paris [2], Capital Bikeshare in Washington, DC, [17], Hubway in Boston, MA, [17], or Barclays Cycle Hire in London [15].

In spite of the autonomy of users, providers must take care of balancing such systems. Indeed, when users systematically move bikes away from certain stations to others, the provider must ensure both that there are always sufficient bikes where demanded and that there are sufficient free racks where customers want to drop their bikes. Typically trucks are used for that matter. The increasing popularity of bike hire systems leads to significant sizes. Vélib, for instance, operates more than 20,000 bikes in over 1,200 stations [4]. Therefore, balancing stations efficiently is an important and large challenge.

In recent publications, this problem is commonly modeled through graphs [2, 16, 17]. Consider a graph $G = (V, E)$ whose vertices are stations and edges are routes that a truck may use in order to move bikes between stations. Moving the truck along edge $e \in E$ costs $c_e \in \mathbb{R}_+$. Each station v has an initial number of bikes denoted by x_v and a desired number of bikes denoted by y_v. Moving trucks typically occurs during the night when no bikes are moved by customers [2, 9]. Each unbalanced station can be either in default ($x_v < y_v$) or excess ($x_v > y_v$). Denote the start station of the truck by p and the terminal station by q.

The objective of the STATIC STATIONS BALANCING PROBLEM (SSBP) is to find a sequence of moves such that the total costs of balancing is minimal [2]. Static refers to

the assumption of customers not moving any bikes during ongoing balancing operations. In general, this problem is **NP**-hard as it comprises the traveling salesmen problem as a special case. Benchimol et al. [2] study the SSBP and provide several heuristics. Interestingly, they find that the SSBP is solvable in linear time when G is a tree. Central to their algorithm is that trees are cycle free and thus there is always only one path connecting two vertices. Therefore their algorithm is independent of c_e ($\forall e \in E$). If G contains cycles, however, this is does not hold because there are at least two vertices connected by two distinct paths such that even when moving to an adjacent station costs of either route need to be considered. Benchimol et al. [2] state that the computational complexity class of G being a cycle is still open.

The special case of G being a cycle is interesting for three reasons. First, geographically this may relate to stations around park or a lake, for instance around Central Park in New York City, NY, or Lake Harriet in Minneapolis, MN. Second, to model stations as a line relies on the assumption that the costs between both end stations are sufficiently high such that no truck would ever use another path than the path modeled by the line. By studying the cycle, some theoretical thresholds can be obtained which allow to quantify *sufficiently high*. Finally, many graphs have subgraphs which are cycles. Thus understanding cycles better may be a step towards providing better solutions to general graphs.

1.1 Objectives and results

The objective of is this bachelor thesis is to derive structural insights on the problem of balancing circular stations of a self-service bike hire system, that is, when G is a cycle. This includes both characterizing the optimal solution to the circular balancing problem as well as identifying additional requirements which allow to solve the circular balancing problem efficiently. Research questions answered in this thesis include: If the truck exclusively drove in one direction, how often would it need to circle the entire cycle? What is a lower bound on balancing costs of a circular system when a certain edge must be used? And when can the circular balancing problem be solved efficiently by adapting the tree algorithm by Benchimol el al. [2]?

The thesis provides several results. First, for each edge there is one direction in which bikes never need to be shipped in the optimal solution, which can be either clockwise or anti-clockwise. Second, for the case that a truck balances stations by simple moving forward an upper bound on the number of times it needs to circle around the entire cycle is provided. This already provides a first upper bound on the balancing costs. This thesis also provide a lower bound on balancing costs if the truck uses certain edges. Third, the algorithm to balance a tree in [2] is applied to the case of a line and a further upper bound on the costs of balancing a cycle is provided. Finally, this upper bound is compared to the aforementioned lower bound to decide when treating the cycle as line and omitting one edge is optimal. Several examples are provided to illustrate structural insights.

1.2 Definitions and CSSBP model

Some related basic concepts of graph theory will be defined next before describing the model.

1.2.1 Related graph theory concepts

Based on Gross et al. [11],

Definition 1 (graph) *A graph $G = (V, E)$ consists of two sets V and E. The elements of V are called vertices and the elements of E are called edges. Each edge has a set of one or two vertices associated to it.*

Vertices will also be referred to as *stations* interchangeably. The following notation is adapted from [2]. For any subset of vertices U, denote by $\delta(U)$ the set of edges having exactly one endpoint in U and if $U = \{v\}$ write $\delta(v)$ instead of $\delta(\{v\})$. Related to graphs are subgraphs which can be defined according to Gross et al. [11] as follows,

Definition 2 (subgraph) *A subgraph of a graph $G = (V_G, E_G)$ is a graph $H = (V_H, E_H)$ such that $V_H \subset V_G$ and $E_H \subset E_G$. If H is a subgraph of G write $H \subset G$.*

A concept that is implicitly of importance in several statements in this thesis is that of isomorphism as defined by [18].

Definition 3 (isomorphism) *Two graphs $G = (V, E)$ and $H = (W, F)$ are called isomorph, if there is a bijection $\phi : V \longrightarrow W$ such that for all $v, w \in V$ the number of edges between v and w in G are the same as the number of edges between $\phi(v)$ and $\phi(w)$ in H.*

An immediate consequence is that indices of vertices and edges can be changed to simplify the analysis because it is sufficient to solve a given problem on some isomorph graph since one could use the inverse of bijection ϕ to map the optimal sequence to the original problem in linear time.

There are particularly three special graphs of interest, namely lines, cycles, and trees. Following the terminology of [2] the term line is used whereas others refer to these graphs as *path graphs* [11, 18].

Definition 4 (line) *A graph G is called a line if it consists of a vertex set $\{1, \ldots, n\}$ and edge set $\{\{1, 2\}, \{2, 3\}, \ldots, \{n-1, n\}\}$.*

By connecting both ends of a line, a cycle is obtained. More formally, based on [18],

Definition 5 (cycle) *A graph G is called a cycle if it consists of a vertex set $\{1, \ldots, n\}$ and edge set $\{\{1, 2\}, \{2, 3\}, \ldots, \{n-1, n\}, \{n, 1\}\}$.*

Therefore, each cycle has some subgraphs which are lines. Based on [18] define,

Definition 6 (tree) *A graph G is called a tree if there is no subgraph $H \subset G$ that is a cycle and if G is connected, which means that there is a path between each pair of vertices.*

1.2.2 CSSBP model

Now turn the attention to definitions specifically related to the static station balancing problem of a self-service bike hire system expressed by a graph $G = (V, E)$. The following formal definitions are adapted from [2].

Definition 7 (state) *Let $G = (V, E)$ be a graph. A state s is a couple $(\mathbf{x}, p)$ where $\mathbf{x} \in \mathbb{R}_+^V$ and p a vertex in V.*

Moreover, the following definition is particularly useful for shorthand notations.

Definition 8 (sets of balanced stations) *Let $G = (V, E)$ be a graph and let $U \subset V$. Define $x(U) := \sum_{v \in U} x_v$ and $y(U) := \sum_{v \in U} y_v$. Define the set of balanced stations, $B(\mathbf{x}, \mathbf{y})$, as $B(\mathbf{x}, \mathbf{y}) = \{v \in V | x_y = y_v\}$. If $x(U) = y(U)$ call the set U balanced.*

Note that $u \in U \subset V$ and U balanced is not sufficient to conclude $u \in B(\mathbf{x}, \mathbf{y})$. For a truck with capacity C, the states $s = (\mathbf{x}, p)$ and $s' = (\mathbf{y}, q)$ are called adjacent if simultaneously $x_y = y_v$ for all $v \notin \{p, q\}$, $pq \in E$, $x_p - y_p = y_q - x_q$ and $|x_p - y_p| \leq C$. While a move is the transition from a state to an adjacent state with costs $c(pq)$, a sequence of moves consists of several moves.

In this thesis the CIRCULAR STATIC STATIONS BALANCING PROBLEM (CSSBP) is studied formulated as follows.

Instance: A cycle $G = (V, E)$, a cost function $c \in \mathbb{R}_+^E$, a capacity C and two states $i = (\mathbf{x}, p)$ and $t = (\mathbf{y}, q)$.

Task: Find the minimal cost of a sequence of moves that allows to go from the state i to the state t and the first move of such an optimal sequence.

In most of the analysis it will be assumed that all stations are unbalanced which does not restrict much generality, because whenever there are unbalanced stations which are neither p nor q, one could simply remove them and add up the costs of the edges to form a new edge and close the cycle again. Moreover, it is assumed that it is always possible to drop all bikes from the truck at any station, which Benchimol et al. [2] call the preemptive version. An important assumption throughout this thesis is $x(V) = y(V)$. If this did not hold, the CSSBP would be infeasible.

1.3 Plan

Related literature is reviewed in the Section 2. In particular, the relationship to the work by Benchimol et al. [2] will be elaborated on by showing how their results differ from and relate to the special case of circular stations. In Section 3, several properties that characterize the optimal solution of the CSSBP will be derived. In Section 4, conditions will be derived under which the CSSBP can be solved in polynomial time. Finally, results are discussed in Section 5.

2 Relation to previous work

Station balancing problems are increasingly gaining importance in literature. In this section, first a review of some recent work will be provided before turning to a paper by Benchimol et al. in depth, as this is closest related to this thesis. While not all publications related to station balancing problems can be referenced in this review, please refer to [15] for a good and recent overview.

2.1 Related balancing problems

The problem of balancing stations of bike hire systems is closely related to the one-commodity pickup-and-delivery traveling salesman problem (1-PDTSP) as introduced by Hernández-Pérez and Salazar-González and studied in [12, 13]. This problem, in turn, is a generalization of the traveling salesmen problem so that a truck in the 1-PDTSP must visit each station exactly once and each station is either a pick-up customer or a delivery customer. Also, the truck has some capacity limit. Clearly, even without capacity limit this problem is **NP**-hard since it generalizes the traveling salesmen problem. Hernández-Pérez and Salazar-González [13] provide a 0-1 formulation and a branch-and-cut procedure for deriving solutions.

Chemla et al. [5] study a related version, the single vehicle one-commodity capacitated pick-up and delivery problem. They first provide an intractable mixed integer program formulation before providing some structural insights for better understanding decisions in

balancing problems. Chemla et al. [4] focus a dynamic version of the balancing problem, namely one where the truck seeks to balance the stations during day time so that customers may move bikes before the balancing is completed. They show that this problem is **NP**-hard and provide solution heuristics. They also derive a pricing scheme seeking to incentivize customers to choose stations that reduce balancing costs.

Erdoğan et al. [9] study a static bicycle relocation problem with demand intervals closely related to [12]. However, in their case any amount of bikes per station between a lower and an upper bound is sufficient, which is easier and usually less expensive than an exact amount. They provide an integer formulation and derive several valid inequalities to strengthen it. An interesting version of the static bike balancing problem is considered by Raviv et al. [16], whose objective is not only to reduce balancing costs but also to minimize dissatisfaction of users.

Finally, Schuijbroek et al. [17] consider an integrated problem related to the static stations balancing problem. Their problem is to identify the optimal service level at each station such that neither stock-outs occur to often nor that there are too many bikes at each station available leaving insufficient free racks available for customers who want to return their bikes. Besides the focus on the service level, Schuijbroek et al. [17] also take the cost of balancing into account. They provide a heuristic and apply it to data obtained from bike hire systems in two US east coast cities to show the effectiveness.

While the problems addressed by the aforementioned literature are related and focus the balancing problem in general, all authors concur that related versions of the static balancing problem are **NP**-hard and, therefore, for arbitrary networks no efficient solutions are known. However, the special case of circular stations, which may warrant efficient solutions at least under certain conditions, has so far not been addressed to the best of the author's knowledge.

2.2 Benchimol et al.: Balancing the stations of a self-service bike hire system

The work closest related to this thesis is Benchimol et al. [2]. They study the problem of balancing the stations of a self-service bike hire system on a graph $G = (V, E)$ which is not restricted to be a circle. They contribute several interesting results. In the remainder of this section, these results and their relationship to the special case of a cycle are discussed. Benchimol et al. [2] formulate the following research version of the SSBP.

Instance: A graph $G = (V, E)$, a cost function $c \in \mathbb{R}_+^E$, a capacity C and two states $i = (\mathbf{x}, p)$ and $t = (\mathbf{y}, q)$.

Task: Find the minimal cost of a sequence of moves that allows to go from the state i to the state t and the first move of such an optimal sequence.

Benchimol et al. [2] first provide insights into the complexity of the SSBP. Since the seminal work of [6], many problems have been shown to be **NP**-hard, particularly by means of reducing an **NP**-complete problem to the new problem. Indeed, Benchimol et al. [2] proceed in a similar fashion. Since this thesis is concerned with a special case of their SSBP, it seems appropriate to review their proofs in depth and decide whether the same **NP**-complete problems could be reduced to the CSSBP. As Arora and Barak [1] observe, this often turns out to be a fruitful approach to study the complexity of restricted instances. Benchimol et al.'s main complexity result is that the SSBP is **NP**-hard even for a complete graph with unit costs and a $\mathbf{y}$ constant on V. The idea of their proof is to reduce the PARTITION PROBLEM, which is known to be an **NP**-complete problem [10], to the SSBP in polynomial time. They construct a capacity constraint C and initial distribution of bikes $\mathbf{x}$ based on the input parameters of the PARTITION problem such that, if and only if the answer to PARTITION is yes, the optimal solution to the SSBP requires exactly $n + 2$ moves. While this proof works well for complete graphs, this logic cannot be related to the CSSBP because it strictly requires a complete graph. If cycles are balanced within $n + 2$ moves, the truck moved about once along each of the $n - 1$ edges. Attempting to relate this to PARTITION, one would need to know upfront which vertices should be adjacent in the cycle. This, however, is the essence of the PARTITION

problem itself and thus there is no straight polynomial time reduction.

A further complexity result in [2] is that the SSBP is **NP**-hard even for a bipartite graph with unit costs, fixed C (even $C = 1$) and $y_v = 1$ for all $v \in V$. This time Benchimol et al. show how the HAMILTONIAN CYCLE IN BIPARTITE GRAPHS PROBLEM, also a known **NP**-complete problem [10], can be reduced in polynomial time. The decision problem of the HAMILTONIAN CYCLE IN BIPARTITE GRAPHS PROBLEM is whether there is a simple circuit that includes all vertices in G [10]. The core idea of the proof by [2] is to use the color classes A and B for the bipartite graph and to have $x_v = 2$ if $v \in A$ and $x_v = 0$ else while $p = q$ is any vertex in A and $y_v = 1$ for all $v \in V$. Now, it is clear to see that if and only if the optimal solution to the SSBP is n there exists a Hamiltonian Cycle. While the CSSBP is concerned with cycles and cycles with an even number of vertices are bipartite graphs [18], the question of whether there exists a Hamiltonian Cycle within a cycle is trivial. At the flip side, the HAMILTONIAN CYCLE IN BIPARTITE GRAPHS PROBLEM, in general, cannot be reduced to the CSSBP in polynomial time to decide whether CSSBP is **NP**-hard.

Finally, for the case that G is a tree and the preemptive version is considered, [2] provide a solution in linear time implying that this case is in the complexity class **P**. However, by the definition of trees, trees are cycle free [18]. Therefore, a possible driver of the complexity of general cases of the SSBP are cycles. In relation to the complexity of cycles, Benchimol et al. [2] state that "the complexity status of this case is open" (p. 17).

Besides complexity results, Benchimol et al. [2] also suggest a series of algorithms and heuristics for the SSBP. For the general case they adapt an algorithm by Chalasani and Motwani [3] and show that it provides a 9.5-approximation algorithm. For complete graphs with unit costs they show that a greedy algorithm provides a 2-approximation algorithm. While these results are of general interest, they are not further summarized here in detail because the case of a tree with the special case of lines is closer related to and reflected in some aspects within this thesis.

2.2.1 The case of a tree

Benchimol et al. [2] provide an efficient algorithm for the case of a tree. Key idea of this algorithm is to consider the connected components $K_1, \ldots, K_h$ of $G - \{p\}$. Since G is a tree, there are no edges between either pair of K_i and K_j $(i \neq j)$. The algorithm is depicted in Figure 1. In order to provide an expression for the optimal costs, several definitions and notations are required, which are partly directly adopted from [2]. First, for an edge $e \in E$ let U_e denote the subset of vertices such that $\delta(U_e) = e$ and $x(U_e) \geq y(U_e)$. In cases of equality, make arbitrary choices.

(1) All stations are balanced.

  (a) $p \neq q$
  move along the edge stemming from p in direction of q.

  (b) $p = q$
  don't move (it is finished).

(2) There are unbalanced stations.

  (a) There is a component K_i in excess.
  take no bike and enter the component K_i in excess.

  (b) There is no a component in excess.

    (i) There are several components K_i such that we have simultaneously $x(K_i) \leq y(K_i)$ and at least one unbalanced station in K_i.
    Choose such a K_i such that $q \notin K_i$, take $\min(C, y(K_i) - x(K_i))$ bikes, enter K_i and put the bikes on the first vertex of the component.

    (ii) There is only one components K_i such that we have simultaneously $x(K_i) \leq y(K_i)$ and at least one unbalanced station in K_i.
    Take $\min(C, y(K_i) - x(K_i))$ bikes, enter K_i and put the bikes on the first vertex of the component.

Figure 1: Rule to balance stations of a tree. Adopted from Benchimol et al. [2].

A useful notation according to [2] is

$$\mu\left(p,q,U_e,\mathbf{x},\mathbf{y}\right):=\begin{cases}0 & \text{if } p,q\in U_e \text{ and } \overline{U}_e\subset B\left(\mathbf{x},\mathbf{y}\right)\\ 0 & \text{if } p,q\in \overline{U}_e \text{ and } U_e\subset B\left(\mathbf{x},\mathbf{y}\right)\\ 1 & \text{if } p\in U_e \text{ and } q\in \overline{U}_e\\ 1 & \text{if } p\in \overline{U}_e \text{ and } q\in U_e\\ 2 & \text{if } p,q\in U_e \text{ and } \overline{U}_e\setminus B\left(\mathbf{x},\mathbf{y}\right)\neq\emptyset\\ 2 & \text{if } p,q\in \overline{U}_e \text{ and } U_e\setminus B\left(\mathbf{x},\mathbf{y}\right)\neq\emptyset\,,\end{cases} \tag{1}$$

which relates to the minimal amount edge e is used and transcends into the subtour elimination constraint of the related traveling salesmen problem. Indeed, $\mu\left(p,q,U_e,\mathbf{x},\mathbf{y}\right)$ states that edge e has to be traversed at least twice if both p and q are on one side of e but some stations on the other side require balancing. In case p and q are on different sides of e, e needs to be traversed at least once. Finally, only if p and q are on the same side and all stations on the other side are balanced, it is possible never to use edge e. Note that these numbers are independent of capacity. Furthermore define

$$\eta\left(p,q,U_e\right):=\begin{cases}-1 & \text{if } p\in U_e \text{ and } q\in \overline{U}_e\\ 0 & \text{if } p\in U_e \text{ and } q\in U_e\\ 0 & \text{if } p\in \overline{U}_e \text{ and } q\in \overline{U}_e\\ 1 & \text{if } p\in \overline{U}_e \text{ and } q\in U_e\,.\end{cases} \tag{2}$$

The function $\eta\left(p,q,U_e\right)$ serves as a correction to account for the location of the truck when calculating how often an edge needs to be traversed under capacity considerations. Finally, define

$$z_e\left(p,q,\mathbf{x},\mathbf{y}\right):=\max\left(2\left\lceil\frac{x\left(U_e\right)-y\left(U_e\right)}{C}\right\rceil+\eta\left(p,q,U_e\right),\mu\left(p,q,U_e,\mathbf{x},\mathbf{y}\right)\right) \tag{3}$$

Then Benchimol et al. [2] derive and prove the following result.

Proposition 1 (Theorem 5 in [2]) *The rule described above provides a first move of an optimal sequence for the SSBP on a tree (and hence, if repeated, provides an optimal sequence) and the optimal cost is:*

$$\sum_{e \in E} c_e z_e \left(p, q, \mathbf{x}, \mathbf{y}\right) \tag{4}$$

This result merits two interesting insights. Not only is a myopic algorithm possible, but also is it independent of the actual costs of each edge (cf. Figure 1, where decisions never depend on costs). Benchimol et al. prove this result in several steps. First, they provide an integer linear program which provides a lower bound on balancing all stations. Essentially this integer linear program states that $\sum_{e \in E} c_e z_e \left(p, q, \mathbf{x}, \mathbf{y}\right)$ already provides a lower bound. The main part of the proof is then to show that, indeed, the algorithm in Figure 1 leads to the correct amount $z_e \left(p, q, \mathbf{x}, \mathbf{y}\right)$ of using each edge.

This is done by induction on $\gamma \left(p, q, \mathbf{x}, \mathbf{y}\right) := \sum_{e \in E} z_e \left(p, q, \mathbf{x}, \mathbf{y}\right)$. The first step is clear, if $\gamma \left(p, q, \mathbf{x}, \mathbf{y}\right) = 0$ then the algorithm is in step 1b and Equation (3) leads correctly to $z_e \left(p, q, \mathbf{x}, \mathbf{y}\right) = 0$.

In the induction step it is assumed that $\gamma \left(p, q, \mathbf{x}, \mathbf{y}\right) > 0$ has been correct so far and that a move towards state $(\mathbf{x}', p')$ is the next move according to their algorithm. Then it needs to be shown that using Equation (3) leads to

$$z_e \left(p, q, \mathbf{x}, \mathbf{y}\right) = \begin{cases} 1 + z_e \left(p', q, \mathbf{x}', \mathbf{y}\right) & \text{if } e = pp' \\ z_e \left(p', q, \mathbf{x}', \mathbf{y}\right) & \text{if not} \end{cases} \tag{5}$$

because this would imply that each increase of $\gamma \left(p, q, \mathbf{x}, \mathbf{y}\right)$ by one is due to increasing the right z_e through the algorithm and by induction proves their result. In order to show that Equation (5) actually results from applying the algorithm, Benchimol et al. show its validity through all cases. For the sake of brevity, this is not repeated here but rather case 1a is shown as an example. Indeed, in case 1a, $\mathbf{x} = \mathbf{y}$ because no bikes are moved anymore. For any edge $e \neq pp'$ clearly $z_e \left(p, q, \mathbf{x}, \mathbf{x}\right) = z_e \left(p', q, \mathbf{x}, \mathbf{x}\right)$ because moving from p to p'

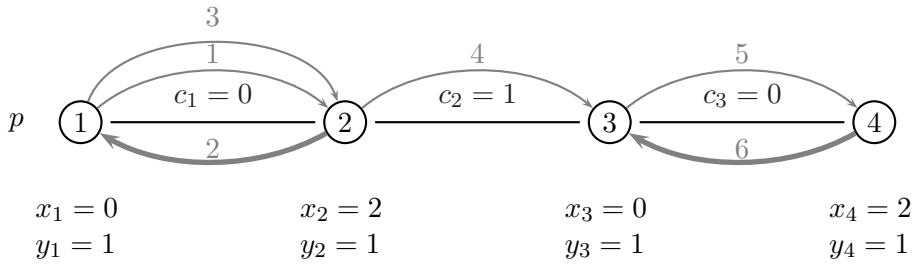
affects neither μ nor η. Moreover, for $e = pp'$, $\eta(p, q, U_e) = 1$, $\mu(p, q, U_e, \mathbf{x}, \mathbf{x}) = 1$ and $\eta(p', q, U_e) = 0$, $\mu(p', q, U_e, \mathbf{x}, \mathbf{x}) = 0$ and hence $z_e(p, q, \mathbf{x}, \mathbf{x}) = 1$ and $z_e(p', q, \mathbf{x}, \mathbf{x}) = 0$ satisfying Equation (5).

Equation (3) can also be interpreted in a more intuitive (and less formalized) way. The idea here is that $\mu(p, q, U_e, \mathbf{x}, \mathbf{y})$ provides a lower bound on the amount of required passes through e. Now, the term $2\left\lceil \frac{x(U_e)-y(U_e)}{C} \right\rceil$ reflects how often the truck must pass through the edge because of capacity constraints. The factor 2 stems from the fact that the truck always needs an inbound and an outbound move. Yet, this term must be corrected by the term $\eta(p, q, U_e)$ which is 1 if $p \in \overline{U}_e$, that is the truck first needs to pass e to enter U_e, and $q \in U_e$, that is the trucks needs to finish in U_e. If $p \in U_e$, that is the truck is already in U_e, and $q \in \overline{U}_e$, that is outside of U_e, the truck requires 1 time less traversing edge e.

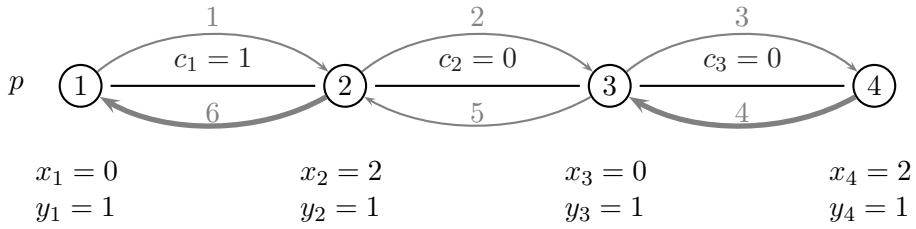
2.2.2 The case of a line

A line is a special case of a tree [18]. Therefore, it is possible to apply the aforementioned algorithm on lines, too. Benchimol et al. [2] provide a different approach for the case of a line when q is not fixed. However, the description appears ambiguous. Indeed, the algorithm seems to suggest that neither costs nor the potential location of q need to be taken into account. Consider the line in Figure 2a. An optimal solution is the sequence depicted through gray arrows, where only in moves 2 and 6 a bike is transported, leading to costs of 1. This is optimal because clearly edge 2 has to be traversed at least once to balance stations 3 and 4 which suggests a lower bound of 1. Whether the truck ends at station 3 or 4 is inconsequential because of $c_3 = 0$. Note that given the costs it cannot be optimal to end at station 1 or 2, because this would lead to minimal costs of 2.

Now consider the line in Figure 2b. The initial task is the same but costs are slightly different. In this case, an optimal solution (again depicted in gray arrows) requires the truck to transport a single bike only in moves 4 and 6 and the truck terminates at station 1. Note that edge 1 has to be traversed at least twice to balance station 1, hence 2 is a lower bound. Moreover, note that any $q \neq 1$ implies traversing edge 1 three times, increasing the costs to 3. Therefore, both figures illustrate that costs need to be taken



(a) optimal $q \in \{3, 4\}$.



(b) optimal $q = 1$.

Figure 2: Illustration of minimal costs on a line with free q with unit capacity. Bold lines indicate when a bike is transported.

into consideration when q is free because otherwise the same sequence would be suggested leading to higher than optimal costs at least in one case. Indeed, the problem with free q can still be solved in polynomial time, for instance by applying the tree algorithm for each of the n stations as candidates for q and picking the cheapest one. In this thesis, a simple formulation for the case of the line is required when q is fixed. Therefore, the tree algorithm reviewed above will be adjusted to the case of a line instead of the line algorithm in [2]. This adjustment is described in Section 4.2.

3 Characterization of the optimal CSSBP solution

Even though the difference between a line and cycle is only manifested through a single edge, the solution can differ significantly. For instance, a truck may require to circle several times the cycle. This way it could traverse some edges consecutively in the same direction; this is impossible on a line. Other than in the tree where only one path exists

between each pair of vertices, in the case of the cycle there are always two paths such that costs have to be taken into account even when visiting an adjacent station. To get some first insight, one way of formulating the CSSBP is to formulate it as dynamic program [7, 14] which primarily serves here as illustration of the size of the associated state space. Consider cycle $G = (V, E)$ where stations are such that $p-1$ is left of p and $p+1$ is right of p. Then a cost-to-go function for the state $(\mathbf{x}, p)$ is given by

$$J(\mathbf{x}, p) := \min \Big\{ c_{pp+1} + \min_{\substack{0 \le k \le C \\ k \le x_p}} \{ J((x_1, \ldots, x_p - k, x_{p+1} + k, \ldots, x_n), p+1) \}, \\ c_{p-1p} + \min_{\substack{0 \le k \le C \\ k \le x_p}} \{ J((x_1, \ldots, x_{p-1} + k, x_p - k, \ldots, x_n), p-1) \} \Big\}$$

with $J(\mathbf{y}, q) = 0$. This formulation captures the intuition of a truck that has to take two decisions on each station: how many of the x_p bikes to load and whether to move them left or right. Even only the latter choice indicates 2^N possible decisions for a fixed sequence length N. In addition, the number of bikes to be transported leads to prohibitively large solution spaces because technically up to $\sum_{v \in V} x_v$ bikes could be put onto each of the stations. Solving the dynamic program thus appears too tedious to provide a meaningful answer to the question of the first move and the costs of the optimal solution. Nevertheless, this does not imply that nothing can be learned about the optimal sequence.

Rather than providing an efficient algorithm, the objective of this section is to derive some characteristics of the optimal solution to gain structural insights. Answers will be provided to questions such as: Will it be required for an optimal solution to sometimes ship bikes along some edge e first in one direction but then other bikes along e into the reversed direction or may some bikes be unloaded during the first pass without adding additional costs? If the solution requires strictly more bikes to be shipped anti-clockwise along a certain edge than clockwise, does this prescribe how many bikes have to be moved along other edges? In other words, if, in the optimal solution, one must ship a certain number of bikes along a specific edge, does this require the use of a specific other edge, too? Moreover, the following scheme to balance all stations would always work: Always

load as many bikes as possible as long as $x_p > y_p$ and unload as many bikes as possible as long as $x_p < y_p$. Then simply move the truck forward, always in the same direction. If such as scheme is followed, how often does the truck need to circle the cycle and what are the maximal costs? Besides providing structural insights by answering these questions, several results will be derived in this section which are required later on.

3.1 Clockwise and anti-clockwise shipments

The first result observes the shipments of bikes along each edge.

Proposition 2 *In the CSSBP, there exists an optimal sequence of moves such that each edge has one direction (clockwise or anti-clockwise) in which the truck never ships bikes.*

Proof (Proposition 2). Certainly, there is always at least one optimal sequence of moves to balance the stations of a given cycle in the preemptive version of the CSSBP. This follows immediately from $\sum_{v \in V} x_v = \sum_{v \in V} y_v$. Let this sequence be denoted by S. If according to S each edge has one direction (clockwise or anti-clockwise) in which the truck never ships bikes, then it is already done. If not, there exist at least two adjacent stations p and q and two pairs of adjacent states $s = (\mathbf{x}, p)$ and $t = (\mathbf{y}, q)$ as well as $s' = (\mathbf{x}', p)$ and $t' = (\mathbf{y}', q)$ such that there is a move from s to t and later on from t' to s' with $\mathbf{x} \neq \mathbf{y}$ and $\mathbf{x}' \neq \mathbf{y}'$. Let the edge between p and q be denoted by e and let $N := x_p - y_p$ and $M := x'_q - y'_q$ be the number of bikes shipped in the moves from s to t and t' to s', respectively. Without loss of generality, define the direction from s to t to be clockwise. Assume that, physically, no bike shipped from p to q will be among the bikes shipped back later on from q to p. Otherwise, simple drop those bikes at p and adjust N and M accordingly. This is possible because the preemptive version is considered here.

Let sequence S be expressed by the states $s_1, \ldots, s_n$ with $s_i = (\mathbf{x}_i, p_i)$ $(i = 1, \ldots, n)$. In this proof a similar sequence S' will be constructed which consists of the states $s'_1, \ldots, s'_n$ such that $s'_i = (\mathbf{x}'_i, p_i)$ $(i = 1, \ldots, n)$, which means the truck takes the same tour as in S leading to the same costs but may differ in terms of load. It will be shown that S' will satisfy $(\mathbf{x}_i)_{p_i} - (\mathbf{x}_{i+1})_{p_i} \geq (\mathbf{x}'_i)_{p_i} - \left(\mathbf{x}'_{i+1}\right)_{p_i}$ for all $i = 1, \ldots, n-1$ and in particular

that the truck is either empty when moving from s to t or from t' to s'. By induction on the edges where bikes are shipped in both directions, an optimal sequence is constructed which ensures that each edge has one direction (clockwise or anti-clockwise) in which the truck never ships bikes.

Let $K := \min\{M, N\}$. To construct S', let $s'_i = s_i$ for all states i reached before state s. Now, instead of transporting N bikes in the move from s to t, drop immediately K bikes at station p. Index these bikes by $k = 1, \ldots, K$ and denote their destination under S, that is the station where they are supposed to be dropped according to S, by $\omega_1, \ldots, \omega_K$. Likewise, note that the truck would later on move bikes indexed by $K+1, \ldots, 2K$ from t' to s'. Denote the origin stations of these bikes as $\alpha_{K+1}, \ldots, \alpha_{2K}$.

Since the truck first moves along e clockwise and later anti-clockwise, there must be a station where it reverses its direction. Let r be the station with the greatest clockwise distance to p that the truck reaches before returning. Then $\alpha_{K+1}, \ldots, \alpha_{2K}$ and $\omega_1, \ldots, \omega_K$ are between q and r or they are either q or r. Now, pick any pair of bikes, bike k and bike $K+k$, and adjust the loading and dropping procedure of S as follows. If $\alpha_{K+k} = \omega_k$ do nothing. If the truck reaches α_{K+k} before reaching ω_k on its way to towards r, load bike $K+k$ at station α_{K+k} and unload it at station ω_k. By this approach, the truck has loaded one less bike while moving from p to q (and vice-versa) and also on its way from q to α_{K+k} while otherwise the load is the same as under S. The latter aspect ensures $(\mathbf{x}_i)_{p_i} - (\mathbf{x}_{i+1})_{p_i} \geq (\mathbf{x}'_i)_{p_i} - \left(\mathbf{x}'_{i+1}\right)_{p_i}$. Yet this leads to the same bike distribution as under S. If, on the other hand station α_{K+k} would be reached after ω_k, the truck passes by α_{K+k} and on its return from station r at the last time it visits station α_{K+k} it should load the bike $K+k$ and drop it at ω_k. Again, by this approach, the truck has loaded one less bike while moving from p to q (and vice-versa) and on its way from q to ω_k while never having more bikes than according to S. This, again, ensures $(\mathbf{x}_i)_{p_i} - (\mathbf{x}_{i+1})_{p_i} \geq (\mathbf{x}'_i)_{p_i} - \left(\mathbf{x}'_{i+1}\right)_{p_i}$. Again this leads to the same bike distribution as under S. Now, repeat this procedure for the remaining $K-1$ pairs of bikes. Then the number of bikes moved between p and q has been reduced by K and the load of the truck has not been increased for any move. Since $K = \min\{M, N\}$, either the truck moves empty from s to t or from t' to s'. Finally, let

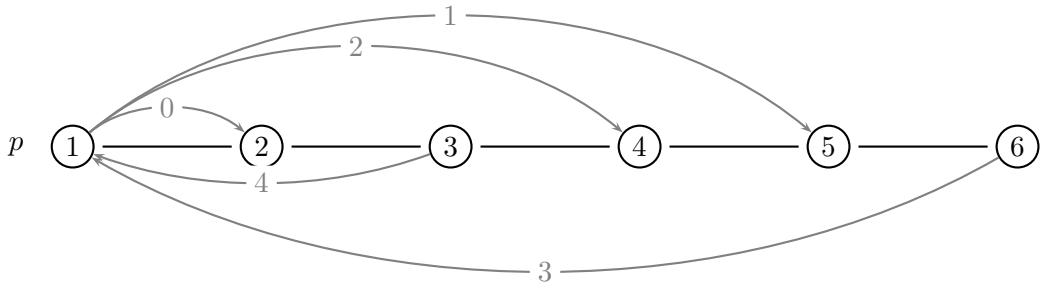
$s'_i = s_i$ for all states i reached after state s'.

Now, repeat this procedure as long as there is an edge along which the truck moves bikes clockwise and anti-clockwise. Note that each application of the procedure reduces the occasions of bikes being shipped in both directions strictly by one and since $(\mathbf{x}_i)_{p_i} - (\mathbf{x}_{i+1})_{p_i} \geq (\mathbf{x}'_i)_{p_i} - \left(\mathbf{x}'_{i+1}\right)_{p_i}$ it will never create a sequence S' where a bike on another edge $e' \neq e$ is shipped that is not shipped according to S. □

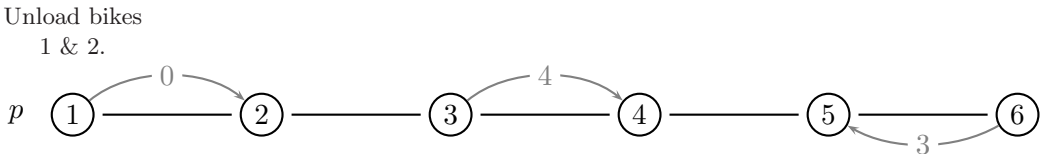
Figures 3a and 3b illustrate the construction described in the proof. In Figure 3a, a sequence is depicted that requires moving bikes along the edge (12) into two directions, which is changed in Figure 3b where only one bike is moved along edge (12) and thus only in one direction. This is the result of the aforementioned procedure, to be more precise, with $K = \min(M, N) = 2$ there are two pairs of bikes, namely (1,3) and (2,4). Bike 4 should be shipped while the truck moves from p towards $r = 6$, bike 3 while the truck returns from r towards p. The illustration shows that the number of bikes are on each edge is reduced while the same truck movements are required. Intuitively this captures the idea to move bikes shorter distances if possible.

3.2 Required use of special edges

Proposition 2 states that along each edge bikes only need to be moved clockwise or anti-clockwise but this Proposition only looks at edges in isolation. *Ceteris paribus*, it is thus far not clear whether the decision to ship some bikes along a certain edge e in at least one direction may limit the decision whether to ship bikes along certain other edges e'. In a complete graph, for instance, this is hardly the case since there are always enough alternatives. But what about the cycle? Does the use of certain edges only qualify to decide whether adjacent edges have to be used or does it warrant more implications? To capture some structural insights about the cycle, several definitions are required first. The aim is to make use of certain patterns induced by the vectors $\mathbf{x}$ and $\mathbf{y}$.



(a) Transporting bikes in two directions along (12), $N = 3, M = 2$.



(b) Transporting bikes in one direction along (12) trough revised scheme, $N = 3, M = 2$.

Figure 3: Ilustration of proof of Proposition 2. Effectively, both sequences lead to the same bike distributions and same costs.

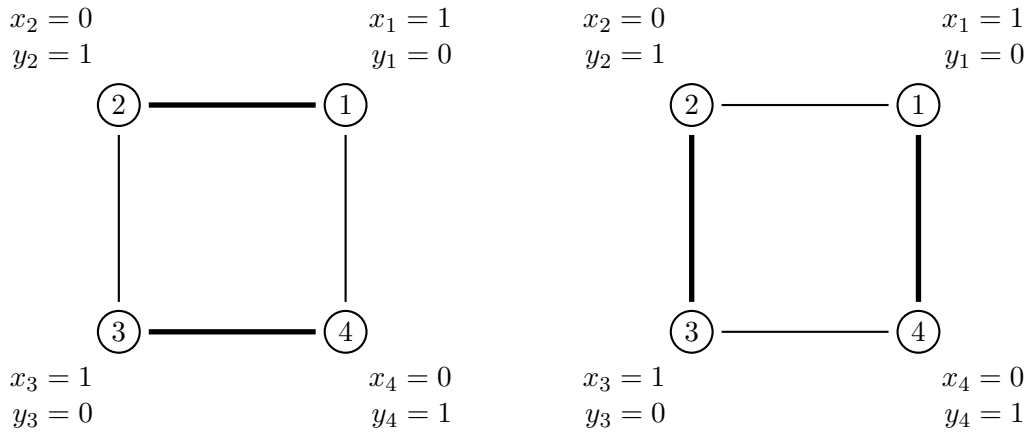
Definition 9 (balanced line) *Let line $L = (V_L, E_L)$ be a subgraph of cycle $G = (V, E)$. L is called balanced if $\sum_{v \in V_L} (x_v - y_v) = 0$.*

Clearly, only if a line is balanced, a truck could balance each station of this line without either adding or removing any bikes from the line.

Definition 10 (partition) *Let $G = (V, E)$ be a cycle and let $L_i = (V_i, E_i) \subset G$ be balanced lines where $i \in \{1, \ldots, m\}$ and $m \geq 2$. If $V = \cup_{i=1}^{m} V_i$, and $V_i \cap V_j = \varnothing$ for $i \neq j$, then call $P := \{L_1, \ldots, L_m\}$ a partition of balanced lines.*

There can be many different partitions for the same cycle. Particularly, if there is at least one balanced station, there is at least one partition.

Definition 11 (set of connecting edges) *Consider cycle $G = (V, E)$ with partition $P = \{L_1, \ldots, L_m\}$ and balanced lines $L_i = (V_i, E_i) \subset G$, $i \in \{1, \ldots, m\}$. The set of connecting edges is defined by $W_P := E \setminus \cup_{i=1}^{m} E_i$.*



(a) Partition P in bold lines.

(b) Partition P' in bold lines.

Figure 4: Consider a truck that moves straight from 1 to 4 and moves a bike from 1 to 2 and one from 3 to 4. It balances P-local but strictly P'-global.

And finally, a distinction between different types of balancing schemes can be made.

Definition 12 (P-local and P-global balancing) *Consider graph G with partition P. The truck balances all balanced lines P-locally if the truck is empty whenever it moves along an edge $e \in W_P$. Otherwise the truck balances the stations P-globally. The truck strictly balances all balanced lines P-globally if there exists an edge $e \in W_P$ such that the total number of bikes moved clockwise and those moved anti-clockwise along e are different.*

P-local balancing thus implies that each bike remains in the same balanced line, hence the term *local*. Note that the reference to the partition is made explicit, because, given a partition P, balancing might occur locally whereas through the same moves and the same cycle but a different partition, this would be considered global balancing. Figure 4 illustrates this.

The result of Proposition 2 can now be expressed in terms of the type of balancing used.

Corollary 1 *If there is a cycle $G = (V, E)$ endowed with Partition P such that all stations are balanced P-globally but not strictly P-globally, then the same solution can be obtained by P-local balancing.*

Proof (Corollary 1). If G is not balanced strictly P-global, then there is no $e \in W_P$ along which more bikes are shipped into one direction than reverse. Therefore, for all $e \in W_P$ either no bikes are shipped at all or the same number is shipped reversely. By Proposition 2 it is possible to adjust the sequence such that through those edges bikes are shipped only in one direction. Since the same number of bikes would have been shipped in both directions, no bikes are shipped along each $e \in W_P$ which is local balancing. □

Now the initially raised questions as to whether the use of certain edges requires the use of certain other edges can be addressed.

Proposition 3 *Consider cycle G with partition P and a set of connecting edges W_P. If the truck balances the stations of G strictly P-globally it must move along each $e \in W_P$ at least once.*

Proof (Proposition 3). Let $P = \{1, \ldots, L_m\}$. Since the truck balances strictly P-global, there exists an edge $e \in W_P$ such that the difference between the total number of bikes shipped clockwise and the total number of bikes shipped anti-clockwise $\neq 0$. Let $e = pq$ and let Δ_{pq} be the total number of bikes shipped from p to q minus the total number of bikes shipped from q to p. Since $e \in W_P$, p and q are in different lines. Let these be denoted by L_p and L_q, respectively. If $\Delta_{pq} > 0$ then L_q would be in excess, which means that at least one station is not balanced, unless Δ_{pq} bikes are shipped to its other adjacent line. Therefore, Δ_{pq} have to be shipped to its other adjacent line to ensure balancing L_q. By induction on the number of lines, Δ_{pq} bikes have to be shipped until they finally arrive at line L_p. If, instead, $\Delta_{pq} < 0$, the same logic applies but in the other direction of the cycle. Therefore, in both cases the truck has to move along each $e \in W_P$ at least once to ship Δ_{pq} bikes along each of such edges. □

Note that Proposition 3 holds for any Partition P. Hence, it can be iterated through all partitions to come up with sets of W_P. Then, whenever a solution would require to use an $e \in W_P$ for some P, this indicates how many bikes have to be shipped along the remaining edges in W_P. In a certain sense, partitions are sets of subsets, which leads to

the question as to how many partitions there are – whether the number of partitions grows exponentially or linearly – and how they can be identified.

Definition 13 *Let $P = \{L_1, \ldots, L_m\}$ and $P' = \{L'_1, \ldots, L'_{m'}\}$ be two different partitions of cycle G. If $\forall i' \in \{1, \ldots, m'\}$ $\exists i \in \{1, \ldots, m\}$ satisfying $V'_{i'} \subset V_i$, then P' is called a refinement of P.*

Now, of primary interest are non-refineable partitions, which are partitions for which no refinement exists, because if P' is a refinement of P, $W_P \subset W_{P'}$ follows from the definition. Hence the implications of Proposition 3 are strongest when P is non-refineable.

Lemma 1 *Cycle $G = (V, E)$ has at most $|V|$ partitions which cannot be further refined.*

Proof (Lemma 1). This claim can be proven constructively. Indeed, enumerate all vertices in G by $1, \ldots, |V|$ clockwise. Then start constructing partition P_1 by moving from vertex 1 clockwise. Include all vertices into the first line until there is a vertex i which satisfies $\sum_{v=1}^{i} (x_v - y_v) = 0$. If $i < |V|$, include all vertices starting by $i + 1$ into line 2 until reaching a vertex j that satisfies $\sum_{v=i+1}^{j} (x_v - y_v) = 0$ and continue by repeating this until all vertices are in balanced lines and the first partition is completed. Else ($i = |V|$), P_1 is not a partition, simply drop it and move on. Next, construct P_2 through the same routine starting at vertex 2. Continue until the starting vertex is $|V|$, afterwards at most $|V|$ partitions have been created.

Now the created partitions cannot be further refined. Assume the opposite, that is, there was a partition that could be further refined. Then there would be a line from some vertex v_1 to vertex v_k which can be split into two lines such that $\sum_{v=v_1}^{v_l} (x_v - y_v) = 0$ and $\sum_{v=v_{l+1}}^{v_k} (x_v - y_v) = 0$ where $v_1 < v_l < v_k$. However, this is impossible because by construction this first line from v_1 to v_l would have already been included in the partition and then the second as a separate line.

Indeed, this approach identifies all partitions which cannot be further refined. Assume the construction approach would be alternated in an attempt to find further partitions. Since all stations on each line must be connected, the only way to alternate the approach

is to start with some arbitrarily station v and add some stations to its left and some to its right, instead of strictly moving into one direction. However, let w be the left-most station added to the line including v. Then there is also a partition whose construction started with station w as obtained by the routine above. Therefore, the proposed construction leads to the construction of all possible partitions that cannot be further refined. $\square$

Based on this lemma additional constraints related to the use of edges in W_P for all of the at most $|V|$ partitions can be created. Furthermore, besides deriving restrictions on the use of certain special edges, Proposition 3 provides a first lower bound on the optimal solution. Indeed, under strict P-global balancing, costs are bounded from below by $\sum_{e \in W_P} c_e$.

3.3 Maximal number of complete circles

The optimal solution may require the truck to drive several times around the cycle. To capture this more formally, define,

Definition 14 (complete circle) *Let $G = (V, E)$ be a cycle with a current position of the truck $p \in V$. A complete circle is a series of moves between adjacent states such that the truck uses each edge exactly once and returns to p.*

An interesting characteristic of the optimal solution is the maximal number of complete cycles. In fact, a solution where the truck simply drives forward provides an intuitive upper bound. Economically there might be situations when an upper bound can be sufficient. Think of the question whether balancing a certain cycle is economically viable, then the upper bound on costs may already provide a positive answer. To formulate an upper bound on the number of complete circles a further definition is needed.

Definition 15 (connected vertices, end point) *Let $G = (V, E)$ be a cycle. $U \subset V$ is called a set of connected vertices if $\exists F \subset E$ such that (U, F) is a line. A vertex $u \in U$ where U is a connected set is called an end point of U if in G $\exists e \in E \setminus F$ such that $e \in \delta(u)$.*

Let $U^{\max}$ be a set of connected vertices which satisfies $x\left(U^{\max}\right)-y\left(U^{\max}\right) \geq x\left(U\right)-y\left(U\right)$ for all sets U of connected vertices. This relates to the path with the largest excess in supply. Since $x(V)=y(V)$, $x\left(V \setminus U^{\max}\right)-y\left(V \setminus U^{\max}\right)=-\left(x\left(U^{\max}\right)-y\left(U^{\max}\right)\right)$ which means that the aggregated default of the remaining stations is the same. Therefore, the larger $x\left(U^{\max}\right)-y\left(U^{\max}\right)$, the more the truck needs to balance. Hence $x\left(U^{\max}\right)-y\left(U^{\max}\right)$ is a measure of overall unbalancedness. The following proposition relates the overall unbalancedness with the upper bound on complete circles.

Proposition 4 *In the optimal solution to the CSSBP, the truck will never require more than* $\left\lceil \frac{x(U^{max})-y(U^{max})}{C} \right\rceil$ *complete circles.*

Proof (Proposition 4). Indeed, it is possible to construct a sequence such that the truck can balance all stations after at most $\left\lceil \frac{x(U^{\max})-y(U^{\max})}{C} \right\rceil$ complete circles. Any sequence with more complete circles than this will lead to strictly greater costs and is thus suboptimal. If all stations are balanced, that is $x(U^{\max})-y(U^{\max})=0$, simply move the truck on the cheapest route from p to q which requires $\left\lceil \frac{x(U^{\max})-y(U^{\max})}{C} \right\rceil = 0$ complete cycles and thus is in line with the proposition. Else, find all sets $U^{\max}$ such that $x(U^{\max})-y(U^{\max}) \geq x(U)-y(U)$ for all sets of connected stations U. Move the truck from p to the closest station p' which is an end point of some set $U^{\max}$. If there is a tie make a random choice. Now, the truck is at station p'. Let $m := \left\lceil \frac{x(U^{\max})-y(U^{\max})}{C} \right\rceil$. It will be shown that if $m>1$ and the truck completes a circle with appropriate loading and unloading of bikes, the initial status is almost the same except for when m is calculated it is reduced by 1. By the principle of induction this will be repeated $m-1$ times requiring the truck to complete $m-1$ circles until $\left\lceil \frac{x(U^{\max})-y(U^{\max})}{C} \right\rceil = 1$. In this case it can be shown that the truck can complete a final circle to balance all stations and then move to q. Adding the distance from p' to q to the distance from p to p' does not result in a complete circle, hence it is possible to balance all stations with at most m complete circles. Clearly, if m complete circles are enough and the path to p' and to q have been the shortest, it is not possible to find a better solution requiring more complete circles. So, what is left to prove is (a) if $m>1$ a complete circle with appropriate loading strategy reduces m by 1 and (b) if

$m = 1$ a complete circle leads to balancing all stations.

Consider first the latter case. When $m = 1$ then $x(U^{\max}) - y(U^{\max}) \leq C$. Now, index the stations beginning with $p' = 1$ such that the next station of $U^{\max}$ is 2 until the last station of the circle is n. Let z be the last station of $U^{\max}$ so that $\{1, \ldots, z\} = U^{\max}$ and $z + 1 \notin U^{\max}$. Then load $x_1 - y_1$ bikes into the truck and balance station 1. Clearly $x_1 - y_1 \geq 0$ because if instead $x_1 - y_1 < 0$ then $\sum_{i=2}^{z} (x_i - y_i) > \sum_{i=1}^{z} (x_i - y_i)$ which contradicts that $1 \in U^{\max}$. Also there is enough capacity since $x_1 - y_1 \leq C$ because if this was not the case, $U^{\max} = \{1\}$ with $m > 1$. Now, move straight towards z and for each station k do the following. If $x_k - y_k \geq 0$ load all bikes at this station and else unload $y_k - x_k$ bikes. In both ways all stations $1, \ldots, z$ will be balanced. Now it can be shown that this, indeed, is feasible given the capacity and non-negativity constraint.

At station $k = 1$ this was possible as shown before. Now assume it is also possible for stations $1, \ldots, k-1$. Case 1: $x_k - y_k \geq 0$. Assume adding $x_k - y_k$ lead to $\sum_{i=1}^{k} (x_i - y_i) > C$. This would imply $x(\{1, \ldots, k\}) - y(\{1, \ldots, k\}) > x(U^{\max}) - y(U^{\max})$, a contradiction. Hence the additional $x_k - y_k$ bikes can be loaded and station k can be balanced. Case 2: $x_k - y_k < 0$. Assume the truck has not loaded sufficient to unload $y_k - x_k$ bikes. Then $\sum_{i=1}^{k} (x_i - y_i) < 0$. But since $\sum_{i=1}^{k} (x_i - y_i) + \sum_{i=k+1}^{z} (x_i - y_i) = x(U^{\max}) - y(U^{\max})$ this would imply $x(\{k+1, \ldots, z\}) - y(\{k+1, \ldots, z\}) = \sum_{i=k+1}^{z} (x_i - y_i) > x(U^{\max}) - y(U^{\max})$, a contradiction to the definition of $U^{\max}$. Therefore, the truck can immediately balance station k by unloading $y_k - x_k$ bikes. By the principle of induction it follows that all stations $k = 1, \ldots, z$ can be balanced by a single forward transition from $k = 1$ to z. When the truck moves from z to $z + 1$ it has loaded $x(U^{\max}) - y(U^{\max})$ bikes.

Now, the truck will do the following. It passes straight on from station $z + 1$ to station n and at any station k it loads $x_k - y_k$ bikes if $x_k - y_k \geq 0$ and unloads $y_k - x_k$ bikes else. This will balance all stations $k = z + 1, \ldots, n$ with the truck terminating at station n such that a final move towards p' completes the circle. Now the actual feasibility of this procedure is shown.

Begin with station $z + 1$. Clearly, $x_{z+1} - y_{z+1} < 0$ because else $x(\{1, \ldots, z + 1\}) - y(\{1, \ldots, z + 1\}) > x(U^{\max}) - y(U^{\max})$, a contradiction. Moreover, since the entire cy-

cle can be balanced $0 = \sum_{i=1}^{n}(x_i - y_i) = \sum_{i=1}^{z}(x_i - y_i) + x_{z+1} - y_{z+1} + \sum_{i=z+2}^{n}(x_i - y_i) \Leftrightarrow y_{z+1} - x_{z+1} = \sum_{i=1}^{z}(x_i - y_i) + \sum_{i=z+2}^{n}(x_i - y_i)$. Recall that stations 1 and n are adjacent, hence $\sum_{i=z+2}^{n}(x_i - y_i) \leq 0$ because else $\sum_{i=1}^{z}(x_i - y_i) + \sum_{i=z+2}^{n}(x_i - y_i) > \sum_{i=1}^{z}(x_i - y_i)$, a contradiction to the construction of $U^{\max}$. Therefore $y_{z+1} - x_{z+1} \leq \sum_{i=1}^{z}(x_i - y_i)$, implying that $z+1$ can be balanced.

Now, assume that all stations up to station $k-1$, $z+1 \leq k-1 \leq n$ have been balanced. Then there are 2 cases related to station k. Case 1: $y_k - x_k > 0$. Recall $0 = \sum_{i=1}^{n}(x_i - y_i) = \sum_{i=1}^{z}(x_i - y_i) + \sum_{i=z+1}^{k-1}(x_i - y_i) + x_k - y_k + \sum_{i=k+1}^{n}(x_i - y_i) \Leftrightarrow y_k - x_k = \sum_{i=1}^{z}(x_i - y_i) + \sum_{i=z+1}^{k-1}(x_i - y_i) + \sum_{i=k+1}^{n}(x_i - y_i)$. By construction of $U^{\max}$, $\sum_{i=1}^{z}(x_i - y_i) \geq \sum_{i=1}^{z}(x_i - y_i) + \sum_{i=k+1}^{n}(x_i - y_i)$ hence $y_k - x_k \geq \sum_{i=1}^{z}(x_i - y_i) + \sum_{i=z+1}^{k-1}(x_i - y_i)$ and hence the demand of station k can be satisfied by unloading $y_k - x_k$ bikes. Case 2: $x_k - y_k \geq 0$. Since by definition of $U^{\max}$ it follows $\sum_{i=1}^{k}(x_i - y_i) \leq \sum_{i=1}^{z}(x_i - y_i) \leq C$ and it was possible to balance all stations until station $k-1$, there is sufficient space in the truck to load $x_k - y_k$ bikes and thus to balance station k. By the principle of induction this can be repeated until station n is reached and thus all stations are balanced. Therefore, if $m = 1$ the truck can drive around the entire cycle and balance all stations straight.

Now, the case of $m > 1$ is left to be shown and the truck is at the first station of $U^{\max}$. Index all stations in increasing order so that $p' = 1$ and $U^{\max} = \{1, \ldots, z\}$ as before. Again a procedure of loading and unloading bikes at the first z stations will be described and subsequently its feasibility proven and a reduction of m by 1. Move the truck straight from station 1 to station z. If at any station $k = 1, \ldots, z$ there is $x_k - y_k \geq 0$, load so many bikes that the truck is filled with $\min\left\{C, \sum_{i=1}^{k}(x_i - y_i)\right\}$ bikes. If $x_k - y_k < 0$ unload $\min\{y_k - x_k, C\}$ bikes. Then move on to the next station. When moving from station z to $z+1$, the truck is filled with C bikes.

Note that for station 1 this is feasible because $x_1 - y_1 \geq 0$ otherwise $1 \notin U^{\max}$. Now, assume some station k with $2 \leq k \leq z$ is reached and that the procedure was so far feasible. Then, the current load of the truck is $\min\left\{C, \sum_{i=1}^{k-1}(x_i - y_i)\right\}$. Again there are

two cases. Case 1: $x_k - y_k \geq 0$. Then simply try to load as many excess bike of station k as possible which results in in a truck filled with $\min\left\{C, \sum_{i=1}^{k}(x_i - y_i)\right\}$ bikes. Case 2: $x_k - y_k < 0$. Note that if $y_k - x_k < C$ the truck will always have sufficient bikes to balance this station straight analog to the aforementioned contradiction with the definition of $U^{\max}$. If $y_k - x_k > C$ this implies that currently C bikes are in the truck which can be unloaded. By induction it follows that this procedure is feasible for all following stations until z. Indeed, note that $x_z - y_z \geq 0$ because otherwise $z \notin U^{\max}$. Now, the truck is filled with $\min\left\{C, \sum_{i=1}^{z}(x_i - y_i)\right\} = \min\left\{C, x\left(U^{\max}\right) - y\left(U^{\max}\right)\right\} = C$ bikes. Since by moving from z to $z+1$ C bikes are moved out of $U^{\max}$, $x\left(U^{\max}\right) - y\left(U^{\max}\right)$ is decreased by C which causes $\left\lceil \frac{x(U^{\max}) - y(U^{\max})}{C} \right\rceil$ to decrease by 1.

So the truck arrives at station $z+1$ with C bikes. Now the balancing procedure is essentially analog to the case of $m = 1$ as discussed above. Therefore, whenever the truck hits a station in default it will unload bikes and whenever it hits a station in excess it will seek to load all bikes until it is filled up to capacity. What is left to show is that after completing its circle there is no set of adjacent stations U such that $\left\lceil \frac{x(U) - y(U)}{C} \right\rceil > m - 1$ and that, after completing the circle by moving from station n to 1, the truck is at a station p' which is an endpoint of some set of adjacent stations which satisfies the requirement of $U^{\max}$ so that the next induction step can continue.

Recall the situation before the induction step, $\left\lceil \frac{x(U^{\max}) - y(U^{\max})}{C} \right\rceil = m$. Assume there is another set $\widetilde{U}$ such that $\left\lceil \frac{x(\widetilde{U}) - y(\widetilde{U})}{C} \right\rceil = m$, that is $x(\widetilde{U}) - y(\widetilde{U}) \geq mC$. Define two integers a and b such that the stations of this set are $\widetilde{U} = \{a, a+1, \ldots, b\}$. Then there must be some stations separating the sets $U^{\max}$ and $\widetilde{U}$ because otherwise the set $U^{\max} \cup \widetilde{U}$ would contradict the definition of $U^{\max}$ in the first place. These stations are $z+1, \ldots, a-1$ and $b+1, \ldots, n$. Moreover, $\sum_{i=z+1}^{a-1}(x_i - y_i) \leq -mC$ and $\sum_{i=b+1}^{n}(x_i - y_i) \leq -mC$ because otherwise, still one could form a larger set comprising $U^{\max}$ and $\widetilde{U}$ contradicting the definition of $U^{\max}$. But due to the fact that $\sum_{i=z+1}^{a-1}(x_i - y_i) \leq -mC$ and that the truck will drop bikes whenever possible on its way from z to a, the truck will be empty when it arrives at station a. If this is not the case, it is simple to see that the set $\widetilde{U}$ could be extended such that the truck arrives empty and $x(\widetilde{U}) - y(\widetilde{U})$ is even larger. Now, exactly

the same routine applies to the stations of $\widetilde{U}$ that applied to $U^{\max}$ and thus the number of bikes of this set is reduced by C. Finally, there can be an arbitrary number of sets like $\widetilde{U}$. But by induction on the number of sets satisfying $x(\widetilde{U}) - y(\widetilde{U}) \geq mC$ it can be shown that for each set the same result holds and that after the truck has completed the round and the new $U^{\max}$ is observed, $\left\lceil \frac{x(U^{\max})-y(U^{\max})}{C} \right\rceil = m - 1$. Indeed, since the number of bikes removed from each set with the properties of $\widetilde{U}$ is the same, all stations that have been in the set $U^{\max}$ will also be in a set $U^{\max}$ defined based on the updated $\mathbf{x}$. In particular, there is a set $U^{\max}$ with endpoint p'. Therefore, the truck has completed a complete circle and by the logic of induction the statement of the proposition is true. □

Proposition 4 can be extended to provide an upper bound on the costs of the solution for any cycle.

Corollary 2 *Balancing costs of any cycle $G = (V, E)$ are bounded from above by* $\left(\left\lceil \frac{x(U^{max})-y(U^{max})}{C} \right\rceil + 1\right) \sum_{e \in E} c_e$.

Proof (Corollary 2). By Proposition 4, a truck needs at most $\left(\left\lceil \frac{x(U^{\max})-y(U^{\max})}{C} \right\rceil\right)$ complete circles causing costs of $\sum_{e \in E} c_e$ each. Thus it suffices to show that the costs of moving the truck from p to some endpoint p' of $U^{\max}$ and from p' to q is below $\sum_{e \in E} c_e$. Indeed, the truck may take the shortest path from p to p' causing costs $c_1 \leq \frac{1}{2} \sum_{e \in E} c_e$ (otherwise the other direction would have been cheaper). Now, q is between p and p' at least in one direction. The truck should take this route. By moving to q costs are $c_2 \leq \max\left\{\sum_{e \in E} c_e - c_1, c_1\right\}$, hence $c_1 + c_2 \leq \sum_{e \in E} c_e$. □

Recall the geographical motivation for considering cycles such as parks or lakes. In the case of Lake Harriet in Minneapolis, MN, for instance, streets are so that trucks should not turn around but move straight on instead.

Corollary 3 *If the truck is not allowed to turn around, balancing all stations requires no more than* $\left\lceil \frac{x(U^{max})-y(U^{max})}{C} \right\rceil + 1$ *complete circles for a fixed q and no more than* $\left\lceil \frac{x(U^{max})-y(U^{max})}{C} \right\rceil$ *complete circles for a non-fixed q.*

Proof (Corollary 3). Consider first the case where q is non-fixed. In this case, the truck starts at p and moves to the next station of $u \in U^{\max}$ such that u is an end point and such that the station $(u-1) \notin U^{\max}$. This requires less than one complete circle. The rest of the proof is analog to Proposition 4. Now, in the case of a fixed q the distance from p to u plus the distance from u to q may exceed an entire circle. The rest is analog to Proposition 4. □

4 Optimal solution of special CSSBP cases

For a given Cycle $G = (V, E)$ with Partition P Proposition 3 relates the shipment of bikes along any edge $e \in W_P$ with the use of all further edges $\tilde{e} \in W_P \setminus \{e\}$. This provides a starting point for a lower bound on a solution requiring strict P-global balancing. In this section, this bound first will be strengthened. Later, an algorithm to balance lines based on [2] will be formulated, which will be applied to the cycle to derive an upper bound on P-local balancing. The main result of this section is Proposition 6 which provides a sufficient condition for solving special cases of the CSSBP efficiently.

4.1 Lower bound on global balancing

Even though the optimal sequence under global balancing cannot be stated in a meaningful way, at least lower bound on its costs can be provided.

Lemma 2 *Consider a cycle $G = (V, E)$ without balanced stations with partition P. A lower bound on any solution with strict P-global balancing is given by*

$$LBG_P := \sum_{e \in E} c_e - \max_{e \in E \setminus W_P} \{c_e\} + \sum_{v \in V} \max \left\{ 2 \left(\left\lceil \frac{x_v - y_v}{C} \right\rceil - 1 \right), 0 \right\} \min_{e \in \delta(v)} \{c_e\}. \tag{6}$$

Proof (Lemma 2). Indeed, by Proposition 3 it is known that under strict P-global balancing each edge $e \in W_P$ must be traversed at least once. Moreover, since all stations are unbalanced initially each station must be visited at least once which requires $n - 1$

different edges to be traversed because there is no path with fewer edges that connects all vertices in a cycle [18]. So there is one edge $e \in E \setminus W_P$ which possible can be avoided reducing the lower bound by c_e.

In addition, if $x_v - y_v > C$, vertex $v \in V$ needs to be visited at least $\left(\left\lceil \frac{x_v - y_v}{C} \right\rceil - 1\right)$ additional times in order to remove all excess bikes. Visiting implies two moves, one ending in v and one departing from v. The truck could use either connected edge for that purpose, assuming the cheaper one is feasible (which is in line with the construction of a lower bound). This leads to additional marginal costs of $\min_{e \in \delta(v)} \{c_e\}$. Since each time v is visited, at least the cheaper edge has to be used twice, inbound and outbound, this leads to costs $\max\left\{2\left(\left\lceil \frac{x_v - y_v}{C} \right\rceil - 1\right), 0\right\} \min_{e \in \delta(v)} \{c_e\}$. Summing up both terms the lower bound of the lemma is obtained. □

4.2 Balancing lines

Cycles and lines are closely related because the removing of an edge transforms a cycle into a line. In this subsection an algorithm based on the tree algorithm by Benchimol et al. [2] will be provided. Consider a special case where the truck faces a line with q at the right end and stations numbered through $1, \ldots, q$. For the truck's current position, p, denote by $K_l := \{1, \ldots, p-1\}$ and $K_r := \{p+1, \ldots, q\}$ the stations left and right of the truck, respectively. Now the algorithm in Figure 5 is an application of the one in Figure 1. It is formulated slightly different because the truck has only two directions at each station where to move next to but does exactly the same. In (2b) the left component is always the one such that $q \notin K_l$, hence the right component is only visited by the truck in case (2bii) when the left component has been entirely balanced.

Similarly, Equation (3) can be simplified. Consider the situation where the truck starts at the left end and q is on the right. Enumerate the stations from $1, \ldots, q$. Index the edge between stations j and $j+1$ as j for all $j = 1, \ldots, q-1$. Then denote $U_e^l := \{1, \ldots, e\}$ and $U_e^r := \{e+1, \ldots, q\}$ and U_e is either U_e^l or U_e^r. This simplifies to $\mu(p, q, U_e, \mathbf{x}, \mathbf{y}) = 1$ because in Equation (1) p and q are always in opposed sets. In terms of the subtour elimination, $\mu(p, q, U_e, \mathbf{x}, \mathbf{y}) = 1$ implies that each edge has to be traversed at least once.

(1) All stations are balanced.

(a) $p \neq q$
move right.

(b) $p = q$
don't move (it is finished).

(2) There are unbalanced stations.

(a) $x(K_l) > y(K_l)$ or $x(K_r) > y(K_r)$
take no bike and if $x(K_l) > y(K_l)$ move left else move right.

(b) Both $x(K_r) \leq y(K_r)$ and $x(K_l) \leq y(K_l)$

(i) There is at least one unbalanced station left of p
Take $\min\left(C, y\left(K_l\right) - x\left(K_l\right)\right)$ bikes and put them on station $p-1$.

(ii) There is no unbalanced station left of p
Take $\min\left(C, y\left(K_r\right) - x\left(K_r\right)\right)$ bikes and put them on station $p+1$.

Figure 5: Rule to balance stations of a line. Adapted from Benchimol et al. [2] tree algorithm.

Moreover, define

$$\eta_e := \eta\left(p, q, U_e\right) = \begin{cases} -1 & \text{if } \sum_{i=1}^{e}\left(x_i - y_i\right) \geq 0 \\ 1 & \text{else} \end{cases} \tag{7}$$

With Equation (3) this leads to

$$z_e\left(p, q, \mathbf{x}, \mathbf{y}\right) = \max\left(2\left\lceil \frac{\left|\sum_{i=1}^{e}\left(x_i - y_i\right)\right|}{C} \right\rceil + \eta_e, 1\right) =: \widetilde{z}_e\left(\mathbf{x}, \mathbf{y}\right) \tag{8}$$

Now consider a balanced line $L = (V, E)$ with $\sum_{i=1}^{e}\left(x_i - y_i\right) \neq 0$ for all $e \in E$, hence $\left\lceil \frac{\left|\sum_{i=1}^{e}\left(x_i - y_i\right)\right|}{C} \right\rceil \geq 1$. This leads to,

Lemma 3 *The optimal costs of balancing a balanced line $L = (V, E)$ with $p = 1$ at one end and q at the opposed end of the line and with $\sum_{i=1}^{e} (x_i - y_i) \neq 0$ for all $e \in E$ is $\sum_{e \in E} c_e \widetilde{z}_e (\mathbf{x}, \mathbf{y})$, where $\widetilde{z}_e (\mathbf{x}, \mathbf{y}) = 2 \left\lceil \frac{\left| \sum_{i=1}^{e} (x_i - y_i) \right|}{C} \right\rceil + \eta_e$.*

Proof (Lemma 3). The proof follows by the discussion above and Proposition 1. □

Yet, the condition $\sum_{i=1}^{e} (x_i - y_i) \neq 0$, which allows to drop the term related to the subtour elimination constraint, $\mu (p, q, U_e, \mathbf{x}, \mathbf{y})$, requires some attention. Indeed, if one is not interested in a single line but rather in all lines of a partition, it would be quite tedious to test each time whether this condition holds. The concept of refinement becomes handy. In fact, if a partition is non-refineable, after adjusting some indices, for each line segment and each edge thereof, $\sum_{i=1}^{e} (x_i - y_i) \neq 0$ holds. Consider Cycle $G = (V, E)$ with non-refineable Partition P. For $L_i = (V_i, E_i) \in P$, $i = 1, \ldots, m$, with sets $V_i = \{1, \ldots, n_i\}$ and $E_i = \{1, \ldots, n_i - 1\}$ indexed such that edge j connects vertices j and $j+1$ ($j = 1, \ldots, n_i - 1$) denote by $\overline{c}_i := \sum_{e \in E_i} c_e \left(2 \left\lceil \frac{\left| \sum_{j=1}^{e} (x_j - y_j) \right|}{C} \right\rceil + \eta_e \right)$. Note that it is important to be careful when indexing the edges and vertices of each balanced line. In fact, the numbering must be done such that for each balanced line station q is connected through an edge $e \in W_P$ with station p of another balanced line. From now on, it is assumed that indexing is done correctly.

Corollary 4 *Let all stations of Cycle $G = (V, E)$ be unbalanced and let G be endowed with non-refineable Partition $P = \{L_1, \ldots, L_m\}$. If there is an edge $\tilde{e} \in W_P$ such that $\tilde{e} = pq$, balancing $H = (V, E \setminus \{\tilde{e}\})$ has the optimal solution $\sum_{i=1}^{m} \overline{c}_i + \sum_{e \in W_P \setminus \{\tilde{e}\}} c_e$ which can be achieved by applying the line balancing rule in Figure 5.*

Proof (Corollary 4). Indeed, $H = (V, E \setminus \{\tilde{e}\})$ is by definition a line with the truck initially on one end and q on the other end. Hence the line balancing rule in Figure 5 applies and Equation (8) provides the number of times each edge is traversed. Note that Lemma 3

cannot be applied immediately because for the edges $e \in W_P \setminus \{\tilde{e}\}$, $\sum_{i=1}^{e} (x_i - y_i) = 0$. For any $e \in W_P$ clearly $x\left(U_e^l\right) - y\left(U_e^l\right) = 0$ and $x\left(U_e^r\right) - y\left(U_e^r\right) = 0$ implying $\eta_e = -1$, hence $z_e = 1$ according to Equation (8). Now the problem of how often each edge within a line, L_i, is traversed can be reduced to the case of lines where Lemma 3 applies, because the truck starts empty each time at one end of line L_i $(i = 1, \ldots, m)$ and needs to terminate at the other end to move on towards q and because P is non-refineable. Applying Lemma 3 for each of the $i = 1, \ldots, m$ lines leads to costs of $\bar{c}_i$, thus in total $\sum_{i=1}^{m} \bar{c}_i + \sum_{e \in W_P \setminus \{\tilde{e}\}} c_e$. Finally this procedure leads to local balancing because the truck is always empty when moving along any edge $e \in W_P$. □

If there is a cycle G with Partition P such that there is $pq \in W_P$ then Corollary 4 provides an upper bound on P-local balancing costs on G because it provides a feasible solution. Note that this is not necessarily the best solution using P-local balancing because it might be better to use $\tilde{e}$ even with local balancing.

4.3 Optimality of local balancing

Lemma 2 provides a lower bound on balancing costs if strict P-global balancing is used. Corollary 4 provides an upper bound on P-local balancing. Combining both results,

Proposition 5 *Consider a cycle without balanced stations and endowed with non-refineable partition $P = \{L_1, \ldots, L_m\}$ and costs $\bar{c}_i$ for $i = 1, \ldots, m$. If there is an edge $\tilde{e} \in W_P$ such that $\tilde{e} = pq$ and*

$$LBG_P \geq \sum_{i=1}^{m} (\bar{c}_i) + \sum_{e \in W_P \setminus \{\tilde{e}\}} c_e \tag{9}$$

then P-local balancing is optimal.

Proof (Proposition 5). This proposition is a consequence of Lemma 2 and Corollary 4. Note that Lemma 2 refers to strict P-global balancing, not global balancing. However, from Corollary 1 it is known that if a solution can be obtained by global balancing which is not strict then it can also be obtained by P-local balancing which is thus optimal. □

The Inequality (9) allows for some structural insights into cases where P-local balancing is optimal. The costs of edge $\tilde{e}$ appear only on the left hand side, whereas other terms may appear on both sides. Hence, *ceteris paribus*, as $c_{\tilde{e}} \to \infty$ P-local balancing becomes more attractive. Also, the way capacity is accounted for on both sides of this inequality differs which may imply that certain capacity restrictions are necessary to show the optimality of local balancing. That is, facing the same cycle a truck with smaller capacity may clearly prefer local balancing whereas this does not follow for a truck with larger capacity.

To obtain a simpler result, assume the capacity is large enough such that it is not constraining, that is, assume $|\sum_{i=1}^{e}(x_i - y_i)| \leq C$ for all edges in all balanced lines and that $x_v - y_v \leq C$ for all $v \in V$. Then $\sum_{i=1}^{m}(\bar{c}_i) \leq 3 \sum_{e \in E \setminus \{W_P\}} c_e$ and $\max\left\{2\left(\left\lceil \frac{x_v - y_v}{C} \right\rceil - 1\right), 0\right\} = 0$. Then Inequality (9) is implied by

$$\sum_{e \in E} c_e - \max_{e \in E \setminus W_P}\{c_e\} \geq 3 \sum_{e \in E \setminus \{W_P\}} c_e + \sum_{e \in W_P \setminus \{\tilde{e}\}} c_e$$

$$\Leftrightarrow \; c_{\tilde{e}} - \max_{e \in E \setminus W_P}\{c_e\} \geq 2 \sum_{e \in E \setminus \{W_P\}} c_e$$

Now assume that all edges within the lines of P have unit costs except for $\tilde{e}$. With $n := |V|$ this equation simplifies to,

$$c_{\tilde{e}} - 1 \geq 2 \sum_{e \in E \setminus \{W_P\}} 1 \Leftrightarrow c_{\tilde{e}} \geq 2(n - m) + 1$$

Hence, $c_{\tilde{e}}$ would need to be larger when the circle is larger in terms of the number of stations or when there are fewer balanced lines in P, which provides further support for the idea of only considering non-refineable partitions in the first place. In other words, in a small cycle even relatively cheap $pq \in W_P$ is sufficient to indicate that local balancing is optimal. On the other hand, the factor 2 captures that by using $\tilde{e}$ the truck might reduce the number of each other edge within balanced lines of P to be traversed by 2.

4.4 Example

Consider the situation depicted in Figure 6 where bold lines indicate Partition P. Assume the truck has unit capacity. Using the aforementioned line algorithm by applying Corollary 4 and omitting edge 91 leads to $\sum_{i=1}^{m} \overline{c}_i + \sum_{e \in W_P \setminus \{\tilde{e}\}} c_e = \sum_{i=1}^{3} 4 + (5+5) = 22$. This is an upper bound on local balancing. Now, consider the term $\sum_{e \in E} c_e - \max_{e \in E \setminus W_P} \{c_e\} + \sum_{v \in V} \max\left\{2\left(\left\lceil \frac{x_v - y_v}{C} \right\rceil - 1\right), 0\right\} \min_{e \in \delta(v)} \{c_e\} = 21 - 1 + 6 = 26$. Hence, by Proposition 5, local balancing is optimal. Now, assume $C = 2$, then the lower bound on strict P-global balancing is only 20 so that Proposition 5 cannot be applied to decide whether local balancing is optimal. The above discussion indicated that with increasing costs of the edge to be omitted under local balancing local balancing gets more attractive. And, indeed, in the example in Figure 6 if $c_{91} = 7$ and $C = 2$ there is a lower bound on strict global balancing of 22 such that local balancing is optimal even with $C = 2$.

An interesting observation is thus that the benefit stemming from the use of a certain edge in strict global balancing may also depend on the capacity of the truck, not only on the cost of the edges.

4.5 Optimal solution for special case

Proposition 5 indicates whether local balancing is optimal without stating how this should be done. Under slightly stricter conditions, the aforementioned line balancing algorithm does not only provide a lower bound but is also optimal.

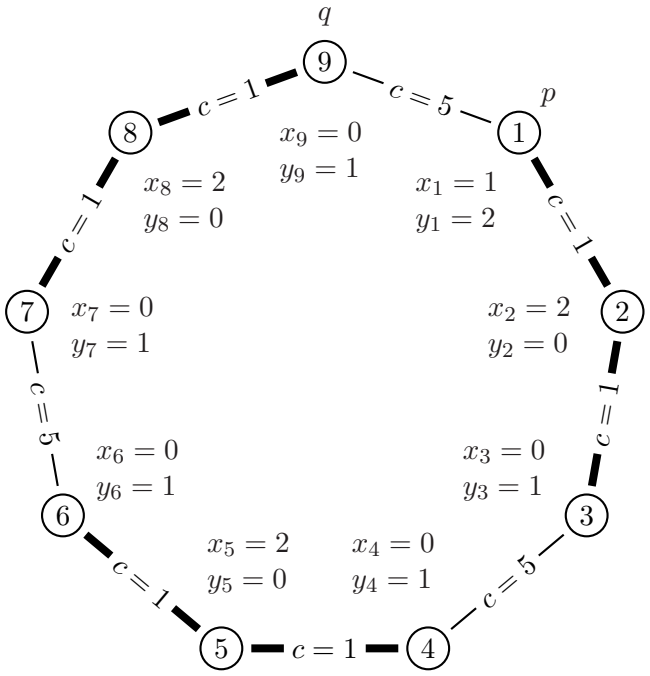


Figure 6: Example of unbalanced cycle. Bold lines represent Partition P.

Proposition 6 *Consider a cycle without any balanced station and with partition* $P = \{L_1, \ldots, L_m\}$ *and costs* $\bar{c}_i$ *for* $i = 1, \ldots, m$. *If there is an edge* $\tilde{e} \in W_P$ *such that* $\tilde{e} = (pq)$ *and*

$$\sum_{e \in E} c_e - \max_{e \in E \setminus \{\tilde{e}\}} \{c_e\} + \sum_{v \in V} \max \left\{ 2 \left(\left\lceil \frac{x_v - y_v}{C} \right\rceil - 1 \right), 0 \right\} \min_{e \in \delta(v)} \{c_e\}$$
$$\geq \sum_{i=1}^{m} (\bar{c}_i) + \sum_{e \in W_P \setminus \{\tilde{e}\}} c_e$$

then minimal costs are given by $\sum_{i=1}^{m} (\bar{c}_i) + \sum_{e \in W_P \setminus \{\tilde{e}\}} c_e$ *and the algorithm in Figure 5 provides the first step to the optimal sequence.*

Proof (Proposition 6). From Lemma 3 it is known that the term on the right hand side provides the costs of balancing the cycle when the algorithm in Figure 5 is applied for each line in the partition P. This algorithm treats the cycle as a line which means that edge $\tilde{e}$ is not allowed to be used and because of its optimality on the line, no better solution

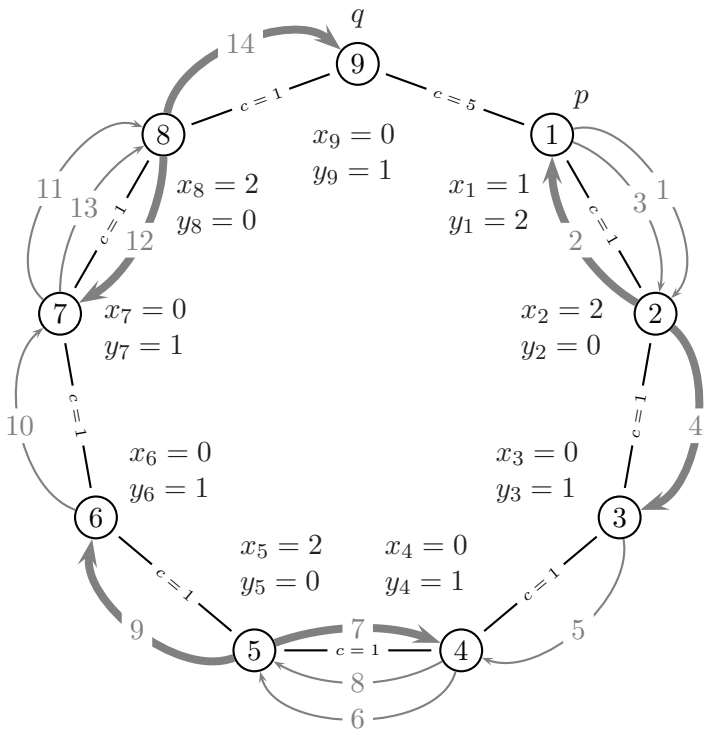


Figure 7: With $C = 1$ it is optimal to apply the line algorithm. Bold lines indicate when a bike is shipped, thin lines indicate empty moves. Numbers indicate the sequence.

can be found. Thus it only needs to be shown that no solution strictly using $\tilde{e}$ can lead to a better solution. The reasoning is essentially analog to Lemma 2. Indeed, except for one edge, which is not $\tilde{e}$, each edge must be used at least once to reach each unbalanced station. Furthermore, the capacity constraint might require the truck to use certain edges several times. This is entirely analog to Lemma 2. □

When applying Proposition 6 to the example in Figure 7 with $C = 1$, the line algorithm leads to an optimal sequence. The solution is illustrated in Figure 7.

5 Conclusion

This thesis was motivated by the increasingly popularity of bike hire systems and the arising problem of balancing their stations. Particularly, this thesis relates to the recent paper "balancing stations of a self sservice 'bike hire' system" by Benchimol et al. [2]. While Benchimol et al. examined general graphs, here the emphasis was on the special

case where all stations are circular, which was called the CSSBP.

This thesis provides some insights on the optimal solution. It compares systematically three balancing schemes, namely local and (strictly) global balancing. When a cycle is partitioned into balanced lines, the optimal solution could either result from local balancing, which does not allow to move bikes from one balanced line to another, or global balancing. The combination of strict global balancing and local balancing is shown to dominate non-strict global balancing. Furthermore, the truck could use several complete circles to balance all stations. One result of this thesis provides an upper bound on this number. Particularly for the case that the truck operates in a one way system, for instance around a lake in a recreational area, this result approximates the optimal costs quite well, especially when the terminal station is free and many complete circles are required.

Moreover, the objective was to find conditions under which the CSSBP can be solved by treating it as a line. In fact, if those conditions hold, the solution is the same as those where the cycle is opened up by removing a single edge. The conditions identified here indicate, that this edge would need to be more expensive than a certain threshold. Intuitively one may think this requires a kind of degenerate cycle $G = (V, E)$ such that there is an $\tilde{e} \in E$ with $c_{\tilde{e}} \geq \sum_{e \in E \setminus \{\tilde{e}\}} c_e$. However, an example showed that also with $c_{\tilde{e}} = \frac{5}{8} \sum_{e \in E \setminus \{\tilde{e}\}} c_e$ local balancing and treating the cycle as a line can be optimal. Clearly, even cases with relatively smaller $c_{\tilde{e}}$ which optimally should be balanced like a line can be constructed. This condition is important as it provides support for treating some cycles as lines, which simplifies the solution substantially.

Yet, an important open questions related to the CSSBP remains, namely, whether CSSBP can be solved in polynomial time.

References

[1] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, first edition, 2009.

[2] Mike Benchimol, Pascal Benchimol, Benoît Chappert, Arnaud De La Taille, Fabien Laroche, Frédéric Meunier, and Ludovic Robinet. Balancing the stations of a self service "bike hire" system. *RAIRO-Operations Research*, 45(01):37–61, 2011.

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

[4] Daniel Chemla, Frédéric Meunier, and Roberto Wolfler Calvo. Bike sharing systems: solving the static rebalancing problem. *Discrete Optimization*, 10(2):120–146, 2013.

[5] Daniel Chemla, Frédéric Meunier, Thomas Pradeau, Roberto Wolfler Calvo, and Houssame Yahiaoui. Self-service bike sharing systems: simulation, repositioning, pricing. *<hal-00824078>*, 2013.

[6] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on theory of computing*, pages 151–158. ACM, 1971.

[7] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, third edition, 2009.

[8] Paul DeMaio. Bike-sharing: History, impacts, models of provision, and future. *Journal of Public Transportation*, 12(4):3, 2009.

[9] Güneş Erdoğan, Gilbert Laporte, and Roberto Wolfler Calvo. The one-commodity pickup and delivery traveling salesman problem with demand intervals. *Working paper*, 2012.

[10] Michael R. Garey and David S. Johnson. *Computers and intractability*. Freeman, first edition, 1979.

[11] Jonathan L. Gross and Jay Yellen. *Handbook of graph theory*. CRC press, 2004.

[12] Hipólito Hernández-Pérez and Juan-José Salazar-González. The one-commodity pickup-and-delivery travelling salesman problem. In *Combinatorial Optimization–Eureka, You Shrink!*, pages 89–104. Springer, 2003.

[13] Hipólito Hernández-Pérez and Juan-José Salazar-González. A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. *Discrete Applied Mathematics*, 145(1):126–139, 2004.

[14] Jon Kleinberg and Éva Tardos. *Algorithm design*. Pearson Education Ltd., first edition, 2014.

[15] Neal Lathia, Saniul Ahmed, and Licia Capra. Measuring the impact of opening the london shared bicycle scheme to casual users. *Transportation Research Part C: Emerging Technologies*, 22:88–102, 2012.

[16] Tal Raviv, Michal Tzur, and Iris A. Forma. Static repositioning in a bike-sharing system: models and solution approaches. *EURO Journal on Transportation and Logistics*, 2(3):187–229, 2013.

[17] Jasper Schuijbroek, Robert Hampshire, and Willem-Jan van Hoeve. Inventory rebalancing and vehicle routing in bike sharing systems. *Carnegie Mellon University Working Paper*, 2013.

[18] Peter Tittmann. *Graphentheorie*. Hanser Fachbuchverlag, 2003.