Algorithms and Complexity Results for the Single-Cut Routing Problem in a Rail Yard

Negin Enayaty Ahangar *[1], Kelly M. Sullivan[2], Shantih M. Spanton[3], and Yu Wang[3,4]

[1]Department of Operations Management, University of Texas at Dallas, Richardson, TX, USA, 75080

[2]Department of Industrial Engineering, University of Arkansas, Fayetteville, AR, USA, 72701

[3]CSX Technology, Jacksonville, FL, USA, 32202

[4]JD Logistics, Beijing, China, 101111

June 3, 2021

Abstract

Rail yards are facilities that play a critical role in the freight rail transportation system. A number of essential rail yard functions require moving connected "cuts" of rail cars through the rail yard from one position to another. In a congested rail yard, it is therefore of interest to determine whether there exists a feasible route for such a move and, if so, which one is shortest. With this motivation, we contribute theory and algorithms for the single-cut routing problem (SCRP) in a rail yard. Two key features distinguish SCRP from a traditional shortest path problem: (i) the entity (i.e., the locomotive and attached cut of cars) occupies space on the network; and (ii) track geometry further restricts route selection. To establish the difficulty of solving SCRP in general, we prove NP-completeness of a related problem that seeks to determine whether there is space in the rail yard network to position the entity in a given direction relative to a given anchor node. However, we then demonstrate this problem becomes polynomially solvable—and therefore, SCRP becomes polynomially solvable, too—for "bounded cycle length" (BCL) yard networks, a practically important class of networks in which the yard topology will not allow the entity to collide with itself. We formalize the resulting two-stage algorithm for BCL yard networks and validate our algorithm on a rail yard data set provided by the class I railroad CSX Transportation.

Keywords: Networks, Rail yard network, Shortest path, Algorithm, Routing

*Corresponding author. E-mail address: negin@utdallas.edu

Figure 1: Track infrastructure at an intermodal rail yard. (This image is authorized for such use by CSX Corporation, Inc. Any person or organization not affiliated with or authorized by CSX may not use, copy, alter or modify any CSX photographs, graphics, videography or other, similar reproductions or recordings without the advance written permission of CSX.)

1 Introduction

In the freight rail industry, rail cars are commonly carried across thousands of miles of track on several distinct trains (a collection of locomotives and rail cars) before reaching their destination. Numerous rail yards dot the physical rail network and act as sorting facilities, classifying rail cars by their final or intermediate destination in order to assemble outbound trains from rail cars arriving via inbound trains and/or customer shipment pickups.

Rail yard operations are critical to freight rail transportation but complex due to significant freight volume and structural limitations on rail car movements. Figure 1 depicts a typical intermodal rail yard, including the track infrastructure that enables (but restricts) car movements through the rail yard. Larger rail yards, and/or rail yards that perform different functions, may have increased complexity. For instance, Bailey Yard (in North Platte, Nebraska) is a classification yard that handles an average of $14,000$ rail cars per day and occupies over 4 square miles of land area [36]. Despite the complexity, rail yard operations are usually orchestrated by one or two yard officials overseeing multiple crews who manually execute instructions.

Rail yard operations depend on the ability to move cars through the rail yard. Connected *cuts* of cars are moved with locomotives for many purposes including bringing the cars into the yard; collecting cars together to go on an outbound train or to a customer; and moving cars to designated inspection, holding, and maintenance areas. With the exception of the gravity-assisted presorting

moves that occur at high-volume hump yards, nearly all routing of cars in the yard is accomplished *manually* by individual locomotives at the yard. Due to the high volume of manual handling moves, how these handling moves are routed has a significant impact on the yard's efficiency, including utilization of locomotives and throughput of cars. This motivates the need to develop an efficient algorithm that can route a cut of cars through a rail-yard. In the short term, these algorithms can be incorporated into decision-support systems to help yard officials better orchestrate yard operations. In the long term, these algorithms can be used as a building block toward automating yard operations. The following paragraphs further detail our motivation in the context of current yard operations planning procedures that are common in the freight rail industry.

Currently, real-time operations are planned "intuitively" by yard personnel. Strategic planning can be done with the assistance of specialized decision-support tools, including yard simulations that output key efficiency metrics such as throughput volume, on-time departure of cars, handling frequency, and total time in the yard. All but the most basic simulations incorporate routing cars through the yard. As routes must be calculated tens of thousands of times in any simulation, calculation speed is imperative. Although the details of these industry-used simulations remain unpublished, most simulations developed by several major rail carriers of which the authors are aware route cars using simple shortest path, modified shortest path, or pre-calculation of routes, often without considering the length of the cut of cars and locomotive; that is, the entity is considered a point mass. In practice, the locomotive and cut occupy space on the track, which can complicate routing decisions. Therefore, the existing routing techniques may fail when highly accurate routes are required.

With this motivation, we consider a routing problem posed over a network in which the edges correspond to individual tracks and the nodes are track connections or switches. On this network, we seek to determine a shortest feasible route within the rail yard for a single *entity*, consisting of the locomotive and attached cut of cars, subject to the geometry of the yard tracks assuming the location of other cuts of cars are unchanged. This problem differs from a traditional shortest path problem in that the route must (i) accommodate the entity's physical length, (ii) avoid prohibited switch node traversals, and (iii) satisfy restrictions on the entity's orientation (i.e., which way the locomotive is facing) at arrival and/or departure. For the sake of simplicity, we assume the network is *static* in the sense that other cars in the yard remain idle while the entity is en route. It is also of interest to solve multi-entity and/or dynamic yard routing problems; rather than solve these problems here, we have focused on analyzing the simpler case in detail in anticipation that our

results can be leveraged in the future to solve the more complex problems. We use a two-stage approach to solve this routing problem: an initial preprocessing stage determines whether there is sufficient track space for the yard to accommodate the entity positioned with one of its endpoints at a given node; given the output of the preprocessing stage, a routing stage then finds a shortest route through the yard network.

Our main contributions are as follows: (i) we introduce the single-cut routing problem (SCRP); (ii) we prove the preprocessing problem is NP-complete; (iii) we develop a linear programming formulation that solves an important special case of the preprocessing stage in polynomial time; (iv) given solutions to each switch node's preprocessing problem, we demonstrate the special case from (iii) also leads to polynomial-time solution of the routing stage and therefore a polynomial-time algorithm for the yard-routing problem; (v) we validate our model and algorithm on a rail yard data set provided by the class I railroad CSX Transportation.

Section 2 summarizes the related literature and positions our work relative to the literature. Section 3 introduces required terminology and notation, which is used in Section 4 to introduce the SCRP. Results for the preprocessing stage are presented in Section 5, and the routing stage algorithm is presented in Section 6. Section 7 summarizes the results from applying our algorithm to the CSX dataset, and Section 8 concludes.

2 Related Literature

The use of sophisticated decision support algorithms in freight rail is common by the industry's larger carriers. Some of the more well known algorithms related to macro-scale *line-of-road* routing are the problems of train design [1, 4, 6, 23, 41], timetabling [7, 11], and real-time traffic management [15, 32, 33, 34]. In these, infrastructure is primarily modeled as a network of nodes, corresponding to rail yards, and edges, corresponding to corridors. The routing itself is often modeled as simply as possible (given the scale of the networks) with the numerous operational constraints adding significant complexity amidst competing objectives of customer satisfaction, cost reduction, and efficient asset utilization. By contrast, the yard-level routing problem we consider here explicitly models the micro-structure of the yard with its detailed connections between individual tracks.

In the European passenger system, published work has introduced optimization tools for rail yard decision making, where the smaller infrastructure footprint has made future automation seem more attainable. Two theoretical problems presented in the passenger space that most resemble the

yard routing problem we consider here are the Train Routing Problem (TRP), and the Train Unit Shunting Problem (TUSP). Both these problems consider yard related servicing and sequencing problems for passenger rail. Certain components of these problems can generalize to freight service as well. As the scope of these problems is on holistic yard operations, these problems handle routing within their solutions with some simplification. Our problem represents an exact solution to the routing subproblem in these problems.

The related Train Routing Problem (TRP) was introduced to the rail literature by Zwaneveld et al. [43]. This problem (and its numerous extensions) relates to the passenger problem of "platforming" in which inbound trains arrive at an intermediate station at a collection of platforms to pick up passengers. Trains must utilize platforms and track segments such that they do not coincide with the moves of other trains and also, they must make their scheduled arrival departure times at all stations [43]. The output of these problems is a route and arrival/departure plan for all trains to minimize deviation from the schedule. The route is represented as a sequence of occupied resources (platforms, track segments) over time for each train, and thus TRP problem is then similar to factory allocation of jobs to production sequences. TRPs are often solved by node packing [43, 44], set packing [28, 30], multicommodity flow [10], alternative graphs [14], and of course heuristics [13]. Excellent reviews of this problem exist in [8, 29, 35]. Whereas the original TRP assumes known arrival and departure times for each train, research has since sought to derive TRP solutions that are robust to delays [5, 9, 12, 16, 21, 25]. The TRP problem differs from timetabling problems which aim to schedule the movement of trains across a network, in that they consider only how trains will be scheduled through the yards as a component of the timetabling process. Although there is recent interest in solving the timetabling and platforming problems in conjunction (see [42]), the problems are usually solved disjointly due to the complexity of the combined problem. In contrast with our problem, the potential routes for the TRP are generated *a priori*. These routes represent directional movements through the yard (e.g., from arrival to an intermediate platform or from an intermediate platform to departure). The precise movements needed to execute, say, the extraction of a single car across a large yard composed of thousands of tracks segments and switches, while using partially occupied tracks, may not be well represented by such a system.

Another relevant problem from the passenger research arena is the Train Unit Shunting Problem (TUSP) and its extension, the Train Unit Shunting and Servicing problem (TUSS). These problems are similar to the TRP in that they consider the activities of units within the yard. The TUSP considers an inbound passenger train arriving at a yard. For this train, the car components may

disassembled and parked (shunted) for the evening, and then matched to an outbound train in the morning [17, 19, 22]. The TUSP aims to coordinate matching, parking, and required movements so that car units are available for their assigned trains while often ensuring the safety constraint of observing a minimum headway between all units. Discussion of the routing between the subproblems of matching and parking was first given in Lentink et al. [27]. The approach to the routing subproblem in the research was similar to the TRP of Zwaneveld et al. [43] and was used in a decomposition approach to solve the TUSP [27]. Research that solves the components of matching, parking and routing in an integrated manner has been done by heuristic [38], as well as exactly for a reduced problem [26] and with strengthened constraints designed to prevent the crossing of units during their routing. While Lentink et al. [27] did include a single given service event in the TUSP formation, the TUSS further extends the problem to capture other service events. The Train Unit Shunting and Servicing (TUSS) problem, which includes the scheduling of several service events (such as maintenance) at potentially dynamic location(s), was first solved by heuristic in Van Den Broek et al. [40]. Machine learning techniques have been used to create feasible simulations of the TUSS [37, 39]. Recent research [20, 31] extends the Multi-Agent Pathfinding (MAPF) problem to solve a modest relaxation of the TUSS by adding service events as additional intermediate destination points to MAPF. The routing subproblem in particular is relaxed somewhat and the authors note that additional focus on the the routing sub-problem could improve the work [31]. Hendrikse [20] gives a discussion of the current attempts within the TUSS to include the infrastructure restrictions of the yard. This includes modeling tracks by multiple nodes to ensure length capacity of tracks, and reversed movements. The model we present here solves the yard routing subproblem exactly with the inclusion of the additional restrictions on movement needed to translate the problem to freight rail. For example, freight rail cars must be moved by a locomotive, where as the passenger cars in the European system move bidirectionally without locomotive power; thus, routes in a freight yard must be planned to ensure car orientation on the track without trapping the locomotive behind the cars on a single outlet track.

Our research is closely related to the work of Aliakbari et al. [3], who consider an identical routing problem with the underlying solution approach presented in this manuscript. They propose an alternative directed network expansion used to solve the routing stage. The routing algorithms of Aliakbari et al. [3] are based upon a directed network expansion that depends on the theory and algorithms for the preprocessing stage contributed by the present manuscript; that is, those algorithms of Aliakbari et al. [3] would not be justified without the results presented in this paper

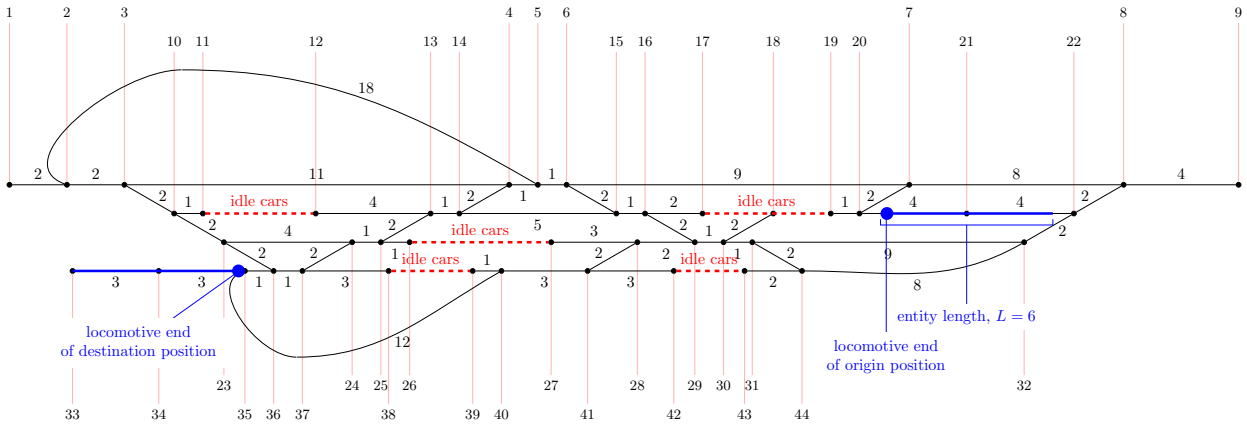

Figure 2: Example yard network in which each edge $e \in E$ is labeled with length c_e and each node's identifier i is labeled with a number attached to the node by a faint red vertical line. For convenience, nodes have been numbered in increasing order from left to right and top to bottom. Each switch node is drawn such that its acute angle appears as an acute angle. Dashed red lines are not to be interpreted as edges; rather, they indicate track segments that are unavailable because they are occupied by idle cars.

for the preprocessing stage. Although our directed network expansion is larger than the one due to Aliakbari et al. [3] by a constant factor of the number nodes and arcs in the original network, the Aliakbari et al. [3] directed network expansion has not been described as precisely as ours is, nor has it been demonstrated through numerical results (as ours is).

3 Yard Network Preliminaries

We consider the problem of routing an entity of length $L \geq 0$ through a rail yard defined by the undirected network $G = (N, E)$ with nodes N and edges E. Figure 2 illustrates an example yard network. Edges in the yard network correspond to track segments and nodes correspond to connections between track segments. Commonly, there are idle cars in the rail yard that obstruct the entity's movement. This feature can be incorporated by removing occupied portions of edges from the network.

The entity consists of a locomotive that can either pull or push the attached cars. Let $c_e \geq 0$ denote the length of edge $e \in E$, and let $N(i) \subseteq N \setminus \{i\}$ denote the set of nodes adjacent to node $i \in N$. Number the nodes adjacent to $i \in N$ as $N(i) = \{N_k(i)\}_{k=1}^{|N(i)|}$ and the edges adjacent to $i \in N$ as $\{e(i,k)\}_{k=1}^{|N(i)|}$ where $e(i,k) \equiv [i, N_k(i)]$. We refer to $N_1(i)$, $N_2(i)$, and $N_3(i)$ respectively as the *first*, *second*, and *third neighbor* of node i and to $\{N_k(i)\}_{k=1}^{|N(i)|}$ as the *ordered adjacency list* of node i. The yard network G has special structure as described below:

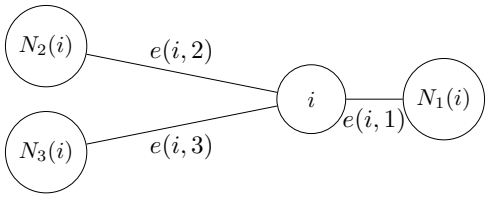

Figure 3: Schematic of a switch node i

Property 1 $|N(i)| \leq 3,\ \forall i \in N$. *That is, each node has degree at most three.*

In practice, there exist yard switches that split up to four ways; however, Property 1 is without loss of generality because these switches can be modeled as two degree-three switches separated by an edge of negligible length. We now formalize definitions that will enable analysis of yard networks.

Definition 1 If $|N(i)| = 3$, then $i \in N$ is said to be a *switch* node. Define $S \subseteq N$ as the set of switch nodes.

For a switch node $i \in S$, the collection of nodes $\{i, N_1(i), N_2(i), N_3(i)\}$ are always oriented (see Figure 3) such that, among the three edges $\{e(i,k)\}_{k=1}^{3}$, exactly one pair of edges forms an *acute angle*. Without loss of generality, suppose the edges $e(i,2)$ and $e(i,3)$ associated with each switch node $i \in S$ form an acute angle. By appropriately numbering the neighbors of each switch node, the following properties will be satisfied:

Property 2 *For $i \in N$ with $|N(i)| \geq 2$, the edge pair $\{e(i,1), e(i,2)\}$ is not an acute angle.*

Property 3 *For $i \in S$, the edge pair $\{e(i,1), e(i,3)\}$ is not an acute angle.*

Property 4 *For $i \in S$, the edge pair $\{e(i,2), e(i,3)\}$ is an acute angle.*

When a network satisfies Properties 1–4, we say that the network has *yard structure* or is a *yard network*. To illustrate Properties 1–4, examine switch node 3 in Figure 2. The edges $[3,4]$ and $[3,10]$ form an acute angle; therefore, $e(3,1)$ is the edge $[3,2]$, $e(3,2)$ is either $[3,4]$ or $[3,10]$, and $e(3,3)$ is the remaining edge adjacent to node 3. For a yard network satisfying Properties 1–4, we now present additional definitions toward representing the entity's positions in the yard network.

Definition 2 A sequence of edges $e(1)$–$e(2)$–$\cdots$–$e(m)$ is said to be a *walk* provided that each successive pair of edges has exactly one endpoint in common. Alternatively, we may refer to the walk $e(1)$–$e(2)$–$\cdots$–$e(m)$ by listing its nodes in sequence, e.g., as $i(0)$–$i(1)$–$\cdots$–$i(m)$.

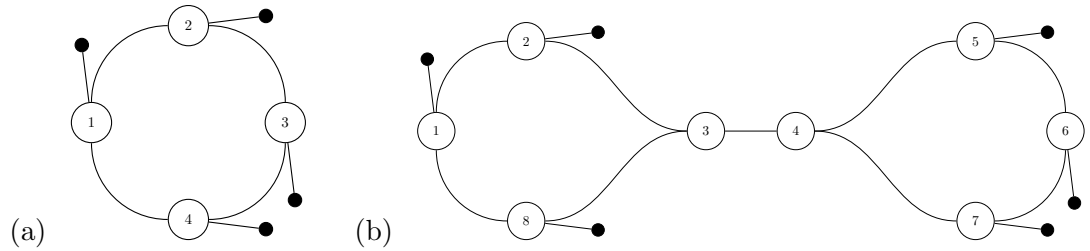

Figure 4: Illustration of closed AAF walks (a) 1–2–3–4–1 and (b) 3–2–1–8–3–4–5–6–7–4–3. In subfigure (b), note that 3–2–1–8–3 is not a closed AAF walk because $\{[2,3],[3,8]\}$ is an acute angle.

For simplicity of exposition, we also assume the following without loss of generality.

Assumption 1 *If $e \in E$ and $e' \in E$ share two common endpoints, then $e = e'$.*

Assumption 1 enables representing the length of an edge $e = [i,j] \in E$ as either c_e, c_{ij}, or c_{ji}. This assumption is without loss of generality because dual edges of the form $[i,j]$ can be "repaired" by replacing one of the two edges with a new switch node i', a new leaf node i'', and edges $\{e(i,k)\}_{k=1}^{3}$ such that $e(i',1) = [i',i]$, $e(i',2) = [i',j]$, and $e(i',3) = [i',i'']$. In this transformation, the new edge $[i',i'']$ is assigned a length of zero and edges $[i',i]$ and $[i',j]$ are each assigned a length equal to one-half of the replaced edge's length.

Definition 3 A walk $e(1)$–$e(2)$–$\cdots$–$e(m)$ is said to be an *acute-angle-free walk* (abbreviated as "AAF walk") provided that none of its adjacent edge pairs are acute angles. For instance, 15–6–5–2–3–4–5 is an AAF walk in Figure 2, but 3–4–5–2 is not because it traverses the acute angle $\{[4,5],[5,2]\}$.

Definition 4 An AAF walk $e(1)$–$e(2)$–$\cdots$–$e(m)$, with nodes $i(0)$–$i(1)$–$\cdots$–$i(m)$, is said to be *closed* if $i(0) = i(m)$ and $\{e(1), e(m)\}$ is not an acute angle. Because Figure 2 does not contain any closed AAF walks, we have illustrated two closed AAF walks in Figure 4.

Definition 5 An AAF walk $i(0)$–$i(1)$–$\cdots$–$i(m)$ is said to be a *simple acute-angle-free walk* (abbreviated as "SAAF walk") provided that (i) the walk visits $i(m)$ no more than twice and (ii) the remaining $|N|-1$ nodes are visited by the walk no more than once. (Thus, only one node may be visited twice by a SAAF walk, and that node must be the last node on the walk.) For example, 15–6–5–2–3–4–5 is a SAAF walk in Figure 2, but 15–6–5–2–3–4–5–6 is not due to the repetition of node 5.

Definition 6 An AAF walk (respectively, SAAF walk) W, with nodes $i(0)$–$i(1)$–$\cdots$–$i(m)$, is said to be an "AAF(j) walk" (respectively, "SAAF(j) walk") if $j = i(0)$. The walk W is said to be an "AAF(i,j) walk" (respectively, "SAAF(i,j) walk") if $i = i(0)$ and $j = i(m)$. For example, 15–6–5–2–3–4–5 is at once an AAF(15) walk, an AAF(15,5) walk, a SAAF(15) walk, and a SAAF(15,5) walk in Figure 2.

Definition 7 An AAF(j) walk (respectively, SAAF(j) walk) W, with nodes ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$, is said to be *positive anchored* if $i(1) = N_1(j)$. In this case, we say W is an "AAF($j_{[+]}$) walk" (respectively, "SAAF($j_{[+]}$) walk"). For example, 16–15–6–5–2–3–4–5 is a SAAF($16_{[+]}$) walk in Figure 2, but 16–29–30–31–44–32 is not because $N_1(16) \neq 29$ (i.e., edge $[16, 29]$ is in node 16's acute angle).

Definition 8 An AAF(j) walk (respectively, SAAF(j) walk) W, with nodes ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$, is said to be *negative anchored* if $i(1) \in \{N_2(j), N_3(j)\}$. In this case, we say W is an "AAF($j_{[-]}$) walk" (respectively, "SAAF($j_{[-]}$) walk"). For example, 16–29–30–31–44–32 is a SAAF($16_{[-]}$) walk in Figure 2.

Just as Definitions 7 and 8 classify an AAF(j) walk based upon whether its first edge leads to node $N_1(j)$, we may similarly classify an AAF(i, j) walk on the basis of its first and last edge.

Definition 9 An AAF(i,j) walk (respectively, SAAF(i,j) walk) W, with nodes ($i =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$ ($= j$), is said to be an "AAF($i_{[+]}$,j) walk" (respectively, "SAAF($i_{[+]}$,j) walk") if $i(1) = N_1(i)$ and an "AAF($i_{[-]}$,j) walk" (respectively, "SAAF($i_{[-]}$,j) walk") if $i(1) \in \{N_2(i), N_3(i)\}$. The walk W is said to be an "AAF(i,$j_{[+]}$) walk" (respectively, "SAAF(i,$j_{[+]}$) walk") if $i(m-1) = N_1(j)$ and an "AAF(i,$j_{[-]}$) walk" (respectively, "SAAF(i,$j_{[-]}$) walk") if $i(m-1) \in \{N_2(j), N_3(j)\}$. For example, 37–36–23–10–3–2 is a SAAF($37_{[+]}$,$2_{[+]}$) walk in Figure 2 because $N_1(37) = 36$ and $N_1(2) = 3$; however, 37–36–23–10–3 is a SAAF($37_{[+]}$,$3_{[-]}$) walk because $10 \in \{N_2(3), N_3(3)\}$.

The definitions above will be useful in characterizing the entity's movement through the yard, as we now summarize. To traverse the edge pair $\{e(j, 2), e(j, 3)\}$, $j \in S$, in sequence, the entity's entire length must first pass through node j before it reverses its direction to continue with the intended walk. For example, suppose we wish to move the entity from its original position in Figure 2 along a walk beginning 21–22–32–31 (using the acute angle at node 22) toward the destination position. As illustrated in Figure 5, the entire entity must pass node 22 and continue on to the SAAF($22_{[+]}$) walk 22–8–9 before traversing 22–32–31.

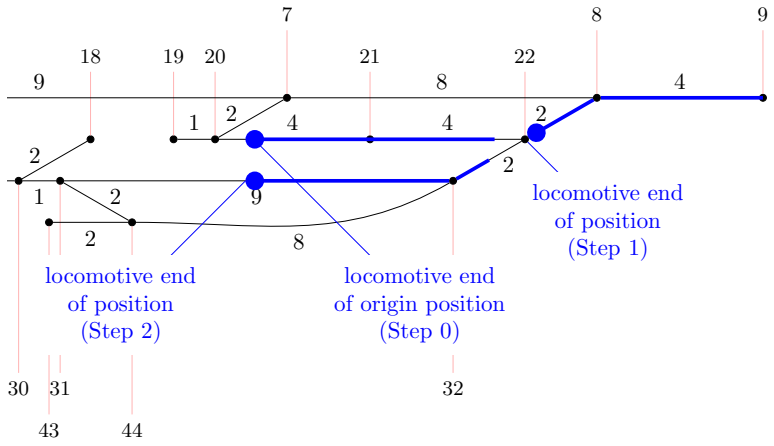

Figure 5: Illustration of a subnetwork of Figure 2 in which the entity traverses the acute angle at node 22.

As a result of the discussion in the previous paragraph, a necessary condition to traverse the acute angle $\{e(j,2), e(j,3)\}$, $j \in S$, is the existence of a SAAF($j_{[+]}$) walk of length at least L (hereafter referred to as a "SAAF($j_{[+]}$) feasible position"). In this case, we say the the acute angle $\{e(j,2), e(j,3)\}$ is feasible. These definitions are formalized below.

Definition 10 A SAAF walk $e(1)$–$e(2)$–$\cdots$–$e(m)$ (with node representation $i(0)$–$i(1)$–$\cdots$–$i(m)$) is said to be a *feasible position* for the entity if $c_{e(1)} + c_{e(2)} + \cdots + c_{e(m)} \geq L$. For instance, 20–21–22 is illustrated as the entity's original feasible position, where $L = 6$. If $j = i(0) \in S$ and $N_1(j) = i(1)$, then we say the feasible position is positive anchored at j or that the feasible position is a "SAAF($j_{[+]}$) feasible position." For example, 20–21–22 is not a SAAF($20_{[+]}$) feasible position in Figure 2 because $N_1(20) \neq 21$.

Definition 11 If there exists a SAAF($j_{[+]}$) feasible position, the acute angle $\{e(j,2), e(j,3)\}$ is said to be a *feasible acute angle*; otherwise, $\{e(j,2), e(j,3)\}$ is said to be an *infeasible acute angle*.

Corresponding to the problem of determining the feasibility of acute angle at switch node $j \in S$, it is of interest to identify SAAF($j_{[+]}$) and SAAF($j_{[-]}$) walks of maximum length. We therefore make the following additional definition.

Definition 12 Let γ_j and $\overline{\gamma}_j$ respectively denote the length of the longest SAAF($j_{[+]}$) walk and the length of the longest SAAF($j_{[-]}$) walk. For example, $\gamma_4 = 27$ corresponding to the SAAF($4_{[+]}$) walk 4–5–6–15–16–29–30–31–44–32–22–8–9 in Figure 2, and $\overline{\gamma}_{31} = 18$ corresponding to the SAAF($31_{[-]}$) walk 31–44–32–22–8–9.

For now, we have neglected the computational effort required to compute γ_j and $\overline{\gamma}_j$. We will address this subject in further detail in Section 5.

4 Problem Definition and Overview of Approach

In this section, we characterize how the entity moves inside the rail yard and formally define the SCRP, which (informally) seeks to determine how the entity of length L should move from a specified *origin feasible position* to a specified *destination feasible position* in order to minimize the total distance traveled by the entity. Without loss of generality, we assume the origin and destination are (dummy) *nodes* $f \in N$ and $g \in N$ (e.g., $f = 21$ and $g = 34$ in Figure 2), which can be interpreted as the initial and final location of the entity's midpoint.

By definition, the entity must travel on an AAF walk between consecutive traversals of acute angles. Suppose the entity traverses H acute angles (at switch nodes $j^1, j^2, \ldots, j^H$, numbered in the order of traversal) on its route from f to g and the corresponding sequence of AAF walks is $W^0, W^1, \ldots, W^H$. The route from f to g must satisfy the following conditions.

Condition 1 The acute angles traversed by the entity must all be feasible, That is, for all $h = 1, 2, \ldots, H$, there exists a SAAF($j^h_{[+]}$) feasible position.

Condition 2 The entity's orientation may further constrain the route selection. That is, at the origin and/or destination node, we may need to ensure the locomotive is positioned at the correct end of the entity. For instance, if g is a leaf node, the entity must arrive to g with the locomotive positioned on the side of the entity that faces the node adjacent to g; otherwise, the locomotive is trapped behind the cars.

Condition 3 The entity must not collide with itself.

Subject to Conditions 1–3, the SCRP aims to identify a route that minimizes the distance traversed by the entity. In traversing the AAF walk W^h ($h \in \{0, \ldots, H\}$), the midpoint traverses the entire length of W^h, plus an additional $L/2$ distance units per acute angle traversed at either end. The total distance traveled by the entity is given by

$$\text{distance}(W^0, \ldots, W^H) = \sum_{h=0}^{H} \text{distance}(W^h) = LH + \sum_{h=0}^{H} c(W^h), \tag{1}$$

where $c(W)$ denotes the total length of all edges in the walk W.

Given the need for rail providers to solve SCRP quickly and repeatedly, our research seeks to identify cases under which SCRP can be solved efficiently. Towards this end, a logical approach is to subdivide the routing problem into two stages: (i) a *preprocessing* stage that determines whether each switch node's acute angle is feasible and (ii) a *routing* stage that finds a shortest f-g walk that prohibits infeasible acute angles, penalizes feasible acute angles, and enforces orientation restrictions at f and g. For instance, if $L = 6$ in Figure 5, the preprocessing stage would identify that (among others) the acute angles at nodes 22 and 35 are feasible (e.g., due to the SAAF($22_{[+]}$) walk 22–8–9 and the SAAF($35_{[+]}$) walk 35–36–37–24–25–26). The routing stage would then determine that these two acute angles would be traversed in sequence (i.e., $j^1 = 22$ and $j^2 = 35$) and that the AAF walks W^0 (21–22), W^1 (22–32–31–30–29–28–41–40–35), and W^2 (35–34) would connect these acute angles to the origin node $f = 21$ and destination node $g = 34$. In this case, note that it would not have been sufficient for the entity to proceed directly to the destination node $g = 34$ along an AAF walk after traversing the acute angle at node 22; doing so would have resulted in a violation of Condition 2 and thus, the entity would have an incorrect final orientation.

As we will demonstrate, the viability of our two-stage approach depends on the presence (or absence) of SAAF walks from a node to itself, and on the length of such walks. To formalize these conditions, we make the following additional definition.

Definition 13 A SAAF walk $i(0)$–$i(1)$–$\cdots$–$i(m)$ is said to be an *acute-angle-free cycle* (abbreviated "AAF cycle") if $i(0) = i(m)$. For instance, 5–4–3–2–5 is at once a SAAF($5_{[-]}$) walk and an AAF cycle in Figure 2; however, it is not a closed AAF walk because $\{[5, 4], [2, 5]\}$ is an acute angle.

In fact, the two-stage approach yields a polynomial algorithm for SCRP under networks, which we refer to hereafter as "bounded cycle length" or "BCL," that satisfy the following assumption.

Assumption 2 *All AAF cycles in the yard network have length at least L.*

Assumption 2 simplifies SCRP in a couple of meaningful ways. Because the entity must always occupy an edge subset that forms a feasible position, Condition 3 is automatically satisfied by enforcing that any SAAF walk that begins and ends with the same node is longer than the entity itself; thus, the two-stage solution approach is valid, where the preprocessing stage evaluates all acute angles for feasibility and Conditions 1 and 2 are then addressed in the routing stage—in the vein of a "shortest path with turn penalties" problem—by reformulating SCRP as a traditional shortest path problem on an expanded directed network. It is important to note that Assumption 2

can be verified in polynomial time using an extension of classical shortest path algorithms. We explain in detail in Remark 2 of Section 5.2.

When Assumption 2 is not satisfied (i.e., for *non-BCL* networks), the two-stage approach fails altogether. For instance, the SAAF($j^h_{[+]}$) feasible position ($h \in \{1, \ldots, H\}$) may intersect the route leading into or away from node j, or the AAF walk W^h ($h \in \{0, \ldots, H\}$) may include AAF cycles. In either of these cases, an entity of sufficient length would collide with itself. As justification of the difficulty in solving the general SCRP (i.e., on possibly non-BCL networks), we prove (in Section 5) NP-completeness of the preprocessing problem.

In addition to the theoretical contributions described above, we now argue that our two-stage algorithm constitutes a significant practical contribution even though it will not handle non-BCL networks. Our reasoning follows:

(i) In practice, it is common to move both *short* entities (i.e., those of length no more than L) and *long* entities (those of length greater than L). In moving short entities, as is common in smaller yards where numerous cuts of cars must be moved to assemble trains, Assumption 2 holds and SCRP can be solved in polynomial time. Although Assumption 2 does not hold for long entities, these moves tend to be limited to specific yard functions performed in a dedicated portion of the yard that was designed for the function. (For instance, a lengthy inbounded train enters the yard and is placed onto one or more dedicated receiving tracks.) These moves will have a small number (usually just one) of potential routes and therefore require little to no planning.

(ii) The structure of an incoming or outgoing lead which fans out into many other tracks to sort and hold cars is typical of many rail yards, including the one depicted in Figure 1. Such yards are devoid of AAF cycles and are therefore BCL.

(ii) Although we cannot guarantee it, the BCL algorithm may provide a building block for solving SCRP on non-BCL networks, as we discuss in Section 8.

Section 5 provides analysis and algorithms for the preprocessing stage, and Section 6 provides the routing algorithm for BCL networks.

5 Preprocessing Stage

In this stage, we check each acute angle $\{e(j,2), e(j,3)\},\ j \in S$, for feasibility. The preprocessing problem for a switch node $j \in S$ is defined below.

PREPROCESS(j)

> **Instance** $I = \{N, E, L, c, j\}$**:** A yard network $G = (N, E)$ with ordered adjacency lists $\{N_k(i)\}_{k=1}^{|N(i)|}$ for all $i \in N$ and edge lengths $c_e \geq 0,\ e \in E$; an entity length $L \geq 0$; a switch node $j \in S$.
>
> **Question:** Does there exist a SAAF($j_{[+]}$) feasible position? (Equivalently: Does there exist a SAAF($j_{[+]}$) walk of length at least L?)

We prove that PREPROCESS($\cdot$) is NP-complete in Section 5.1 but show in Section 5.2 that it is polynomially solvable for BCL networks.

For the purposes of the analysis this section, it will be convenient to impose an additional restriction on the network's topology. Because degree-two nodes do not have acute angles (see Property 2), one can transform the yard network to an equivalent yard network with only leaf nodes and switch nodes. This transformation entails adding for every degree-two node i a new node i' and edge $e' = [i, i']$ with $c_{e'} = 0$ and defining $N_3(i) = i'$ and $e(i,3) = e'$. When convenient, we may therefore make the following assumption without loss of generality.

Assumption 3 *All nodes in $N \setminus S$ are leaf nodes.*

In what follows, we invoke Assumption 3 in Section 5 for the sake of analyzing the preprocessing problem but (beginning in Sections 6) relax this assumption for a clearer exposition of the routing algorithm.

5.1 Complexity

We establish the NP-completeness of PREPROCESS($\cdot$) based on a reduction from the longest path problem, which is known to be NP-complete [18].

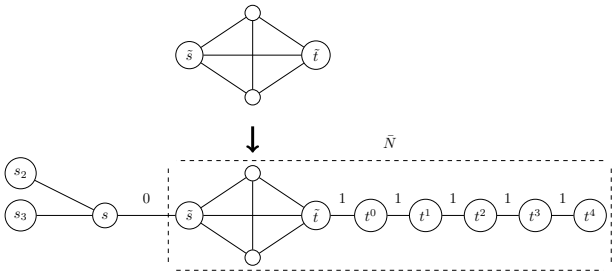

Figure 6: First step in constructing the instance $I = \{N, E, L, c, j\}$ of PREPROCESS(j)

LONGEST-PATH

> **Instance** $\tilde{I} = \{\tilde{N}, \tilde{E}, \tilde{L}, \tilde{s}, \tilde{t}\}$**:** An undirected network $\tilde{G} = (\tilde{N}, \tilde{E})$, a positive length $\tilde{L}$ and two nodes $\tilde{s}, \tilde{t} \in \tilde{N}$.
>
> **Question:** Does there exist a simple path (i.e., in which each node is visited at most once) of at least $\tilde{L}$ edges between the nodes $\tilde{s}$ and $\tilde{t}$ in network $\tilde{G}$?

We now establish the complexity of PREPROCESS($\cdot$).

Theorem 1 *PREPROCESS($\cdot$) is NP-complete.*

Proof Inclusion in NP is clear because the length of a walk can be computed in polynomial time. We prove NP-hardness of PREPROCESS($\cdot$) via reduction from LONGEST-PATH.

Given an arbitrary instance $\tilde{I} = \{\tilde{N}, \tilde{E}, \tilde{L}, \tilde{s}, \tilde{t}\}$ of LONGEST-PATH, we construct an instance $I = \{N, E, L, c, j\}$ of PREPROCESS(j) as described below. We first expand $\tilde{G} = (\tilde{N}, \tilde{E})$ by attaching a chain of $|\tilde{N}| + 1$ nodes $\{t^0, t^1, t^2, \cdots, t^{|\tilde{N}|}\}$ and edges $\{[\tilde{t}, t^0], [t^0, t^1], \ldots, [t^{|\tilde{N}|-1}, t^{|\tilde{N}|}]\}$ to node $\tilde{t}$. Let $\bar{N}$ denote the modified node set.

Thus far, yard structure (i.e., Properties 1–4 as defined in Section 3) has not been imparted on the network. For example, we may currently have that $d(i) > 3$ for some nodes $i \in \bar{N}$, where $d(i)$ is the degree of node i in the expanded network. We now explain an expansion that will provide the required yard structure.

Initially, we construct a new node s and add the edge $[s, \tilde{s}]$, increasing $d(\tilde{s})$ by one. We also construct nodes $\{s_2, s_3\}$ as leaf-node neighbors of s. Now that $d(s) = 3$, we may define s as a switch node (as is required in order for PREPROCESS(s) to be well-defined) with $N_1(s) = \tilde{s}$, $N_2(s) = s_2$, and $N_3(s) = s_3$ and set $j = s$. (In what follows, it is possible that $\tilde{s}$ will be replaced as the first neighbor of s.) The transformation up to this point is shown in Figure 6.

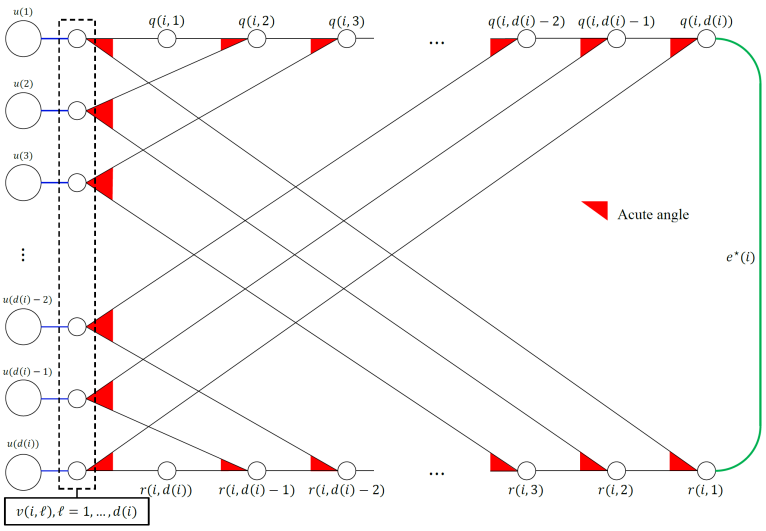

Figure 7: Expansion performed on node $i \in \bar{N}$ with $d(i) \geq 3$

Now, for each node $i \in \bar{N}$ (which does not include $\{s, s_2, s_3\}$), we perform the following: (i) if $d(i) = 1$, no action is required because Properties 1–4 impose no restriction on leaf nodes; (ii) if $d(i) = 2$, no action is required—the pair of edges adjacent to i will not form an acute angle in the yard network; (iii) if $d(i) \geq 3$, perform the expansion that is depicted in Figure 7 and described in the following paragraph.

For $i \in \bar{N}$ with $d(i) \geq 3$, we let $\{u(1), u(2), \ldots, u(d(i))\}$ denote the neighbors of i and replace node i with $3d(i)$ new switch nodes defined by the sets $\{v(i,\ell)\}_{\ell=1}^{d(i)}$, $\{q(i,\ell)\}_{\ell=1}^{d(i)}$, and $\{r(i,\ell)\}_{\ell=1}^{d(i)}$. Edges are added between these nodes as follows:

- For $\ell = 1, \ldots, d(i)$, define $N_1(v(i,\ell)) = u(\ell)$, $N_2(v(i,\ell)) = q(i,\ell)$, and $N_3(v(i,\ell)) = r(i,\ell)$.
- For $\ell = 2, \ldots, d(i) - 1$, define $N_1(q(i,\ell)) = q(i,\ell+1)$, $N_2(q(i,\ell)) = v(i,\ell)$, and $N_3(q(i,\ell)) = q(i,\ell-1)$. Perform the same definitions for $\ell = 1$ and $\ell = d(i)$ with the following exceptions:
  - For $\ell = d(i)$, define $N_1(q(i,\ell)) = r(i,1)$.
  - For $\ell = 1$, $N_3(q(i,\ell))$ is undefined. Thus, $q(i,1)$ will have degree two in the transformed network.
- For $\ell = 2, \ldots, d(i) - 1$, define $N_1(r(i,\ell)) = r(i,\ell-1)$, $N_2(r(i,\ell)) = v(i,\ell)$, and $N_3(r(i,\ell)) = r(i,\ell+1)$. Perform the same definitions for $\ell = 1$ and $\ell = d(i)$ with the following exceptions:
  - For $\ell = 1$, define $N_1(r(i,\ell)) = q(i,d(i))$.

– For $\ell = d(i)$, $N_3(r(i,\ell))$ is undefined. Thus, $r(i,d(i))$ will have degree two in the transformed network.

Because node i—previously a neighbor of nodes $\{u(1), u(2), \ldots, u(d(i))\}$—has been removed, node $v(i,\ell)$ assumes node i's position in the ordered adjacency list for node $u(\ell),\ \ell = 1, \ldots, d(i)$. Define $V(i) = \{v(i,\ell)\}_{\ell=1}^{d(i)} \cup \{q(i,\ell)\}_{\ell=1}^{d(i)} \cup \{r(i,\ell)\}_{\ell=1}^{d(i)}$ as the set of new nodes created by expanding node $i \in \bar{N}$ (where $V(i) = \{i\}$ for $i \in \bar{N}$ with $d(i) \leq 2$) and $e^\star(i)$ as the new edge between $q(i,d(i))$ and $r(i,1)$. For $d(i) \geq 3$, observe that a SAAF walk between any pair of nodes in $\{v(i,\ell)\}_{\ell=1}^{d(i)}$ on the subgraph induced by $V(i)$ must traverse $e^\star(i)$.

Let G denote the resulting network, and observe that the nodes of the resulting network G have maximum degree three with all degree-three nodes having designated first, second, and third neighbors—therefore, G constitutes a yard network. Let N and E respectively denote the nodes and edges of this network, and define S as the set of degree-three nodes. The node sets $V(i),\ i \in \bar{N}$ are disjoint, and only s, s_2, and s_3 are contained within N but not $\bigcup_{i\in\bar{N}} V(i)$.

We assign the length of edge $e \in E$ as $c_e = 0$ if (i) there exists a node $i \in \bar{N}$ such that both of the endpoints of E are elements of $V(i)$ or (ii) one of the endpoints of e is the node s; otherwise, $c_e = 1$. In particular, note that for each $t^\ell,\ \ell = 1, \ldots, |\tilde{N}|$, the edge $[t^{\ell-1}, t^\ell]$ is contained in E with unit length.

We now prove that the answer to PREPROCESS(s) for an entity of length $L = \tilde{L} + |\tilde{N}| + 1$ is "yes" if and only if the answer to LONGEST-PATH is "yes." Suppose there exists a SAAF($s_{[+]}$) feasible position in G. Note that this position intersects a sequence of node sets $V(i(0))$–$V(i(1))$–$\cdots$–$V(i(m))$, where $i(h) \in \bar{N},\ \forall h \in \{0, 1, \ldots, m\}$ and $[i(h-1), i(h)] \in \tilde{E},\ \forall h \in \{1, \ldots, m\}$, and $i(0) = \tilde{s}$ based on the construction of G. The length of this walk from the switch node s until it intersects $V(i(0)) = V(\tilde{s})$ is zero. For a given $h \in \{0, 1, \ldots, m\}$, the walk may pass through several edges whose endpoints are both in $V(i(h))$; however, the combined length of all such edges is zero. The walk also traverses m unit-length edges in moving from $V(i(h-1))$ to $V(i(h))$ for all $h = 1, \ldots, m$, and the total length of the walk is therefore

$$m \geq L\ (= \tilde{L} + |\tilde{N}| + 1). \tag{2}$$

Because each SAAF walk that enters and exits $V(i),\ i \in \bar{N}$, must traverse $e^\star(i)$, it is therefore impossible to enter and exit $V(i)$ a second time. Because each $V(i),\ i \in \bar{N}$, may be exited at most once, the nodes $\{i(0), i(1), \ldots, i(m-1)\}$ are unique. Combining this observation with Equation (2),

the maximum length of any SAAF($s_{[+]}$) walk would be upper-bounded by $|\tilde{N}|$ if the chain t^0–t^1–$\cdots$–$t^{|\tilde{N}|}$ were excluded from E. Because $m > |\tilde{N}|$, the set $\{i(0), i(1), \ldots, i(m)\}$ must include nodes in the chain. Furthermore, a SAAF walk that enters the chain cannot leave the chain, and therefore $i(m) \in \{t^0, t^1, \ldots, t^{|\tilde{N}|}\}$. Without loss of generality, we assume that $i(m) = t^{|\tilde{N}|}$, extending the feasible position if necessary so that it includes all nodes and edges in the chain. Summarizing the above, we have identified a feasible position that satisfies $i(h) = t^{|\tilde{N}|-m+h}$, $h \in \{m - |\tilde{N}|, m - |\tilde{N}| + 1, \ldots, m\}$. Because $[i(h-1), i(h)] \in \tilde{E}$, $\forall h \in \{1, \ldots, m\}$, we have that $(\tilde{s} =)$ $i(0)$–$i(1)$–$\cdots$–$i(m - |\tilde{N}| - 1)$ $(= \tilde{t})$ is an $\tilde{s}$-$\tilde{t}$ path in $\tilde{G}$ of length $m - |\tilde{N}| - 1$, which is at least $\tilde{L}$ by Equation (2). The answer to LONGEST-PATH is therefore "yes."

Conversely, suppose there exists an $\tilde{s}$-$\tilde{t}$ path $(\tilde{s} =)$ $i(0)$–$i(1)$–$\cdots$–$i(h)$ $(= \tilde{t})$ of $h \geq \tilde{L}$ edges in $\tilde{G}$. Based on construction of $\tilde{G}$, there is a SAAF($s_{[+]}$) walk in G with length h that traverses the node sets $V(i(0))$–$V(i(1))$–$\cdots$–$V(i(h))$. Noting that $i(h) = \tilde{t}$ and the SAAF($s_{[+]}$) walk has not yet traversed any nodes in the chain, we may extend this walk to obtain the SAAF($s_{[+]}$) walk $V(i(0))$–$V(i(1))$–$\cdots$–$V(i(h))$–$V(t^0)$–$V(t^1)$–$\cdots$–$V(t^{|\tilde{N}|})$, which has length $h + \tilde{N} + 1 \geq \tilde{L} + \tilde{N} + 1 = L$. Therefore, the answer to PREPROCESS(s) is "yes" and the proof of this theorem is complete. ■

This proof establishes NP-completeness for the general case of PREPROCESS($\cdot$). The reduction, however, yields yard networks with zero-length AAF cycles, which do not appear in practice. Indeed, as we summarize in Section 5.2, PREPROCESS($\cdot$) becomes polynomially solvable for BCL networks in which all AAF cycles must have length at least L.

5.2 Special cases of PREPROCESS($\cdot$)

In this section, we present special cases of yard network for which PREPROCESS($\cdot$) is polynomially solvable.

Case 1: G contains no acute-angle-free cycles

We demonstrate that if G contains no AAF cycles, we can solve PREPROCESS(j) in polynomial time. Although this assumption is not realistic in practice, it provides considerable simplifications that will extend naturally to the more realistic case (see next case) of BCL networks.

When G contains no AAF cycles, we demonstrate that PREPROCESS(j) can be solved for all $j \in S$ by solving a single linear program (LP) that simultaneously determines γ_j for every switch node $j \in S$. Naturally, $\gamma_j \geq L$ will be necessary and sufficient to imply the answer to PREPROCESS(j) is "yes."

Let y_i and $x_i,\ i \in S$, respectively be variables that represent γ_i and $\overline{\gamma}_i$. The LP is given as

$$\min \sum_{i \in S} y_i, \tag{3a}$$

$$\text{s.t. } y_i \geq c_{e(i,1)}, \qquad \forall i \in S:\ |N(N_1(i))| = 1, \tag{3b}$$

$$y_i \geq x_{N_1(i)} + c_{e(i,1)}, \qquad \forall i \in S:\ N_1(i) \in S, N_1(N_1(i)) = i, \tag{3c}$$

$$y_i \geq y_{N_1(i)} + c_{e(i,1)}, \qquad \forall i \in S:\ N_1(i) \in S, N_1(N_1(i)) \neq i, \tag{3d}$$

$$x_i \geq c_{e(i,k)}, \qquad \forall i \in S,\ k \in \{2,3\} : |N(N_k(i))| = 1, \tag{3e}$$

$$x_i \geq x_{N_k(i)} + c_{e(i,k)}, \qquad \forall i \in S,\ k \in \{2,3\}:\ N_k(i) \in S,\ N_1(N_k(i)) = i, \tag{3f}$$

$$x_i \geq y_{N_k(i)} + c_{e(i,k)}, \qquad \forall i \in S,\ k \in \{2,3\}:\ N_k(i) \in S,\ N_1(N_k(i)) \neq i, \tag{3g}$$

invoking Assumption 3 to eliminate degree-two nodes. Every SAAF($i_{[+]}$) (respectively, SAAF($i_{[-]}$)) walk contains the edge $e(i,1)$ (respectively, either $e(i,2)$ or $e(i,3)$): If this edge connects i to a leaf node, Constraints (3b) and (3e) respectively provide a (constant) lower bound on the length of the longest SAAF($i_{[+]}$) and SAAF($i_{[-]}$) walks in the network. Otherwise, Constraint (3c) or (3d) bounds the length of the longest SAAF($i_{[+]}$) walk, and Constraint (3f) or (3g) bounds the length of the longest SAAF($i_{[-]}$) walk. The objective function (3a) forces the y-variables to assume the smallest possible values given the bounding constraints.

Before establishing the relevant properties of LP (3), we first illustrate its application to the notional yard network depicted in Figure 8. (Note that 4–5–7–4 includes the acute angles at nodes 4 and 7; therefore, the network has no AAF cycles.) The LP for this example is

$$\min\ y_3 + y_4 + y_5 + y_7, \tag{4a}$$

$$\text{s.t. } x_3 \geq 3,\ x_3 \geq 2,\ x_5 \geq 3,\ y_7 \geq 2; \tag{4b}$$

$$y_3 \geq x_4 + 2,\ x_4 \geq x_5 + 3,\ x_4 \geq y_7 + 6,\ y_4 \geq x_3 + 2,$$

$$x_5 \geq y_7 + 2,\ y_5 \geq y_4 + 3,\ x_7 \geq y_5 + 2,\ x_7 \geq y_4 + 6. \tag{4c}$$

Constraints (4b) specify lower bounds on the maximum SAAF walk length from nodes 3, 5, and 7 in the direction of their leaf node neighbors. The x-variables are used in the first three constraints because the leaf nodes are either the second or third neighbor of node 3 or 5; however, y_8 is used in the sixth constraint because the leaf node is the first neighbor of node 8. Constraints (4c) ensure that variables x and y of the switch nodes are in a correct relationship with their neighbors' variables

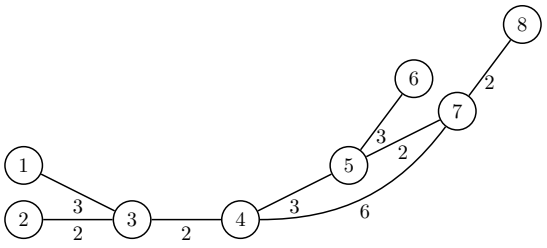

Figure 8: Notional yard network with no AAF cycles. Each switch node is drawn such that its acute angle appears as an acute angle.

based on how they are connected. For example, "$x_4 \geq x_5 + 3$"ensures that x_4 (the maximum length of a SAAF($4_{[-]}$) walk) should be at least 3 (the length of edge $[4, 5]$) plus x_5 (the maximum length of a SAAF walk from node 5 in the direction of nodes 6 and 7). An optimal solution to this LP is $(x_3, x_4, x_5, x_7) = (3, 8, 4, 11)$ and $(y_3, y_4, y_5, y_7) = (10, 5, 8, 2)$. If $L = 6$, PREPROCESS(j) would return "yes" for $j = 3$ and $j = 5$ (because $y_j \geq L$) and "no" otherwise.

Theorem 2 *If G contains no AAF cycles, then (a) LP* (3) *is feasible and (b) $y_j = \gamma_j$, $\forall j \in S$, in every optimal solution.*

Proof Given in Appendix A. ∎

Case 2: All acute-angle-free cycles in G have length at least L

We now demonstrate that PREPROCESS($j^\star$) for a given switch node $j^\star \in S$ remains polynomially solvable for BCL networks using a modified version of LP (3). The procedure for this case will require solving a different LP associated with each switch node $j^\star \in S$. We begin by modifying the network $G = (N, E)$ to obtain a new network $G^{(j^\star)} = (N^{(j^\star)}, E^{(j^\star)})$ with switch nodes $S^{(j^\star)}$ as follows: (i) Let $j_2 \equiv N_2(j^\star)$ and $j_3 \equiv N_3(j^\star)$ denote the second and third neighbors of $j^\star$ in the original network; (ii) create four new nodes to define $N^{(j^\star)} = N \cup \{j_2', j_2'', j_3', j_3''\}$; (iii) set $E^{(j^\star)} = E \cup \{[j^\star, j_2'], [j_2'', j_2], [j^\star, j_3'], [j_3'', j_3]\} \setminus \{[j^\star, j_2], [j^\star, j_3]\}$; (iv) respectively define the second and third neighbors of $j^\star$ in $G^{(j^\star)}$ as j_2' and j_3'; (v) in a similar fashion, define j_2'' and j_3'' as the appropriately numbered neighbors of j_2 and j_3 in $G^{(j^\star)}$, replacing $j^\star$'s position in their previous adjacency list; and (vi) respectively assign lengths 0, $c_{j^\star, j_2}$, 0, and $c_{j^\star, j_3}$ to edges $[j^\star, j_2']$, $[j_2'', j_2]$, $[j^\star, j_3']$, and $[j_3'', j_3]$. This modification is intended to disconnect the node $j^\star$ from its second and third neighbors as any SAAF($j^\star{}_{[+]}$) feasible position should end immediately if it re-visits $j^\star$. Note that $j^\star$ is a switch node in the transformed network, so the variable $y_{j^\star}$ is included in LP (3).

With some minor modifications, the LP (3) can be applied to $G^{(j^\star)}$ to solve PREPROCESS($j^\star$); hereafter, let $\text{LP}_{j^\star}$ denote this LP. Having $y_{j^\star} \geq L$ in an optimal solution to $\text{LP}_{j^\star}$ turns out to be a

sufficient condition to prove the acute angle at switch node $j^\star$ is feasible, but it is not a necessary condition. In fact, a feasible acute angle at switch node $j^\star$ may correspond to an infeasible $\text{LP}_{j^\star}$ in certain cases. We will demonstrate that infeasibility of $\text{LP}_{j^\star}$ results when $G^{(j^\star)}$ contains a closed AAF walk. Provided that such a walk is reachable from $j^\star$, Assumption 2 will allow us to use infeasibility of $\text{LP}_{j^\star}$ to conclude that the acute angle at switch node $j^\star$ is feasible. Towards solving PREPROCESS($j^\star$) in this manner, we first remove from $G^{(j^\star)}$ any edges in $E^{(j^\star)} \setminus \{[j^\star, j_2'], [j^\star, j_3']\}$ that are not contained in any AAF(${j^\star}_{[+]}$) walk. We accomplish this by performing a modified breadth first search (keeping track of which edges have been reached and in what direction) starting from node $j^\star$ to identify which edges' constraints should remain in $\text{LP}_{j^\star}$. Therefore, the resulting $\text{LP}_{j^\star}$ is formulated over a yard network $G^{(j^\star)} = (N^{(j^\star)}, E^{(j^\star)})$ that satisfies the following assumption.

Assumption 4 *All edges in $E^{(j^\star)} \setminus \{e(j^\star, 2), e(j^\star, 3)\}$ are reachable via an AAF(${j^\star}_{[+]}$) walk.*

After removing edges that are not contained in an AAF(${j^\star}_{[+]}$) walk, it is possible that some switch nodes were converted into degree-two nodes by removing either their second or third neighbor—in this case, we restore Assumption 3 as summarized immediately before the statement of Assumption 3.

Theorem 3 *For BCL networks, the acute angle at switch node $j^\star \in S^{(j^\star)}$ is feasible if and only if (a) $LP_{j^\star}$ is infeasible or (b) $y_{j^\star} \geq L$ in an optimal solution of $LP_{j^\star}$.*

Proof Given in Appendix B ∎

The proof of Theorem 3 also allows us to formalize procedures (in Remark 1) to ensure Assumption 4 is satisfied and (in Remark 2) to determine whether a yard network is BCL. These remarks are available in Appendix B. Theorem 3 also implies the following corollary.

Theorem 4 *PREPROCESS($j^\star$) is polynomially solvable for BCL networks.*

Proof By Theorem 3, PREPROCESS($j^\star$) can be solved using $\text{LP}_{j^\star}$, whose formulation is polynomial in the size of the PREPROCESS($j^\star$) input. The result follows because LPs are polynomially solvable [24]. ∎

6 Routing Stage

In this section, we prescribe a procedure by which the routing stage can be solved in polynomial time for BCL networks given a solution to each switch node's preprocessing problem. Let S^+ and

Table 1: Ordered adjacency lists for the instance in Figure 9

Node (i)	$N_1(i)$	$N_2(i)$	$N_3(i)$
1	2		
2	1	3	7
3	4	2	
4	5	3	
5	6	4	
6	7	5	
7	8	6	2
8	7		

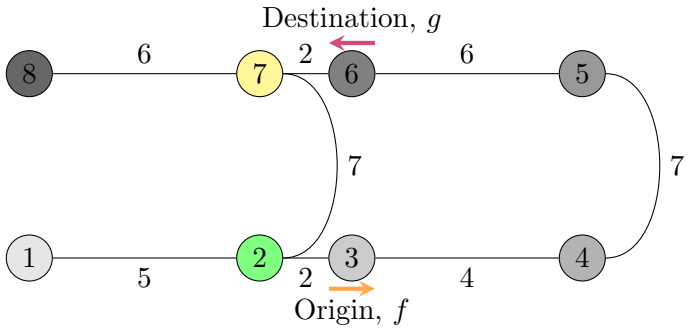

Figure 9: Notional SCRP instance in which the entity has length $L = 2$, origin node $f = 3$ (with the locomotive facing node $N_1(3) = 4$ as indicated by →), destination node 6 (with the locomotive facing node $N_1(6) = 7$ as indicated by ←).

S^- partition the set of switch nodes S such that PREPROCESS(i) = "yes" for all $i \in S^+$ and PREPROCESS(i) = "no" for all $i \in S^-$. Given S^+ and S^-, SCRP reduces to finding a shortest walk $(f =)\ i(0)$–$i(1)$–$\cdots$–$i(m)$ $(= g)$ such that: (i) for $h \in \{1, \ldots, m-1\}$, if $\{[i(h-1), i(h)], [i(h), i(h+1)]\}$ is an acute angle then $i(h) \in S^+$; (ii) the entity's orientation restrictions are satisfied at the origin and destination; and (iii) for each $h \in \{1, \ldots, m-1\}$ such that $\{[i(h-1), i(h)], [i(h), i(h+1)]\}$ is an acute angle, a penalty of L is added to the walk's length. We will refer to a walk satisfying (i)–(ii) as a *SCRP walk* and to its length, after adjusting due to (iii), as *penalized length*. Note that (i)–(ii) are a restatement of Conditions 1–2 for a single (possibly non-AAF) walk as opposed to a collection of AAF walks, and Condition 3 is satisfied due to the BCL assumption.

We now demonstrate that the problem of identifying a SCRP walk of shortest penalized length can be solved by traditional shortest path methods after building an expanded, directed network that addresses (i)–(iii). The directed network $\bar{G} = (\bar{N}, \bar{A})$ with node set $\bar{N}$ and arc set $\bar{A}$ is constructed using the following procedure, which we illustrate in Figure 10 for an example defined in Figure 9 and Table 1.

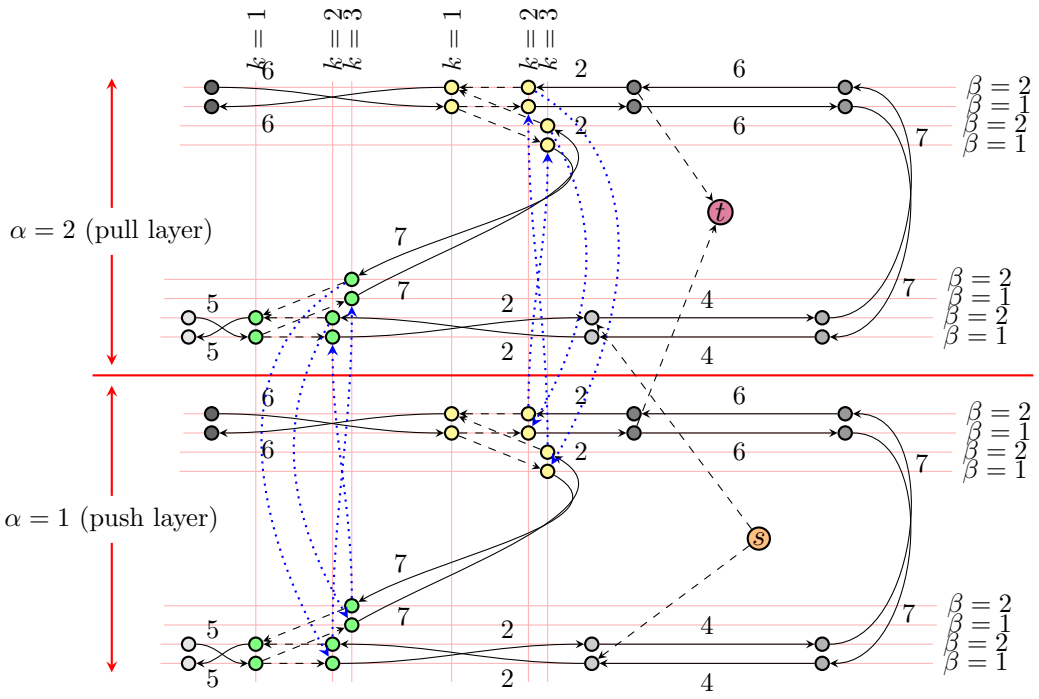

Figure 10: Expanded directed network for the notional yard network in which dashed arcs (- - →) have length zero and dotted blue arcs (⋯→) have length L. Nodes of the form $i^{\langle\alpha,\beta\rangle}$ and $i^{\langle\alpha,\beta,k\rangle}$ are color coded to match the node the corresponding node i from Figure 9.

6.1 Constructing Nodes

In the directed network $\bar{G}$, two identical layers are constructed and indexed by $\alpha \in \{1, 2\}$. Traversing a node in layer $\alpha = 1$ ($\alpha = 2$) implies the locomotive pushes (pulls) the entity through the corresponding node in G. Steps 1–2 define the nodes used in each layer.

Step 1: For each non-switch node $i \in N \setminus S$, construct two nodes $\{i^{\langle\alpha,\beta\rangle} : \beta \in \{1, 2\}\}$ within each layer $\alpha \in \{1, 2\}$. (Note: In this section, we do not employ Assumption 3. Therefore, the set $N \setminus S$ may include both leaf nodes and degree-two nodes.) Traversing a node $i^{\langle\alpha,\beta\rangle}$ with $\beta = 2$ ($\beta = 1$) implies the entity is moving toward (away from) its first neighbor (i,e., $N_1(i)$) when it passes through node i in G.

Step 2: For each switch node $i \in S$, construct six nodes $\{i^{\langle\alpha,\beta,k\rangle} : \beta \in \{1, 2\},\ k \in \{1, 2, 3\}\}$ in each layer $\alpha \in \{1, 2\}$. The index β has the same interpretation for switch nodes as for non-switch nodes. Traversing a node $i^{\langle\alpha,\beta,k\rangle}$ corresponds to a route that arrives to or departs from node $i \in S$ using edge $e(i, k)$.

The nodes created by Steps 1–2 for the example network are depicted in Figure 10, where the node

color maps the constructed nodes back to their corresponding node in Figure 9.

In addition to the nodes in layers $\alpha \in \{1, 2\}$, dummy source and sink nodes are created in Step 3 to allow for selecting potential starting and ending nodes among those nodes created in Steps 1–2. For example, it may be possible for the entity to begin with the locomotive either pushing or pulling; therefore, we would connect s to a node in each layer.

Step 3: Construct source node s and sink node t in $\bar{N}$.

In the following subsections, we define the arcs constructed (in Section 6.2) within each layer, (in Section 6.3) across layers, and (in Section 6.4) adjacent to the source and sink.

6.2 Constructing Within-Layer Arcs

Directed arcs constructed within each layer of $\bar{G}$ to characterize the entity's potential movement without reversing direction (i.e., along an AAF walk). In this construction the endpoints of each directed arcs are assigned to be consistent with the interpretation of α and β given in Section 6.1. Step 4 constructs arcs in each layer to represent potential moves between a pair of nodes that are adjacent in G.

Step 4: For each edge $e = [i, j] \in E$, construct two directed arcs in each layer $\alpha \in \{1, 2\}$ of $\bar{A}$ as follows. Let $k', k'' \in \{1, 2, 3\}$ denote values such that $j = N_{k'}(i)$ and $i = N_{k''}(j)$, and let

$$b(i,j) = \begin{cases} 1 & \text{if } j = N_1(i), \\ 2 & \text{otherwise,} \end{cases} \quad \text{and} \quad b(j,i) = \begin{cases} 1 & \text{if } i = N_1(j), \\ 2 & \text{otherwise.} \end{cases} \tag{5}$$

The directed arcs are constructed in a manner that depends on whether or not i and/or j are switch nodes. If both $i \in S$ and $j \in S$, construct the two directed arcs

$$\left\{ (i^{\langle \alpha, 3-b(i,j), k' \rangle}, j^{\langle \alpha, b(j,i), k'' \rangle}), (j^{\langle \alpha, 3-b(j,i), k'' \rangle}, i^{\langle \alpha, b(i,j), k' \rangle}) \right\}, \tag{6}$$

in each layer $\alpha \in \{1, 2\}$ of $\bar{A}$, each with length c_e. If at least one of $\{i, j\}$ is not a switch node, add two directed arcs to each layer similarly after (if $i \notin S$) removing index k' and/or (if $j \notin S$) removing index k''.

Figure 11 depicts a "zoomed in" view of the arcs created in Step 4 by three edges from Figure 9. For edge $e = [i, j] = [3, 4]$, only the endpoint node 4 is considered the first neighbor by Table 1 and

therefore $k' = 1$, $k'' = 2$, $b(i,j) = 1$ and $b(j,i) = 2$. By Equation (6) (ignoring the indices k' and k'' because neither 3 nor 4 is a switch node), the resulting two directed arcs depicted in Figure 11(a) for each layer $\alpha \in \{1,2\}$ are therefore $\{(3^{\langle\alpha,2\rangle}, 4^{\langle\alpha,2\rangle}), (4^{\langle\alpha,1\rangle}, 3^{\langle\alpha,1\rangle})\}$. For $e = [i,j] = [1,2]$, both endpoints nodes 1 and 2 are the first neighbor of each other (i.e., $k' = k'' = b(i,j) = b(j,i) = 1$), yielding in each layer $\alpha \in \{1,2\}$ the two directed arcs $\{(1^{\langle\alpha,2\rangle}, 2^{\langle\alpha,1,1\rangle}), (2^{\langle\alpha,2,1\rangle}, 1^{\langle\alpha,1\rangle})\}$ under Equation (6), as depicted in Figure 11(b). For $e = [i,j] = [2,7]$, neither node 2 nor node 7 is the first neighbor of the other. By Table 1, we have have $k' = k'' = 3$ and $b(i,j) = b(j,i) = 2$; therefore, the two directed arcs $\{(2^{\langle\alpha,1,3\rangle}, 7^{\langle\alpha,2,3\rangle}), (7^{\langle\alpha,1,3\rangle}, 2^{\langle\alpha,2,3\rangle})\}$ are constructed in each layer $\alpha \in \{1,2\}$, as depicted in Figure 11(c).

Step 5 constructs arcs in each layer to represent potential acute-angle-free moves through a switch node in G.

Step 5: For each switch node $i \in S$, construct the four directed arcs

$$\bigcup_{k=2}^{3} \left\{ (i^{\langle\alpha,1,1\rangle}, i^{\langle\alpha,1,k\rangle}), (i^{\langle\alpha,2,k\rangle}, i^{\langle\alpha,2,1\rangle}) \right\}, \tag{7}$$

in each layer $\alpha \in \{1,2\}$, each with length zero.

Figure 12 depicts (using dashed black arcs) the subgraph of Figure 10 constructed in this step due to switch node 7.

6.3 Constructing Cross-Layer Arcs

Directed arcs are constructed to represent the potential for the entity to reverse direction at a switch node in G. Step 6 defines these arcs.

Step 6: For each switch node $i \in S^+$, construct the four directed arcs

$$\left\{ (i^{\langle 1,2,2\rangle}, i^{\langle 2,1,3\rangle}), (i^{\langle 1,2,3\rangle}, i^{\langle 2,1,2\rangle}), (i^{\langle 2,2,2\rangle}, i^{\langle 1,1,3\rangle}), (i^{\langle 2,2,3\rangle}, i^{\langle 1,1,2\rangle}) \right\},$$

in $\bar{A}$, each with length L.

In Figure 10, $\{2,7\} \subseteq S^+$ due to the SAAF walks 7–8 and 2–1. Figure 12 depicts (using dotted blue arcs) the subgraph of Figure 10 constructed in this step due to switch node 7.

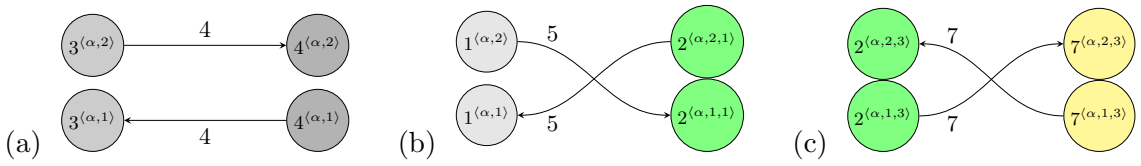

Figure 11: Illustration of the directed arcs in $\bar{A}$ constructed in Step 3 for (a) $e = [3, 4]$, (b) $e = [1, 2]$, and (c) $e = [2, 7]$. In each case, a copy of the depicted arcs are constructed for each $\alpha \in \{1, 2\}$ and comprise a subgraph of the fully expanded network in Figure 10.

6.4 Constructing Source- and Sink-Adjacent Arcs

Arcs are constructed from node $s \in \bar{N}$ and into node $t \in \bar{N}$ to impose orientation restrictions at the origin and destination nodes $f, g \in N$. Let

$$\beta' = \begin{cases} 1 & \text{if the locomotive faces } N_1(f) \text{ in the origin feasible position,} \\ 2 & \text{otherwise,} \end{cases}$$

and

$$\beta'' = \begin{cases} 1 & \text{if the locomotive faces } N_1(g) \text{ in the destination feasible position,} \\ 2 & \text{otherwise.} \end{cases}$$

Step 7 defines the arcs used to impose orientation restrictions at f and g.

Step 7: If the origin and destination are switch nodes (i.e., $f, g \in S$), construct the four directed arcs

$$\{(s, f^{\langle 1,\beta',1\rangle}), (s, f^{\langle 2,3-\beta',1\rangle}), (g^{\langle 1,\beta'',1\rangle}, t), (g^{\langle 2,3-\beta'',1\rangle}, t)\}, \tag{8}$$

in $\bar{A}$. If at least one of $\{f, g\}$ is not a switch node, modify Equation (8) by removing the k-index (if $f \notin S$) from the arcs leaving s (i.e., these arcs now connect s to $f^{\langle 1,\beta'\rangle}$ and $f^{\langle 2,3-\beta'\rangle}$) and/or (if $g \notin S$) from the arcs entering t (i.e., these arcs now connect $g^{\langle 1,\beta''\rangle}$ and $g^{\langle 2,3-\beta''\rangle}$ to t).

In Figure 10, the origin node is $f = 3$ and the destination node is $g = 6$, and the locomotive should face the first neighbor (i.e., $N_1(\cdot)$) at these nodes; therefore, $\beta' = \beta'' = 1$ and the arcs $\{(s, 3^{\langle 1,1\rangle}), (s, 3^{\langle 2,2\rangle}), (6^{\langle 1,1\rangle}, t), (6^{\langle 2,2\rangle}, t)\}$ are constructed. By modifying the nodes to (from) which s (t) is connected, this step can also accommodate the cases where the origin's (destination's) orientation is not specified or only its direction of arrival is specified.

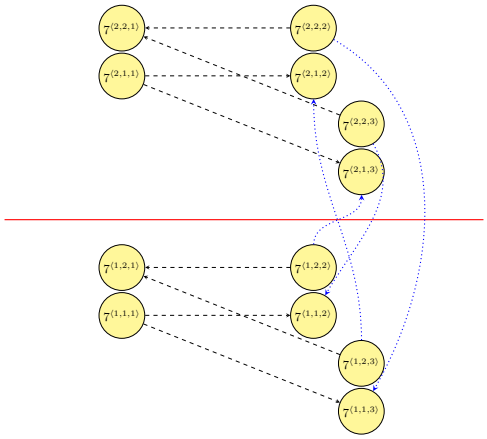

Figure 12: Illustration of the directed arcs in $\bar{A}$ constructed in Steps 4–5 for switch node $i = 7$. Dashed arcs (- - →) have length zero and dotted blue arcs (⋯→) have length L.

7 Computational Results

This section presents computational results for the two-stage routing approach to solve the SCRP. The computational experiments are performed on a real yard network data set, provided by CSX, with 4601 nodes, 4725 edges, and 287 switch nodes. As the length of the shortest AAF cycle in this network is 2380 units, we solve the SCRP only for values $L \leq 2380$ to ensure that the network is BCL (and the two-stage approach is valid). All tests were conducted on an Intel Core i7 CPU @ 2.93 GHz, 8 GB RAM computer using IBM ILOG CPLEX Optimization Studio 12.6 to solve all linear programs.

From the CSX data set, we randomly generated 400 origin-destination (o-d) node pairs, to which we then applied the two-stage routing approach for entity lengths $L \in \{100, 500, 1000, 2000\}$ under the objective function (OF1) of minimizing the penalized route length. The origin's and destination's orientation were not specified; that is, we allowed the entity to be oriented in either direction at the beginning and end of the route. The $4 \times 400 = 1600$ instances derived by using each o-d pair in combination with each value of L all share a common network topology and therefore require solving the same set of 287 PREPROCESS(·) LPs (i.e., $\text{LP}_{(j^\star)}$, $\forall j^\star \in S$). We solved this set of LPs only once, and the total solution time across LPs was 1.377 seconds. Using the results of the PREPROCESS(·) LPs, we then solved the resulting 1600 instances of the routing problem. We solved the routing instances by linear programming, using the classical shortest path LP (see, e.g., [2, pp. 5–6]) over the expanded directed network given in Section 6, and observed that the solution times were no more than 1 second across all of the instances. Noting that this solution

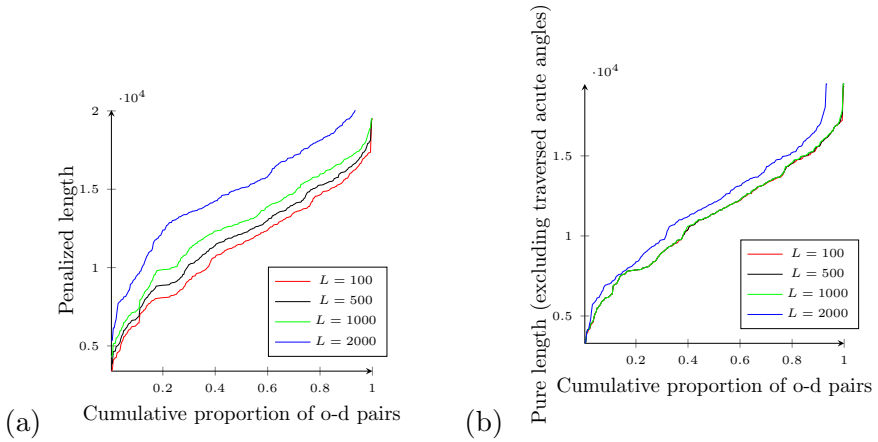

Figure 13: Impact of L on (a) penalized length and (b) pure length of the shortest route under objective OF1.

time was always smaller than the required preprocessing time, we decided to forego implementation of Dijkstra's algorithm for the routing stage. Among the original 400 o-d pairs, 275 produced at least one optimal route (across the four values of L) that included at least one traversal of an acute angle. The results for the $4 \times 275 = 1100$ instances corresponding to these o-d pairs are summarized in Figures 13–14.

Figures 13(a) and 13(b) show the impact of the increase in L on the optimal route's (a) penalized length and (b) *pure length* (i.e., the total length excluding penalties due to traversing acute angles). These two figures compare the cumulative proportion of instances within the y-axis value for different values of L, where instances to the right of where each series hits the plot's upper limit are infeasible. As can be seen in Figure 13(b), increasing the length of the entity up to 1000 units tends not to change the optimal route; however, when L increases to 2000 units, the previous optimal route is no longer feasible.

Figure 14(a) shows the number of times the entity traverses an acute angle under objective OF1, where instances to the right of where each series hits the plot's upper limit are infeasible. Even though on average this value decreases as the entity length increases, there exist some instances where this value increases depending on the origin and destination which is illustrated in Figure 14(b). In Figure 14(b), the two plots at each point in the x-axis corresponds to the same instance, and a value of -1 indicates infeasibility.

To investigate the impact of the penalty for traversing acute angles, we performed an additional set of experiments in which we re-solved the 275 instances with $L = 100$ under the objective function (OF2) of minimizing the number of traversed acute angles. Figures 15(a) and 15(b) compare OF1

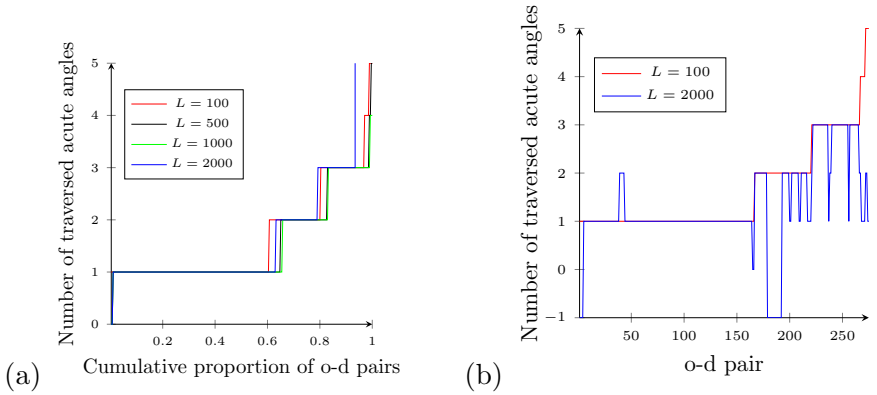

Figure 14: Impact of L on number of traversed acute angles under objective OF1. Subfigure (a) depicts the cumulative proportion of instances within a given number of traversed acute angles. Subfigure (b) depicts the number of traversed acute angles separately for each o-d pair.

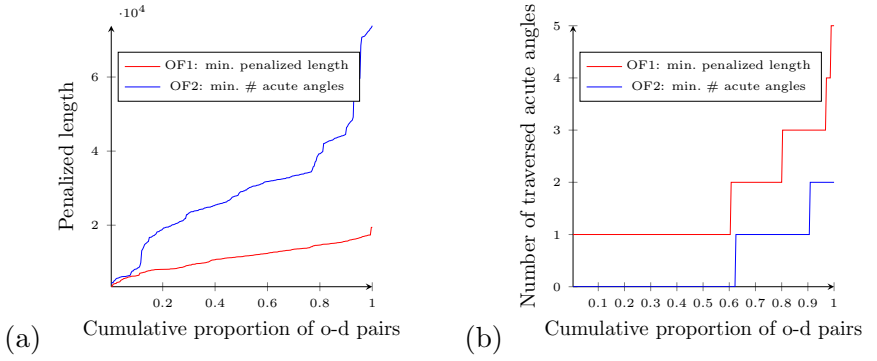

Figure 15: Impact of objective function on optimal route's (a) penalized length and (b) number of traversed acute angles for an entity with $L = 100$.

versus OF2 for the instances with $L = 100$, showing that is a significant trade-off between decreasing the number of traversed acute angles and decreasing the penalized length; that is, there may exist routes with few or no traversed angles, but these routes commonly have much greater penalized length than an optimal route under OF1. In practical situations where traversing acute angles is expensive or time consuming, it may therefore make sense to utilize an objective—such as OF2 or an extension of OF1 that increases the length assigned to those directed arcs constructed in Step 5 of Section 6—that prioritizes avoiding acute angle traversals..

In order to analyze the potential impacts due to idle cars congesting the yard, we generated a *congested yard topology* by independently removing each edge in the original network with probability 1%. Figure 16 compares the results for OF1 and $L = 100$ under the original and congested yard networks, where those cases in which the series hits the upper limit of the plot correspond to infeasible instances. The impact of congestion is pronounced for about 26% of the o-d pairs, causing

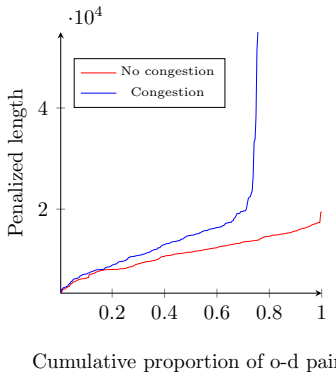

Figure 16: Impact of congestion on penalized length of the optimal route for an entity with $L = 100$.

SCRP to become infeasible in these cases; however, the increase in penalized length is modest for the remaining o-d pairs.

8 Conclusion

We have introduced, and contributed theory and algorithms for, the problem of identifying a shortest route for an entity (i.e., a locomotive and attached cut of cars) through a rail yard. The problem differs from previous routing problems in the sense that it considers restrictions imposed due to the geometry of the track segments and switch nodes and allows for specifying the entity's orientation upon arrival. We propose an algorithm, which decomposes the problem into PREPROCESS$(\cdot)$ and ROUTING$(\cdot)$ subproblems, and prove that the algorithm has polynomial complexity for specialized "BCL" yard network topology in which it is impossible for the entity to collide with itself. For general yard networks, we have also demonstrated that the PREPROCESS$(\cdot)$ problem (i.e., determining whether the yard will accommodate the entity with one of its ends positioned at a given node) is NP-complete.

Due to the operational nature of the problem, solution time is of utmost importance. Our computational experiments demonstrate the algorithm's speed and summarize solution properties for a real yard network.

In regards to algorithms for BCL networks (i.e., networks that satisfy Assumption 2), there is opportunity for improvement of our polynomial algorithm. Future research may seek to derive faster algorithms for either (i) the preprocessing stage (e.g., to avoid solving linear programs) or (ii) the routing stage (e.g., by exploiting our characterization of SCRP feasible solutions in Section 6 in order to operate on the smaller yard network instead of the expanded, directed network).

Additionally, future work may seek to address non-BCL networks (for which the polynomial algorithm fails). Given the characteristic sparsity of yard networks, we have not given up hope that this case can be solved (exactly) in a reasonable amount of time. A natural follow-on investigation would be to explore integer programming models and solution algorithms for the non-BCL case. Our own preliminary investigation of this direction has revealed challenges in formulating such a model compactly. It is interesting to note that such a formulation would likely encode, using binary variables, the entity's occupied position at discrete "steps" along the route. The set of potential positions occupied by the entity turns out to be equal to the union of the set of minimal SAAF($j_{[+]}$) and SAAF($j_{[-]}$) feasible positions across all switch nodes $j \in S$; thus, such an integer programming formulation might also lead to a polyhedral investigation of PREPROCESS($\cdot$).

An alternate course of research for the non-BCL case may pursue fast algorithms that do not guarantee optimality. For instance, one might seek to modify the yard network (e.g., by removing edges) in order to force satisfaction of Assumption 2, and then to apply the two-stage algorithm of this paper.

Finally, although our research considers a fundamental routing that arises in the context of rail yard operations, it does not fully address all sources of complexity in planning rail yard operations. Follow-on research may investigate the problem of simultaneously routing multiple entities through the yard and/or using multiple routes for a single entity by splitting it into shorter sub-entities.

9 Acknowledgments

The authors are thankful to Tom Anderson of the CSX Legal department for reviewing of the paper.

References

[1] R. K. Ahuja, K. C. Jha, and J. Liu. Solving real-life railroad blocking problems. *Interfaces*, 37(5):404–419, 2007.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[3] M. Aliakbari, J. Geunes, and K. M. Sullivan. A local search algorithm for train unit shunting with service scheduling. *Optimization Letters (accepted)*, 2021.

[4] C. Barnhart, H. Jin, and P. H. Vance. Railroad blocking: A network design application. *Operations Research*, 48(4):603–614, 2000.

[5] N. Bešinović and R. M. Goverde. Stable and robust train routing in station areas with balanced infrastructure capacity occupation. *Public Transport*, 11(2):211–236, 2019.

[6] R. Borndörfer, T. Klug, T. Schlechte, A. Fügenschuh, T. Schang, and H. Schülldorf. The freight train routing problem for congested railway networks with mixed traffic. *Transportation Science*, 50(2):408–423, 2016.

[7] V. Cacchiani, A. Caprara, and M. Fischetti. A Lagrangian heuristic for robustness, with an application to train timetabling. *Transportation Science*, 46(1):124–133, 2012.

[8] V. Cacchiani, L. Galli, and P. Toth. A tutorial on non-periodic train timetabling and platforming problems. *Euro Journal on Transportation and Logistics*, 4(3):285–320, 2015.

[9] G. Caimi, D. Burkolter, and T. Herrmann. Finding delay-tolerant train routings through stations. In F. H, d. D, and K. P, editors, *Operations Research Proceedings 2004*, pages 136–143. Springer, 2005.

[10] G. Caimi, F. Chudak, M. Fuchsberger, M. Laumanns, and R. Zenklusen. A new resource-constrained multicommodity flow model for conflict-free train routing and scheduling. *Transportation Science*, 45(2):212–227, 2011.

[11] A. Caprara, M. Fischetti, and P. Toth. Modeling and solving the train timetabling problem. *Operations Research*, 50(5):851–861, 2002.

[12] A. Caprara, L. Galli, L. Kroon, G. Maróti, and P. Toth. Robust train routing and online re-scheduling. In *10th Workshop on algorithmic approaches for transportation modelling, optimization, and systems (ATMOS'10)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2010.

[13] M. Carey and S. Carville. Scheduling and platforming trains at busy complex stations. *Transportation Research Part A: Policy and Practice*, 37(3):195–224, 2003.

[14] F. Corman, R. M. Goverde, and A. D'Ariano. Rescheduling dense train traffic over complex station interlocking areas. In A. RK, R. Möhring, and Z. CD, editors, *Robust and online large-scale optimization*, pages 369–386. Springer, 2009.

[15] A. D'Ariano, F. Corman, D. Pacciarelli, and M. Pranzo. Reordering and local rerouting strategies to manage train traffic in real time. *Transportation Science*, 42(4):405–419, 2008.

[16] T. Dewilde, P. Sels, D. Cattrysse, and P. Vansteenwegen. Improving the robustness in railway station areas. *European Journal of Operational Research*, 235(1):276–286, 2014.

[17] R. Freling, R. M. Lentink, L. G. Kroon, and D. Huisman. Shunting of passenger train units in a railway station. *Transportation Science*, 39(2):261–272, 2005.

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

[19] R. Haijema, C. Duin, and N. M. Van Dijk. Train shunting: A practical heuristic inspired by dynamic programming. *Planning in intelligent systems: aspects, motivations, and methods*, pages 437–475, 2006.

[20] K. K. Hendrikse. Branch-and-cut-and-price to solve a relaxation of the train unit shunting and service problem. Master's thesis, Erasmus Universiteit Rotterdam, 2021.

[21] T. M. Herrmann. *Stability of timetables and train routings through station regions*. PhD thesis, ETH Zurich, 2006.

[22] D. Huisman, L. G. Kroon, R. M. Lentink, and M. J. Vromans. Operations research in passenger railway transportation. *Statistica Neerlandica*, 59(4):467–497, 2005.

[23] J. G. Jin, J. Zhao, and D.-H. Lee. A column generation based approach for the train network design optimization problem. *Transportation Research Part E: Logistics and Transportation Review*, 50:1–17, 2013.

[24] L. G. Khachiyan. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1):53–72, 1980.

[25] L. Kroon, G. Maróti, M. R. Helmrich, M. Vromans, and R. Dekker. Stochastic improvement of cyclic railway timetables. *Transportation Research Part B: Methodological*, 42(6):553–570, 2008.

[26] L. G. Kroon, R. M. Lentink, and A. Schrijver. Shunting of passenger train units: an integrated approach. *Transportation Science*, 42(4):436–449, 2008.

[27] R. M. Lentink, P.-J. Fioole, L. G. Kroon, and C. Van't Woudt. Applying operations research techniques to planning of train shunting. *Planning in Intelligent Systems: Aspects, Motivations, and Methods*, pages 415–436, 2006.

[28] R. Lusby, J. Larsen, D. Ryan, and M. Ehrgott. Routing trains through railway junctions: a new set-packing approach. *Transportation Science*, 45(2):228–245, 2011.

[29] R. M. Lusby, J. Larsen, M. Ehrgott, and D. Ryan. Railway track allocation: models and methods. *OR Spectrum*, 33(4):843–883, 2011.

[30] R. M. Lusby, J. Larsen, M. Ehrgott, and D. M. Ryan. A set packing inspired method for real-time junction train routing. *Computers & Operations Research*, 40(3):713–724, 2013.

[31] J. Mulderij, B. Huisman, D. Tönissen, K. van der Linden, and M. de Weerdt. Train unit shunting and servicing: a real-life application of multi-agent path finding. *arXiv preprint arXiv:2006.10422*, 2020.

[32] P. Pellegrini, G. Marlière, and J. Rodriguez. Optimal train routing and scheduling for managing traffic perturbations in complex junctions. *Transportation Research Part B: Methodological*, 59:58–80, 2014.

[33] M. Sama, P. Pellegrini, A. D'Ariano, J. Rodriguez, and D. Pacciarelli. Ant colony optimization for the real-time train routing selection problem. *Transportation Research Part B: Methodological*, 85:89–108, 2016.

[34] M. Sama, P. Pellegrini, A. D'Ariano, J. Rodriguez, and D. Pacciarelli. On the tactical and operational train routing selection problem. *Transportation Research Part C: Emerging Technologies*, 76:1–15, 2017.

[35] P. Sels, P. Vansteenwegen, T. Dewilde, D. Cattrysse, B. Waquet, and A. Joubert. The train platforming problem: The infrastructure management company perspective. *Transportation Research Part B: Methodological*, 61:55–72, 2014.

[36] Union Pacific. Bailey Yard. `https://www.up.com/aboutup/facilities/bailey_yard`. Accessed: 2019/03/06.

[37] A. van de Ven, Y. Zhang, W.-J. Lee, R. Eshuis, and A. Wilbik. Determining capacity of shunting yards by combining graph classification with local search. In *ICAART (2)*, pages 285–293, 2019.

[38] M. Van Den Akker, H. Baarsma, J. Hurink, M. Modelski, J. Jan Paulus, I. Reijnen, D. Roozemond, and J. Schreuder. Shunting passenger trains: getting ready for departure. Technical report, 2008.

[39] R. van den Broek, H. Hoogeveen, M. van den Akker, and B. Huisman. A local search algorithm for train unit shunting with service scheduling. Technical report, 2021.

[40] R. W. van den Broek. Train shunting and service scheduling: an integrated local search approach. Master's thesis, 2016.

[41] M. Yaghini, M. Momeni, and M. Sarmadi. An improved local branching approach for train formation planning. *Applied Mathematical Modelling*, 37(4):2300–2307, 2013.

[42] Q. Zhang, X. Zhu, L. Wang, and S. Wang. Simultaneous optimization of train timetabling and platforming problems for high-speed multiline railway network. *Journal of Advanced Transportation*, 2021, 2021.

[43] P. J. Zwaneveld, L. G. Kroon, H. E. Romeijn, M. Salomon, S. Dauzere-Peres, S. P. Van Hoesel, and H. W. Ambergen. Routing trains through railway stations: Model formulation and algorithms. *Transportation Science*, 30(3):181–194, 1996.

[44] P. J. Zwaneveld, L. G. Kroon, and S. P. Van Hoesel. Routing trains through a railway station based on a node packing model. *European Journal of Operational Research*, 128(1):14–33, 2001.

A Appendix: Proof of Theorem 2

Towards proving this theorem, it will be useful to define two subroutines that contract the network in a way that preserves both yard structure and Assumptions 2–3.

CONTRACT-1(i)

> **Given:** A leaf node i that is the *first neighbor* of switch node $i' \in S$, i.e., $i = N_1(i')$ and $e(i', 1) = [i, i']$.
>
> **Procedure:** Remove the leaf node i, the switch node i', and all edges adjacent to i'. For $k \in \{2, 3\}$, add a new leaf node i_k and a new edge $[N_k(i'), i_k]$ with length $c_{e(i',1)} + c_{e(i',k)}$. Let $i_k,\ k \in \{2, 3\}$ replace i' in the ordered adjacency list of node $N_k(i')$.

CONTRACT-2(i)

> **Given:** A leaf node i that is the *second or third neighbor* of switch node $i' \in S$, i.e., $i = N_k(i')$ and $e(i', k) = [i, i']$ for some $k \in \{2, 3\}$. Let k' denote the (unique) element of $\{2, 3\} \setminus \{k\}$.
>
> **Procedure:** Remove the leaf node i, the switch node i', and all edges adjacent to i'. Add a new edge $[N_1(i'), N_{k'}(i')]$ with length $c_{e(i',1)} + c_{e(i',k')}$. Let $N_1(i')$ and $N_{k'}(i')$ respectively replace i' in the ordered adjacency list of nodes $N_{k'}(i')$ and $N_1(i')$.

Figure 17 illustrates the yard network that results upon executing CONTRACT-2(1) and CONTRACT-1(4) in sequence. We now state an additional definition required in order to prove (in Lemmas 1–3) fundamental properties of these subroutines.

Definition 14 A SAAF($j_{[+]}$) (respectively, SAAF($j_{[-]}$)) walk ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$ is said to be *maximal* if there does not exist a node $j' \in N$ such that $i(0)$–$i(1)$–$\cdots$–$i(m)$–j' is a SAAF($j_{[+]}$) (respectively, SAAF($j_{[-]}$)) walk. For example, 16–15–6–5–2–3–4–5 and 16–29–30–18 are respectively maximal SAAF($16_{[+]}$) and maximal SAAF($16_{[-]}$) walks in Figure 2; however, 16–15–6–5 is not a maximal SAAF($16_{[+]}$) walk because it can be extended as 16–15–6–5–4 or 16–15–6–5–2 into another SAAF($16_{[+]}$) walk.

Lemma 1 *Let i be any leaf node such that $N_1(i') = i$ for some switch node $i' \in S$. Let $j \in S \setminus \{i'\}$, and let G' denote the network that results upon executing CONTRACT-1(i). Then, (a) the maximum length among all SAAF($j_{[+]}$) walks is the same in G and G', and (b) the maximum length among all SAAF($j_{[-]}$) walks is the same in G and G'.*

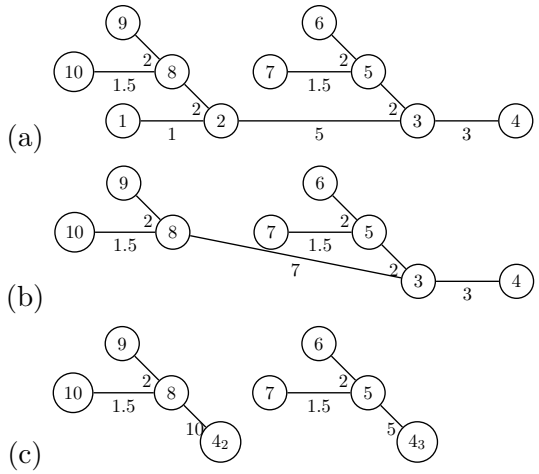

Figure 17: (a) Notional yard network; (b) Resulting yard network after applying CONTRACT-2(1); (c) Resulting yard network after applying CONTRACT-1(4) to the yard network from (b).

PROOF We prove part (a) by establishing that there exists a maximal SAAF($j_{[+]}$) walk of length l in G if ($\leftarrow$) and only if ($\rightarrow$) there exists a maximal SAAF($j_{[+]}$) walk of length l in G'. Because $c_e \geq 0,\ \forall e \in E$, there is always a maximal SAAF($j_{[+]}$) walk among the set of maximum-length SAAF($j_{[+]}$) walks, and this will therefore establish the result.

Proof of ($\rightarrow$): Let W, with nodes ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$, be a maximal SAAF($j_{[+]}$) walk of length l in G. If W includes node i', then (because W is maximal) W must also include i, and we therefore have the following two cases: (I) W includes both node i' and node i; and (II) W includes neither node i' nor node i.

Proof of ($\rightarrow$, Case I): Suppose W includes both node i and i'. Because i is a leaf node, we may assume $i(m) = i$, $i(m-1) = i'$, and $i(m-2) = N_k(i')$ for some $k \in \{2, 3\}$. Because W is a SAAF walk, only its last node may be repeated; therefore, i' must be distinct from $i(0), i(1), \ldots, i(m-2)$. Therefore, after CONTRACT-1(i), the sub-walk ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m-2)$ ($= N_k(i')$) remains a SAAF($j_{[+]}$) walk in G' and its length is $l - c_{e(i',k)} - c_{e(i',1)}$ in both G and G'. Observe that CONTRACT-1(i) has inserted in G' the new leaf node i_k and the new edge $[N_k(i'), i_k]$ with length $c_{e(i',k)} + c_{e(i',1)}$; thus, $i(0)$–$i(1)$–$\cdots$–$i(m-2)$ can be extended in G' to $i(0)$–$i(1)$–$\cdots$–$i(m-2)$–i_k, a SAAF($j_{[+]}$) walk with length $l - c_{e(i',k)} - c_{e(i',1)} + (c_{e(i',k)} + c_{e(i',1)}) = l$. Because i_k is a leaf node, this walk is maximal in G' and case I is proved.

Proof of ($\rightarrow$, Case II): Suppose W includes neither node i nor node i'. Note that all edges removed by CONTRACT-1(i) were incident to node i'. Therefore, $i(0)$–$i(1)$–$\cdots$–$i(m)$ remains a SAAF($j_{[+]}$) walk in G', and its length remains l. Furthermore, if i' is adjacent to $i(m)$, it must

be the case that $[i(m-1), i(m)]$ and $[i(m), i']$ form an acute angle. Therefore, if CONTRACT-1(i) added any new edges that are adjacent to $i(m)$, that edge will also form an acute angle with $[i(m-1), i(m)]$. Consequently, W remains maximal in G', thus completing the proof of this direction.

Proof of ($\leftarrow$): Let W, with nodes ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$, be a maximal SAAF($j_{[+]}$) walk of length l in G'. We prove that G contains a maximal SAAF($j_{[+]}$) walk of length l in two cases: (I) W includes either node i_2 or node i_3; and (II) W includes neither node i_2 nor i_3.

Proof of ($\leftarrow$, Case I): Suppose without loss of generality that W includes the new node i_2 added in CONTRACT-1(i). (The proof is symmetric if i_2 is replaced by i_3.) In G', i_2 is adjacent only to node $N_2(i')$. Because i_2 is a leaf node in G', W must also include node $N_2(i')$ such that $i(m-1) = N_2(i')$ and $i(m) = i_2$. Removing $i(m)$ from W yields the SAAF($j_{[+]}$) walk ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m-1)$, which is contained in both G' and G and has length $l - c_{e(i',1)} - c_{e(i',2)}$. In G, node i' occupies the same position in the ordered adjacency list of node $N_2(i')$ as did node i_2 in G'; therefore, because i' is not in G' (and is therefore distinct from $i(0), i(1), \ldots, i(m-1)$), ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m-1)$–i' is a SAAF($j_{[+]}$) walk of length $l - c_{e(i',1)}$ in G. Similarly, $i \neq i'$ is not in G' and we therefore have that ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m-1)$–i'–i is a SAAF($j_{[+]}$) walk of length l, which is also maximal because i is a leaf node. This completes the proof of this case.

Proof of ($\leftarrow$, Case II): Suppose W includes neither node i_2 nor i_3. Note that all edges removed by CONTRACT-1(i) were incident to node i'. Therefore, $i(0)$–$i(1)$–$\cdots$–$i(m)$ remains a SAAF($j_{[+]}$) walk in G, and its length remains l.

The proof of part (b) is identical to part (a) after replacing "SAAF($j_{[+]}$)" throughout with "SAAF($j_{[-]}$)." ■

In the absence of AAF cycles, we have the following additional characterization of maximal SAAF($j_{[+]}$) and maximal SAAF($j_{[-]}$) walks.

Lemma 2 *If G contains no AAF cycles, then the following statement is true for every switch node $j \in S$: every maximal SAAF($j_{[+]}$) walk and every maximal SAAF($j_{[-]}$) walk ends at a leaf node.*

Proof We prove the statement only for maximal SAAF($j_{[+]}$) walks because the proof is analogous for maximal SAAF($j_{[-]}$) walks. Let ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$ denote a maximal SAAF($j_{[+]}$) walk. If $i(m)$ is not a leaf node, then by Assumption 3, $i(m)$ is a switch node; thus, there exists a node $j' \in N$ such that the edge pair $\{[i(m-1), i(m)], [i(m), j']\}$ is not an acute angle—if $i(m-1) = N_1(i(m))$, then $j' = N_2(i(m))$; else $j' = N_1(i(m))$. If $j' = i(h)$ for some $h \in \{0, 1, \ldots, m-1\}$, then we

have identified the AAF cycle $i(h)$–$i(h+1)$–$\cdots$–$i(m)$–j' (a contradiction with the assumption that there are no AAF cycles); else, $i(0)$–$i(1)$–$\cdots$–$i(m)$–j' is a SAAF($j_{[+]}$) walk (a contradiction because $i(0)$–$i(1)$–$\cdots$–$i(m)$ is a maximal SAAF($j_{[+]}$) walk). ■

Using Lemma 2, we may prove (in the vein of Lemma 1 for CONTRACT-1($\cdot$)) conditions under which CONTRACT-2($\cdot$) preserves γ_i and $\overline{\gamma}_i$.

Lemma 3 *Let i be a leaf node with the smallest value of $c_{e(i,1)}$. (That is, i is a leaf node with the shortest-length adjacent edge.) If $i \in \{N_2(i'), N_3(i')\}$ for some $i' \in S$, then let G' denote the network that results upon executing CONTRACT-2(i). For any $j \in S \backslash \{i'\}$ (a) the maximum length of any SAAF($j_{[+]}$) walk is the same in G and G', and (b) the maximum length of any SAAF($j_{[-]}$) walk is the same in G and G'.*

Proof We will assume without loss of generality that $i = N_3(i')$ (i.e., $k = 3$ and $k' = 2$ in the definition of CONTRACT-2(i)) as the proof for $i = N_2(i')$ is symmetric. We prove part (a) by establishing that ($\rightarrow$) for every maximum-length, maximal SAAF($j_{[+]}$) walk in G, there exists a SAAF($j_{[+]}$) walk of the same length in G', and ($\leftarrow$) for every maximum-length, maximal SAAF($j_{[+]}$) walk in G', there exists a SAAF($j_{[+]}$) walk of the same length in G.

Proof of ($\rightarrow$): Let W, with node representation ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$, denote a maximum-length, maximal SAAF($j_{[+]}$) walk in G. We prove this result in three cases: (I) W includes neither i' nor i, (II) W includes i' but not i, and (III) W includes both i' and i.

Proof of ($\rightarrow$, Case I): If W includes neither i' nor i, then W is a SAAF($j_{[+]}$) walk with the same length in G', and there is nothing to prove.

Proof of ($\rightarrow$, Case II): If W includes i' but not i, then let $h \in \{1, 2, \ldots, m-1\}$ denote the index such that $i(h) = i'$. (Note $h \neq 0$ because j is distinct from i' by assumption in the lemma's statement, and $h \neq m$ because we know that maximal SAAF($j_{[+]}$) walks end at a leaf nodes but $i' \in S$.) Because $i = N_3(i')$ is not traversed, we know that either $i(h-1) = N_1(i')$ and $i(h+1) = N_2(i')$ or $i(h-1) = N_2(i')$ and $i(h+1) = N_1(i')$. Therefore, the total length of the edges $[i(h-1), i(h)]$ and $[i(h), i(h+1)]$ is $c_{e(i',1)} + c_{e(i',2)}$. As a consequence of CONTRACT-2(i), nodes $i(h-1)$ and $i(h+1)$ are neighbors in G' such that ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(h-1)$–$i(h+1)$–$i(h+2)$–$\cdots$–$i(m)$ is a SAAF($j_{[+]}$) walk in G'. Because the new edge $[i(h-1), i(h+1)]$ in G' has length $c_{e(i',1)} + c_{e(i',2)}$, this SAAF($j_{[+]}$) walk has the same length as W.

Proof of ($\rightarrow$, Case III): If W includes both i' and i, then we must have that $i(m-1) = i'$ and $i(m) = i$—in this case we could construct another maximal SAAF($j_{[+]}$) walk W' by replacing

$[i(m-1), i(m)] = e(i', 3)$ in W with $e(i', 2)$ and then iteratively extending the SAAF($j_{[+]}$) walk (as in the proof of Lemma 2 until it is no longer possible to add additional edges. By Lemma 2, W' ends at a leaf node; furthermore, because i is a leaf node with the smallest value of $c_{e(i,1)}$, the last edge of W' must have length at least $c_{e(i,1)}$. Because the edge $e(i, 1)$ is the only edge in W that is not in W', we have that W' is also a maximum-length, maximal SAAF($j_{[+]}$) walk. Case III is therefore proven as a consequence of Case II, thereby proving ($\rightarrow$).

Proof of ($\leftarrow$): Let W, with nodes ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$, denote a maximum-length, maximal SAAF($j_{[+]}$) walk in G'. Let e' denote the unique edge added to G' by CONTRACT-2(i). If e' is not traversed by W, then W is also a SAAF($j_{[+]}$) walk in G, and its length remains the same. We therefore assume e' is traversed by W, i.e., there exists $h \in \{1, \ldots, m\}$ such that $e' = [i(h-1), i(h)]$, where $c_{e'} = c_{e(i',1)} + c_{e(i',2)}$. In this case, ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(h-1)$–i'–$i(h)$–$i(h+1)$–$\cdots$–$i(m)$ is a SAAF($j_{[+]}$) walk in G, and its length is the same in G' as in G. This proves ($\leftarrow$) and part (a).

The proof of part (b) is identical to part (a) after replacing "SAAF($j_{[+]}$)" throughout with "SAAF($j_{[-]}$)." ∎

The combination of Lemmas 1–3 yields a procedure for decomposing the yard network, one switch node at a time, in such a way that preserves γ_j and $\overline{\gamma}_j$. This procedure will yield a proof (see Theorem 2) of correctness for LP (3). The following additional lemmas will be used in proving the correctness of LP (3).

Lemma 4 *If G contains no AAF cycles and $S \neq \emptyset$, then G contains a leaf node i that is adjacent to a switch node (say $i' \in S$) such that executing either CONTRACT-1(i) or CONTRACT-2(i) yields a new network G' that satisfies the following property: For every $j \in S \setminus \{i'\}$, (a) the maximum length among all SAAF($j_{[+]}$) walks is the same in G and G', and (b) the maximum length among all SAAF($j_{[-]}$) walks is the same in G and G'.*

Proof This result follows as a direct consequence of Lemmas 1 and 3 upon selecting i as a leaf node that has the shortest adjacent edge. Based on Lemma 2, the leaf node i exists in G since it contains no AAF cycles. If $i = N_1(i')$, Lemma 1 provides the result upon executing CONTRACT-1(i); otherwise, Lemma 3 provides the result upon executing CONTRACT-2(i). ∎

Lemma 5 *Suppose ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$ is a maximum-length SAAF($j_{[+]}$) walk in G, which contains no AAF cycles. Then for every $h = 1, \ldots, m-1$, the sub-walk $i(h)$–$i(h+1)$–$\cdots$–$i(m)$ is a maximum-length SAAF($i(h)_{[+]}$) walk (if $i(h+1) = N_1(i(h))$) and a maximum-length SAAF($i(h)_{[-]}$) walk otherwise.*

PROOF Suppose that $i(h+1) = N_1(i(h))$. By contradiction, let $(i(h) =)\ i'(0)$–$i'(1)$–$\cdots$–$i'(m')$ be a SAAF$(i(h)_{[+]})$ walk that is strictly longer than the SAAF$(i(h)_{[+]})$ walk $i(h)$–$i(h+1)$–$\cdots$–$i(m)$. We have either (I) $i(0)$–$i(1)$–$\cdots$–$i(h)$–$i'(1)$–$i'(2)$–$\cdots$–$i'(m')$ is a SAAF$(j_{[+]})$ walk or (II) the node sets $\{i(0), i(1), \ldots, i(h)\}$ and $\{i'(1), i'(2), \ldots, i'(m')\}$ overlap. In case (I), the new walk must be longer than $i(0)$–$i(1)$–$\cdots$–$i(m)$ because $i'(0)$–$i'(1)$–$\cdots$–$i'(m')$ is longer than $i(h)$–$i(h+1)$–$\cdots$–$i(m)$ (a contradiction). Case (II) is also a contradiction because it implies existence of an AAF cycle, $i(\ell)$–$i(\ell+1)$–$\cdots$–$i(h)$–$i'(1)$–$i'(2)$–$\cdots$–$i'(\ell')$, where $\ell' \in \{1, \ldots, m'\}$ is the smallest index such that $i'(\ell') \in \{i(1), i(2), \ldots, i(h)\}$. ■

The following definitions will aid in proving the correctness of LP (3). For $j \in S$, define

$$Q(j) \equiv \{i \in S :\ \text{there exists a SAAF}(i_{[+]}) \text{ walk that includes } j\}, \tag{9a}$$

$$\bar{Q}(j) \equiv \{i \in S :\ \text{there exists a SAAF}(i_{[-]}) \text{ walk that includes } j\}. \tag{9b}$$

Note that $Q(j) \cap \bar{Q}(j) \neq \emptyset$ implies existence of a closed AAF walk (by merging the SAAF$(i_{[+]})$ and SAAF$(i_{[-]})$ walk), which therefore implies existence of an AAF cycle. We may therefore assume $Q(j) \cap \bar{Q}(j) = \emptyset,\ \forall j \in S$. Further, because j is reachable along a SAAF$(i_{[+]})$ or SAAF$(i_{[-]})$ walk if and only if i is reachable along a SAAF$(j_{[+]})$ or SAAF$(j_{[-]})$ walk, $Q(j) \cup \bar{Q}(j)$ must equal the set of switch nodes reachable from node j.

We now build towards a proof that an optimal solution of the LP (3) establishes the value of γ_i. Towards this end, we first establish the impact of the subroutines CONTRACT-1$(\cdot)$ and CONTRACT-2$(\cdot)$ on LP (3). Both CONTRACT-1$(\cdot)$ and CONTRACT-2$(\cdot)$ will remove a single switch node from the network. With reference to the removed switch node i', it will simplify exposition throughout the remainder of this section to rename variables $(x_i, y_i),\ i \in S \setminus \{i'\}$ as $(w_i^{(i')}, z_i^{(i')}),\ i \in S \setminus \{i'\}$ as follows:

$$\text{For } i' \in S,\ i \in Q(i')\text{: } w_i^{(i')} \equiv y_i \text{ and } z_i^{(i')} \equiv x_i, \tag{10a}$$

$$\text{For } i' \in S,\ i \in \bar{Q}(i')\text{: } w_i^{(i')} \equiv x_i \text{ and } z_i^{(i')} \equiv y_i. \tag{10b}$$

Lemma 6 *Let i be a leaf node that is the first neighbor of a switch node i', i.e., $i = N_1(i')$. Let G denote the original network, and let G' denote the network after CONTRACT-1(i) has been executed. Similarly, let LP(G) and LP(G') denote the LP (3) under each network, and let $\Gamma(G)$ and $\Gamma(G')$ denote the respective sets of variables associated with these LPs. Then LP(G') is a restriction*

of LP(G) in the sense that (a) $\Gamma(G') \subseteq \Gamma(G)$ and (b) any feasible solution to LP(G′) can be extended into a feasible solution for LP(G) by assigning values to the variables in $\Gamma(G) \setminus \Gamma(G')$.

PROOF Let i' denote the switch node that was removed by CONTRACT-1(i). The set of variables associated with LP(G') are $(x_j, y_j),\ j \in S \setminus \{i'\}$, while the set of variables associated with LP(G) are $(x_j, y_j),\ j \in S$. This proves part (a) as $\Gamma(G') \cup \{x_{i'}, y_{i'}\} = \Gamma(G)$. Consider a solution $(\bar{x}_j, \bar{y}_j),\ j \in S \setminus \{i'\}$ to $LP(G')$, and according to Equations (10) let $(\bar{w}_j^{(i')}, \bar{z}_j^{(i')}),\ j \in S \setminus \{i'\}$, denote the corresponding values of $(w_j^{(i')}, z_j^{(i')})$. We now establish that this solution can be extended into a solution $(\bar{x}_j, \bar{y}_j),\ j \in S$, for $LP(G)$ by assigning $\bar{y}_{i'} = c_{e(i',1)}$ and $\bar{x}_{i'} = \max\{\bar{z}^{(i')}_{N_2(i')} + c_{e(i',2)}, \bar{z}^{(i')}_{N_3(i')} + c_{e(i',3)}\}$.

Assuming neither $N_2(i')$ nor $N_3(i')$ is a leaf node (with a description of the simplification in these special cases to follow), the constraints in LP(G) but not LP(G') are

$$y_{i'} \geq c_{e(i',1)}, \tag{11a}$$

$$x_{i'} \geq c_{e(i',2)} + z^{(i')}_{N_2(i')}, \tag{11b}$$

$$w^{(i')}_{N_2(i')} \geq c_{e(i',2)} + y_{i'}, \tag{11c}$$

$$x_{i'} \geq c_{e(i',3)} + z^{(i')}_{N_3(i')}, \tag{11d}$$

$$w^{(i')}_{N_3(i')} \geq c_{e(i',3)} + y_{i'}; \tag{11e}$$

it suffices to prove that each of these constraints is satisfied by $(\bar{x}_j, \bar{y}_j),\ j \in S$, along with the corresponding values $(\bar{w}_j^{(i')}, \bar{z}_j^{(i')}),\ j \in S \setminus \{i'\}$. Constraint (11a) is satisfied at equality by $\bar{y}_{i'} = c_{e(i',1)}$. Constraints (11b) and (11d) are satisfied because $\bar{x}_{i'} = \max\{\bar{z}^{(i')}_{N_2(i')} + c_{e(i',2)}, \bar{z}^{(i')}_{N_3(i')} + c_{e(i',3)}\}$. Feasibility to LP($G'$) implies $\bar{w}^{(i')}_{N_2(i')} \geq c_{e(i',2)} + c_{e(i',1)}$ and $\bar{w}^{(i')}_{N_3(i')} \geq c_{e(i',3)} + c_{e(i',1)}$, respectively corresponding to the switch node $N_2(i')$ and $N_3(i')$ in the direction of the new leaf nodes constructed by CONTRACT-1(i). Because $\bar{y}_{i'} = c_{e(i',1)}$, Constraints (11c) and (11e) must therefore be satisfied. If $N_2(i')$ is a leaf node, the proof is modified by setting $\bar{z}^{(i')}_{N_2(i')} = 0$ and removing Constraint (11c); likewise, if $N_3(i')$ is a leaf node, set $\bar{z}^{(i')}_{N_3(i')} = 0$ and remove Constraint (11e). Therefore, the solution $(\bar{x}_j, \bar{y}_j),\ j \in S$, satisfies all constraints that are included in LP(G) but not not LP(G') and must be feasible to LP(G). This completes the proof of part (b). ∎

Lemma 7 *Let i be a leaf node with the smallest value of $c_{e(i,1)}$. (That is, i is a leaf node with the shortest-length adjacent edge.) If $i \in \{N_2(i'), N_3(i')\}$ for some $i' \in S$, then let G' denote the network that results upon executing CONTRACT-2(i). Let LP(G) and LP(G′) denote LP* (3)

under the original and contracted network, and let $\Gamma(G)$ *and* $\Gamma(G')$ *denote their respective sets of variables. Then LP(*G'*) is a restriction of LP(*G*) in the sense that (a)* $\Gamma(G') \subseteq \Gamma(G)$ *and (b) any feasible solution to LP(*G'*) can be extended into a feasible solution for LP(*G*) by assigning values to the variables in* $\Gamma(G) \setminus \Gamma(G')$.

PROOF With the exception of node i', all switch nodes from G remain switch nodes in G'; therefore, $\Gamma(G') \cup \{x_{i'}, y_{i'}\} = \Gamma(G)$, proving part (a).

We now prove part (b). Towards this end, we assume without loss of generality that $i = N_3(i')$ as the proof for $i = N_2(i')$ is symmetric. Let $(\bar{x}_j, \bar{y}_j),\ j \in S \setminus \{i'\}$, denote any feasible solution to LP(G'). In accordance with Equations (10), let $(\bar{w}_j^{(i')}, \bar{z}_j^{(i')}),\ j \in S \setminus \{i'\}$ denote the corresponding values of $(w_j^{(i')}, z_j^{(i')})$ under solution $(\bar{x}_j, \bar{y}_j),\ j \in S \setminus \{i'\}$. Observe for $k \in \{1, 2\}$ that $\bar{z}_{N_k(i')}^{(i')} = \bar{x}_{N_k(i')}$ and $\bar{w}_{N_k(i')}^{(i')} = \bar{y}_{N_k(i')}$ if $N_1(N_k(i')) = i'$; otherwise, $\bar{z}_{N_k(i')}^{(i')} = \bar{y}_{N_k(i')}^{(i')}$ and $\bar{w}_{N_k(i')}^{(i')} = \bar{x}_{N_k(i')}$.

To prove part (b), it suffices to prove that $(x_{i'}, y_{i'})$ can be assigned values $(\bar{x}_{i'}, \bar{y}_{i'})$ such that the collection $(\bar{x}_j, \bar{y}_j),\ j \in S$, satisfies the set of constraints

$$x_{i'} \geq c_{e(i',3)}, \tag{12a}$$

$$y_{i'} \geq z_{N_1(i')}^{(i')} + c_{e(i',1)}, \tag{12b}$$

$$w_{N_1(i')}^{(i')} \geq x_{i'} + c_{e(i',1)}, \tag{12c}$$

$$x_{i'} \geq z_{N_2(i')}^{(i')} + c_{e(i',2)}, \tag{12d}$$

$$w_{N_2(i')}^{(i')} \geq y_{i'} + c_{e(i',2)}, \tag{12e}$$

that are included in LP(G) but not in LP(G'). (Note: Above we have assumed $N_1(i')$ and $N_2(i')$ are not leaf nodes. We summarize at the end of this proof the modifications that are necessary if this is not the case.)

We claim that under the assignment $\bar{x}_{i'} = \bar{z}_{N_2(i')}^{(i')} + c_{e(i',2)}$ and $\bar{y}_{i'} = \bar{z}_{N_1(i')}^{(i')} + c_{e(i',1)}$, the solution $(\bar{x}_j, \bar{y}_j),\ j \in S$, satisfies System (12) and therefore proves the result. This assignment immediately satisfies Constraints (12b) and (12d) at equality. Additionally, due to Constraints (3c), (3d), (3f), and (3g) of LP(G') corresponding to node $i = N_1(i')$ in the direction of node $N_2(i')$, we have that

$$\bar{w}_{N_1(i')}^{(i')} \geq \bar{z}_{N_2(i')}^{(i')} + c_{e(i',1)} + c_{e(i',2)}, \tag{13}$$

and corresponding to node $i = N_2(i')$ in the direction of node $N_1(i')$, we have that

$$\bar{w}^{(i')}_{N_2(i')} \geq \bar{z}^{(i')}_{N_1(i')} + c_{e(i',1)} + c_{e(i',2)}. \tag{14}$$

Therefore, upon substituting $\bar{x}_{i'} = \bar{z}^{(i')}_{N_2(i')} + c_{e(i',2)}$ into (13) and $\bar{y}_{i'} = \bar{z}^{(i')}_{N_1(i')} + c_{e(i',1)}$ into (14), Constraints (12c) and (12e) are satisfied.

To complete the proof, we now show that Constraint (12a) must be dominated by the remaining constraints of LP(G) and therefore satisfied by $(\bar{x}_j, \bar{y}_j),\ j \in S$. To see this, let $(i' =)\ i(0)$–$i(1)$–$\cdots$—$i(m)$ (with edge representation $e(1)$–$e(2)$–$\cdots$–$e(m)$) denote any maximal SAAF($i'_{[-]}$) walk in G such that $i(1) = N_2(i')$. Then, LP(G) includes the constraint set

$$x_{i'} \geq z^{(i')}_{i(1)} + c_{e(1)}, \tag{15a}$$

$$z^{(i')}_{i(h)} \geq z^{(i')}_{i(h+1)} + c_{e(h+1)},\ \forall h = 1, \ldots, m-2, \tag{15b}$$

$$z^{(i')}_{i(m-1)} \geq c_{e(m)}, \tag{15c}$$

where $z^{(i')}_{i(m)}$ does not appear on the right-hand side of Constraint (15c) because (via Lemma 2), maximal SAAF($i'_{[+]}$) walks must end at a leaf node. Summing Constraints (15) yields $x_{i'} \geq \sum_{h=1}^{m} c_{e(h)} \geq c_{e(m)}$. By assumption in this lemma's statement, we selected i as a leaf node with the minimum-length adjacent edge, and we must therefore have $c_{e(m)} \geq c_{e(i',3)}$. Thus, we have that $x_{i'} \geq c_{e(i',3)}$ and Constraint (12a) is satisfied by $(\bar{x}_j, \bar{y}_j),\ j \in S$.

Finally, we address the case where either $N_1(i')$ or $N_2(i')$ is a leaf node. If $N_1(i')$ is a leaf node, the proof proceeds exactly as above after assigning the value $\bar{z}^{(i')}_{N_1(i')} = 0$ and removing constraint (12c). Similarly, if $N_2(i')$ is a leaf node, apply the above proof after assigning $\bar{z}^{i'}_{N_2(i')} = 0$ and removing constraint (12e). ∎

Lemmas 6–7 lead to the following proof of correctness for LP (3).

Theorem 2 *If G contains no AAF cycles, then (a) LP (3) is feasible and (b) $y_j = \gamma_j,\ \forall j \in S$, in every optimal solution.*

Proof By iteratively selecting a leaf node i with the shortest adjacent edge, we can execute either CONTRACT-1(i) or CONTRACT-2(i), depending on whether or not i is the first neighbor of some switch node. By selecting i in this way, Lemma 4 guarantees the resulting network will preserve $\gamma_j,\ j \in S$, in each iteration, with the exception that $\gamma_{i'}$ (where i' is the neighbor of the leaf node i)

is no longer defined. Because each iteration removes a switch node, the number of iterations will be $P \equiv |S|$. Therefore, for $p = 0, 1, \ldots, P$, let G^p denote the network that results after p iterations and let LP(G^p) denote the corresponding instance of LP (3). Define $S(p)$ as the set of switch nodes in G^p such that

$$S = S(0) \supseteq S(1) \supseteq \cdots \supseteq S(P) = \emptyset. \tag{16}$$

We complete the proof in two parts: We first establish the existence of a feasible solution for which $y_j = \gamma_j$ for every switch node $j \in S$, thus establishing result (a); we then establish result (b) by proving that γ_j is a lower bound on feasible y_j, implying that $y_j = \gamma_j,\ \forall j \in S$ for any optimal solution under Objective (3a).

To prove result (a), we prove the following statement inductively: For all $p = 0, 1, \ldots, P$, there exists a feasible solution to LP(G^p) in which $y_j = \gamma_j$ and $x_j = \overline{\gamma}_j,\ \forall j \in S(p)$. The base case, $p = P$, is trivial because $S(P) = \emptyset$; thus, LP(G^P) has no variables or constraints and is therefore vacuously feasible.

We now assume the result holds for a given value of p and prove the result must also hold for $p - 1$. Let i' denote the (unique) switch node in $S(p-1) \setminus S(p)$, let i denote the leaf node that was contracted to remove node i'. Let $(\bar{x}_j, \bar{y}_j),\ j \in S(p)$, denote a fixed solution that satisfies the induction hypothesis, i.e., $\bar{y}_j = \gamma_j$ and $\bar{x}_j = \overline{\gamma}_j,\ \forall j \in S(p)$. Without loss of generality, we assume that $i \in \{N_1(i'), N_3(i')\}$ as the case $i = N_2(i')$ is symmetric to the case where $i = N_3(i')$. By applying Lemma 6 (if $i = N_1(i')$) or Lemma 7 (if $i = N_3(i')$), we may assign values to $(\bar{x}_{i'}, \bar{y}_{i'})$ such that the collection $\{\bar{x}_j, \bar{y}_j :\ j \in S(p-1)\}$ is feasible to LP(G^{p-1}). We now prove this assignment establishes case $p - 1$ in the inductive proof.

To prove $\bar{y}_{i'} = \gamma_{i'}$ in case $p - 1$ (denoted "Result I"), let $(i' =)\ i(0)$–$i(1)$–$\cdots$–$i(m)$ (with edges $e(1)$–$e(2)$–$\cdots$–$e(m)$ such that $e(1) = e(i', 1)$) denote a maximum-length SAAF($i'_{[+]}$) walk in G^{p-1}. Lemma 4 implies

$$\gamma_{i'} = \sum_{h=1}^{m} c_{e(h)}. \tag{17}$$

We have either $i(2) = N_1(i(1))$, which we denote as "(I, Case A)," or $i(2) \in \{N_2(i(1)), N_3(i(1))\}$, which we denote as "(I, Case B)". We prove that Lemmas 6–7 set $\bar{y}_{i'} = \gamma_{i'}$ in each of these cases.

To prove (I, Case A), suppose $i(2) = N_1(i(1))$. By Lemma 5, $i(1)$–$i(2)$–$\cdots$–$i(m)$ is a maximum-

length SAAF($i(1)_{[+]}$) walk. We proceed with the proof of this case by assuming that CONTRACT-2(i) was executed in iteration p, and we end the proof of this case by summarizing the simplifications that result if CONTRACT-1(i) was executed instead. Lemma 7 sets

$$\bar{y}_{i'} = c_{e(i',1)} + \bar{z}^{(i')}_{N_1(i')}, \tag{18a}$$

$$= c_{e(1)} + \bar{z}^{(i')}_{i(1)}, \tag{18b}$$

$$= c_{e(1)} + \bar{y}_{i(1)}, \tag{18c}$$

$$= c_{e(1)} + \gamma_{i(1)}, \tag{18d}$$

$$= c_{e(1)} + \sum_{h=2}^{m} c_{e(h)}, \tag{18e}$$

which equals $\gamma_{i'}$ by Equation (17) and yields the result. In the above, Equation (18a) is taken directly from Lemma 7. Equation (18b) holds because $i(1) = N_1(i')$ in order for ($i' =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$ to be a SAAF($i'_{[+]}$) walk. Equation (18c) employs the substitution $\bar{z}^{(i')}_{i(1)} = \bar{y}_{i(1)}$ from Equation (10) because i' is reachable from $i(1)$ along the SAAF($i(1)_{[-]}$) walk $i(1)$–$i(0)$. Equation (18d) invokes the induction hypothesis, and Equation (18e) utilizes the results that (Lemma 4) $\gamma_{i(1)}$ equals the longest SAAF($i(1)_{[+]}$) walk length in G^{p-1} and (Lemma 5) the length of this walk is $\sum_{h=2}^{m} c_{e(h)}$. (If CONTRACT-1(i) was executed in iteration p, then $m = 1$ and Equations (18) hold upon setting $\bar{z}^{(i')}_{N_1(i')} = 0$ as specified by Lemma 6).

In (I, Case B), Suppose $i(2) \in \{N_2(i(1)), N_3(i(1))\}$. By Lemma 5, $i(1)$–$i(1)$–$\cdots$–$i(m)$ is a maximum-length SAAF($i(1)_{[-]}$) walk. In this case, the proof is analogous to case I with the modification that $\bar{y}_{i(1)}$ is replaced by $\bar{x}_{i(1)}$ (because i' is now reachable along the SAAF($i(1)_{[+]}$) walk $i(1)$–$i(0)$) and $\gamma_{i(1)}$ is replaced by $\overline{\gamma}_{i(1)}$ (upon invoking the induction hypothesis). The analog of Equation (18e) holds upon invoking Lemmas 4–5 to guarantee that $\overline{\gamma}_{i(1)} = \sum_{h=2}^{m} c_{e(h)}$. This completes the proof of (I, Case B) and Result I.

Towards proving that $\bar{x}_{i'} = \overline{\gamma}_{i'}$ in case $p-1$ (denoted "Result II"), let ($i' =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$ (with edge representation $e(1)$–$e(2)$–$\cdots$–$e(m)$) denote a maximum-length SAAF($i'_{[-]}$) walk in G^{p-1}. We assume without loss of generality that $i(1) = N_2(i')$ (and therefore $e(1) = e(i', 2)$) with reasoning to follow. If $i(1) = N_3(i')$, the earlier assumption that $i \in \{N_1(i'), N_3(i')\}$ permits the possibility that $i = N_3(i')$—and $i(1)$ is therefore the leaf node i, i.e., $\overline{\gamma}_{i'} = c_{e(i',3)} = c_{e(i,1)}$. If $i \neq N_3(i')$, we have that $i = N_1(i')$ and we can assume $i(1) = N_2(i')$ by swapping, if necessary, the second and third neighbors in the ordered adjacency list of node i'. If $i = N_3(i')$, the following

rationale justifies assuming $i(1) = N_2(i')$ without loss of generality: By Lemma 2, any SAAF($i'_{[-]}$) walk whose first two nodes are i' and $N_2(i')$ can be extended into a maximal SAAF($i'_{[-]}$) walk whose last edge is adjacent to a leaf node; if $i(1) = N_3(i')$, our selecting i as a leaf node with the shortest-length adjacent edge (having length $c_{e(i,1)} = c_{e(i',3)} = \overline{\gamma}_{i'}$) ensures that the last edge in such a walk (and therefore the walk itself) is at least as long as $\overline{\gamma}_{i'}$ and therefore also a maximum-length SAAF($i'_{[-]}$) walk; we may therefore assume there is a maximum-length SAAF($i'_{[-]}$) walk among the set of walks whose first two nodes are i' and $N_2(i')$. Lemma 4 now implies

$$\overline{\gamma}_{i'} = \sum_{h=1}^{m} c_{e(h)}. \tag{19}$$

Again, we have either (II, Case A) $i(2) = N_1(i(1))$ or (II, Case B) $i(2) \in \{N_2(i(1)), N_3(i(1))\}$.

In (II, Case A), suppose $i(2) = N_1(i(1))$ such that Lemma 5 guarantees $i(1)$–$i(2)$–$\cdots$–$i(m)$ is a maximum-length SAAF($i(1)_{[+]}$) walk. We will first prove this case assuming CONTRACT-1(i) was executed in iteration p and then provide a summary of the simplifications that result if CONTRACT-2(i) was executed instead. Lemma 6 sets $\bar{x}_{i'} = \max\left\{c_{e(i',2)} + \bar{z}^{(i')}_{N_2(i')}, c_{e(i',3)} + \bar{z}^{(i')}_{N_3(i')}\right\}$, which yields

$$\begin{aligned}
\bar{x}_{i'} &= \max\left\{c_{e(i',2)} + \bar{z}^{(i')}_{N_2(i')}, c_{e(i',3)} + \bar{z}^{(i')}_{N_3(i')}\right\}, && \text{(20a)}\\
&= \max\left\{c_{e(1)} + \bar{y}_{i(1)}, c_{e(i',3)} + \bar{z}^{(i')}_{N_3(i')}\right\}, && \text{(20b)}\\
&= \max\left\{c_{e(1)} + \gamma_{i(1)}, c_{e(i',3)} + \bar{z}^{(i')}_{N_3(i')}\right\}, && \text{(20c)}\\
&= \max\left\{c_{e(1)} + \sum_{h=2}^{m} c_{e(h)}, c_{e(i',3)} + \bar{z}^{(i')}_{N_3(i')}\right\}, && \text{(20d)}\\
&= \max\left\{\overline{\gamma}_{i'}, c_{e(i',3)} + \bar{z}^{(i')}_{N_3(i')}\right\}, && \text{(20e)}
\end{aligned}$$

after applying the logic of Equations (18) to the first term in the maximum. We now prove that $\overline{\gamma}_{i'} \geq c_{e(i',3)} + \bar{z}^{(i')}_{N_3(i')}$ (and therefore $\bar{x}_{i'} = \overline{\gamma}_{i'}$). By Lemma 5 and the induction hypothesis, $\bar{z}^{(i')}_{N_3(i')}$ equals either $\overline{\gamma}_{N_3(i')}$ (if $N_1(N_3(i')) = i'$) or $\gamma_{N_3(i')}$ (otherwise); therefore, there exists a SAAF walk W of length $\bar{z}_{N_3(i')}$ that begins at node $N_3(i')$. One of these cases yields that W is a SAAF($N_3(i')_{[-]}$) walk and $N_3(i')$–i' is a SAAF($N_3(i')_{[+]}$) walk while the other yields that W is a SAAF($N_3(i')_{[+]}$) walk and $N_3(i')$–i' is a SAAF($N_3(i')_{[-]}$) walk; thus, W cannot include i' (as this would imply $Q(i') \cap \bar{Q}(i') \neq \emptyset$ and the existence of an AAF cycle in G^{p-1}). Therefore, we can create a SAAF($i'_{[-]}$) walk by merging $[i', N_3(i')] = e(i', 3)$ with the edges of W. This walk has length $c_{e(i',3)} + \bar{z}^{(i')}_{N_3(i')}$,

and we must therefore have $\overline{\gamma}_{i'} \geq c_{e(i',3)} + \bar{z}^{(i')}_{N_3(i')}$ because $\overline{\gamma}_{i'}$ is defined as the maximum length among SAAF($i'_{[-]}$) walks. This proves that $\bar{x}_{i'} = \overline{\gamma}_{i'}$ if CONTRACT-1(i) is executed in iteration p. If CONTRACT-2(i) is executed instead, then Lemma 7 sets $\bar{x}_{i'} = c_{e(i',2)} + \bar{z}^{(i')}_{N_2(i)}$, which equals $\overline{\gamma}_{i'}$ due to logic analogous to Equations (20a)–(20e).

In (II, Case B), suppose $i(2) \in \{N_2(i(1)), N_3(i(1))\}$. The proof of this case is identical to case (II:A) with the exception that $\bar{y}_{i(1)}$ is replaced by $\bar{x}_{i(1)}$ and $\gamma_{i(1)}$ is replaced by $\overline{\gamma}_{i(1)}$. The extension from (II, Case A) to (II, Case B) is analogous to the extension from (I, Case A) to (I, Case B).

With results I and II established, we have now completed inductive proof for case $p-1$, yielding that LP(G^p) contains a feasible solution in which $y_j = \gamma_j$ and $x_j = \overline{\gamma}_j,\ \forall j \in S(p)$, for all cases $p = 0, 1, \ldots, p$. In particular, case $p = 0$ yields that $y_j = \gamma_j$ and $x_j = \overline{\gamma}_j,\ \forall j \in S$, is feasible for LP (3), thus proving result (a).

To prove result (b), we establish that $y_j \geq \gamma_j,\ \forall j \in S$, for every feasible solution to LP (3). For any switch node $j \in S$, let ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$ (with edge representation $e(1)$–$e(2)$–$\cdots$–$e(m)$) be any maximum-length, maximal SAAF($j_{[+]}$) walk such that $\gamma_j = \sum_{h=1}^{m} c_{e(h)}$. Then LP (3) contains the constraints

$$y_j \geq z^{(j)}_{i(1)} + c_{e(1)}, \tag{21a}$$

$$z^{(j)}_{i(h)} \geq z^{(j)}_{i(h+1)} + c_{e(h+1)},\ \forall h = 1, \ldots, m-2, \tag{21b}$$

$$z^{(j)}_{i(m-1)} \geq c_{e(m)}, \tag{21c}$$

where $z^{(j)}_{i(m)}$ does not appear on the right-hand side of (21c) because Lemma 2 guarantees that $i(m)$ is a leaf node. Summing inequalities (21) yields

$$y_j \geq \sum_{h=1}^{m} c_{e(h)} = \gamma_j. \tag{22}$$

Note that the feasible solution with $y_j = \gamma_j,\ \forall j \in S$, as established in result (a), simultaneously achieves the lower bound (22) provided for each $y_j,\ j \in S$. Because all y-variables appear with a positive coefficient in Objective (3a), this solution must be optimal for (3). For the same reason, any solution with $y_j > \gamma_j$ for some $j \in S$ would be sub-optimal for LP (3), thereby proving result (b). ∎

B Appendix: Proof of Theorem 3

We first prove the necessary results that allow us to use the infeasibility of $\text{LP}_{j^\star}$ or having $y_{j^\star} \geq L$ in an optimal solution to $\text{LP}_{j^\star}$ to conclude that the answer to PREPROCESS($j^\star$) is "yes." We first note that a feasible $\text{LP}_{j^\star}$ must have an optimal solution because zero is a trivial lower bound on the objective value.

Lemma 8 *Suppose $LP_{j^\star}$ is feasible. In any optimal solution to $LP_{j^\star}$, $y_{j^\star} \geq \gamma_{j^\star}$.*

PROOF Let $e(1)$–$e(2)$–$\cdots$–$e(m)$ (with node representation $i(0)$–$i(1)$–$\cdots$–$i(m)$) denote a $\text{SAAF}({j^\star}_{[+]})$ walk of length $\gamma_{j^\star}$ in $G^{(j^\star)}$. We prove that $y_{j^\star} \geq \gamma_{j^\star}$ in any solution to $\text{LP}_{j^\star}$. We prove the result for the case where $N_1(i(h-1)) = i(h)$ for all $h \in \{1, 2, \ldots, m\}$ and $i(m)$ is a switch node and follow with a summary of how the proof changes for the other cases. (By Assumption 3, nodes $\{i(0), i(1), \ldots, i(m-1)\}$ must be switch nodes.) With this SAAF walk, we can extract from $\text{LP}_{j^\star}$ the constraints

$$y_{i(h-1)} \geq c_{e(h)} + y_{i(h)}, \ \forall h = 1, \ldots, m. \tag{23}$$

Summing these inequalities yields

$$y_{j^\star} = y_{i(0)} \geq c_{e(1)} + c_{e(2)} + \cdots + c_{e(m)} + y_{i(m)}. \tag{24}$$

Because $c_{e(1)} + c_{e(2)} + \cdots + c_{e(m)} = \gamma_i$ and $y_{i(m)} \geq 0$, this establishes $y_{j^\star} \geq \gamma_{j^\star}$ and completes the proof.

For $h = 1, \ldots, m$, if $i(h) \in \{N_2(i(h-1)), N_3(i(h-1))\}$ the proof can be modified by replacing $y_{i(h)}$ throughout with $x_{i(h)}$. If node $i(m)$ is not a switch node, the proof holds after replacing $y_{i(m)}$ with 0. ■

Lemma 9 *$LP_{j^\star}$ is infeasible if and only if there exists a closed AAF walk in $G^{(j^\star)}$.*

PROOF Proof of ($\leftarrow$): Suppose such a walk W exists. Denote its edges by $e(1)$–$e(2)$–$\cdots$–$e(m)$ and its nodes by ($\hat{j} =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$ ($= \hat{j}$). (All of these nodes must be switch nodes by Assmption 3.) By Assumption 4 all edges in W are reachable by an $\text{AAF}({j^\star}_{[+]})$ walk consisting of the union of an $\text{AAF}({j^\star}_{[+]})$ walk to some switch node in W and a subset of the edges in W; therefore $\text{LP}_{j^\star}$ contains two constraints for each of the edges in W. Following Equation (24) and

substituting $i(0) = i(m) = \hat{j}$, the constraints of $\text{LP}_{j^\star}$ imply

$$y_{\hat{j}} \geq c_{e(1)} + c_{e(2)} + \cdots + c_{e(m)} + y_{\hat{j}}, \tag{25}$$

where $y_{\hat{j}}$ (on both sides of the inequality) becomes $x_{\hat{j}}$ if $i(1) \in \{N_2(\hat{j}), N_3(\hat{j})\}$; however, $c_{e(1)} + c_{e(2)} + \cdots + c_{e(m)} \geq L$ because the network is BCL, thus proving that $\text{LP}_{j^\star}$ is infeasible.

Proof of ($\rightarrow$): We now prove that $\text{LP}_{j^\star}$ must be feasible if no such closed AAF walk exists. Towards this end, observe that each switch node $i \in S^{(j^\star)}$ yields exactly one constraint from Constraints (3b)–(3d): Define $v_{i,N_1(i)}$ as the dual variable associated with this constraint. Similarly, each switch node $i \in S^{(j^\star)}$ yields exactly one constraint for each $k \in \{2,3\}$ from Constraints (3e)–(3g): Let $v_{i,N_k(i)},\ k \in \{2,3\}$, denote the associated dual variable. For simplicity of exposition, let $v_{i,j} \equiv 0$ whenever $[i,j] \in E^{(j^\star)}$ but node i is a leaf node (i.e., $|N(i)| = 1$). By Farkas' lemma, $\text{LP}_{j^\star}$ is infeasible if and only if there is a solution to the system

$$\sum_{i \in S^{(j^\star)}} \sum_{k=1}^{3} c_{e(i,k)} v_{i,N_k(i)} > 0, \tag{26a}$$

$$-v_{N_1(i),i} + v_{i,N_2(i)} + v_{i,N_3(i)} = 0,\ \forall i \in S^{(j^\star)}, \tag{26b}$$

$$v_{i,N_1(i)} - v_{N_2(i),i} - v_{N_3(i),i} = 0,\ \forall i \in S^{(j^\star)}, \tag{26c}$$

$$v_{i,N_k(i)} \geq 0,\ \forall i \in S^{(j^\star)}, \tag{26d}$$

$$v_{N_k(i),i} \geq 0,\ \forall i \in S^{(j^\star)}. \tag{26e}$$

Constraints (26b)–(26e) comprise a "feasible circulation" polyhedron (e.g., as defined in [2]) on a directed network $\bar{G}^{(j^\star)} = (\bar{N}^{(j^\star)}, \bar{A}^{(j^\star)})$ in which each node $i \in N^{(j^\star)}$ has been split into two nodes ($i^{\langle 1 \rangle}$ and $i^{\langle 2 \rangle}$) such that Constraint (26b) defines the flow balance constraint for node $i^{\langle 1 \rangle}$ and Constraint (26c) defines the flow balance constraint for node $i^{\langle 2 \rangle}$. Let $\bar{N}^{(j^\star)} = \{i^{\langle \beta \rangle} :\ i \in N^{(j^\star)},\ \beta \in \{1,2\}\}$ denote the expanded set of nodes. Upon interpreting Constraints (26b) and (26c) as flow balance constraints for nodes $i^{\langle 1 \rangle}$ and $i^{\langle 2 \rangle}$, $i \in N^{(j^\star)}$, $v_{i,j}$ can be interpreted as the flow on a directed arc from either node $i^{\langle 1 \rangle} \in \bar{N}^{(j^\star)}$ or node $i^{\langle 2 \rangle} \in \bar{N}^{(j^\star)}$ to either node $j^{\langle 1 \rangle} \in \bar{N}^{(j^\star)}$ or node $j^{\langle 2 \rangle} \in \bar{N}^{(j^\star)}$. For $[i,j] \in E^{(j^\star)}$, and define $b(i,j)$ and $b(j,i)$ as in Equation (5). The directed arcs in the resulting network are

$$\bar{A}^{(j^\star)} = \bigcup_{[i,j] \in E^{(j^\star)}} \left\{ (i^{\langle 3-b(i,j) \rangle}, j^{\langle b(j,i) \rangle}), (j^{\langle 3-b(j,i) \rangle}, i^{\langle b(i,j) \rangle}) \right\}. \tag{27}$$

(Note: Figure 18 depicts the resulting directed network $\bar{G}^{(j^\star)}$ for the yard network in Figure 2 assuming $j^\star = 14$. Although unnecessary for the purposes of this proof, it is interesting to note that the directed walks of $\bar{G}^{(j^\star)}$ beginning at node $j^{\star\langle 2\rangle}$ correspond to SAAF($j^\star{}_{[+]}$) walks of $G^{(j^\star)}$.)

Now suppose $\text{LP}_{j^\star}$ is infeasible. Then, by Farkas' lemma, there exists a solution $\bar{v}$ to System (26). By Constraint (26a), we must have that $\bar{v} \neq 0$. By the flow decomposition theorem (see, e.g., [2, pp. 80–81]), $\bar{v}$ (and indeed any solution to Equations (26b)–(26e)) may be represented as a summation of flows around directed cycles, of which there must be at least one because $\bar{v} \neq 0$. Let $\bar{W}$ be any directed cycle in $\bar{G}^{(j^\star)}$, and denote the nodes of $\bar{W}$ as $i(0)^{\langle\beta(0)\rangle}$–$i(1)^{\langle\beta(1)\rangle}$–$\cdots$–$i(m)^{\langle\beta(m)\rangle}$ $(= i(0)^{\langle\beta(0)\rangle})$, where $i(h) \in S^{(j^\star)}$ and $\beta(h) \in \{1, 2\},\ \forall h \in \{1, \ldots, m\}$. In order for this to be a directed walk, we must have

$$\{(i(h-1)^{\langle\beta(h-1)\rangle}, i(h)^{\langle\beta(h)\rangle}), (i(h)^{\langle\beta(h)\rangle}, i(h-1)^{\langle\beta(h-1)\rangle})\} \subseteq \bar{A}^{(j^\star)},\ \forall h \in \{1, \ldots, m\}, \tag{28}$$

and therefore (due to Equation (27) for $[i, j] = [i(h-1), i(h)]$ and $[i, j] = [i(h), i(h+1)]$)

$$\beta(h) = b(i(h), i(h-1)) = 3 - b(i(h), i(h+1)),\ \forall h = 1, \ldots, m, \tag{29}$$

where $\beta(m+1) \equiv \beta(1)$ and $i(m+1) \equiv i(1)$. In addition, each $[i, j] \in E^{(j^\star)}$ yields exactly one directed arc from $i^{\langle 1\rangle}$ or $i^{\langle 2\rangle}$ in $\bar{N}^{(j^\star)}$ to either $j^{\langle 1\rangle}$ or $j^{\langle 2\rangle}$ in $\bar{N}^{(j^\star)}$; therefore, due to Assumption 1, the directed cycle $\bar{W}$ must satisfy

$$i(h-1) \neq i(h+1),\ \forall h = 1, \ldots, m. \tag{30}$$

We now demonstrate for $h = 1, \ldots, m$, the edge pair $\{[i(h-1), i(h)], [i(h), i(h+1)]\}$ is not an acute angle. Towards this end, suppose $i(h-1) \neq N_1(i(h))$. Then $b(i(h), i(h-1)) = 2$ according to Equation (5); however, Equation (29) now implies that $b(i(h), i(h+1)) = 1$, which implies under Equation (5) that $i(h+1) = N_1(i(h))$. Therefore, either $i(h-1)$ or $i(h+1)$ is the first neighbor of $i(h)$, proving that $\{[i(h-1), i(h)], [i(h), i(h+1)]\}$ is not an acute angle. The sequence of nodes $i(0)$–$i(1)$–$\cdots$–$i(m)$ $(= i(0))$ is therefore a closed AAF walk in $G^{(j^\star)}$. This completes the proof. ■

Lemma 10 *Suppose $LP_{j^\star}$ is feasible. In any optimal solution to $LP_{j^\star}$, $y_{j^\star} > \gamma_{j^\star}$ only if there exists a switch node j and an AAF cycle W with nodes $(j =)$ $i(0)$–$i(1)$–$\cdots$–$i(m)$ $(= j)$ such that (a) $\{[i(m-1), j], [j, i(1)]\}$ is an acute angle and (b) j is reachable via an AAF($j^\star{}_{[+]}$) walk that*

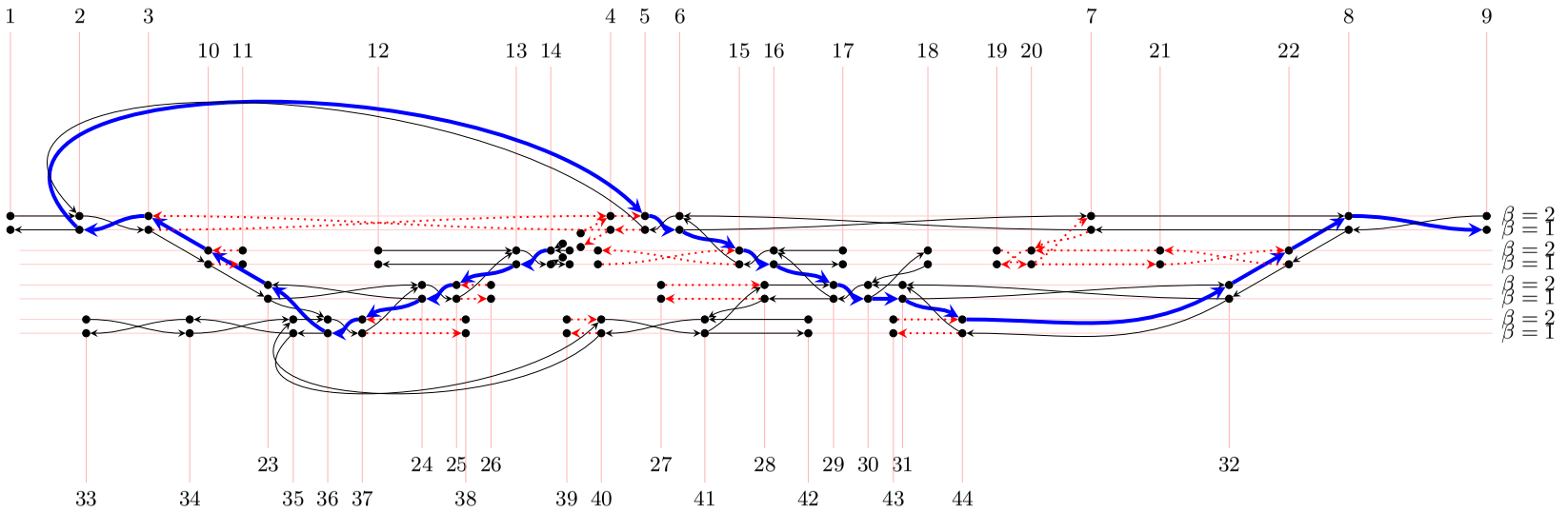

Figure 18: Directed network expansion $\bar{G}^{(j^\star)} = (\bar{N}^{(j^\star)}, \bar{A}^{(j^\star)})$ for the yard network G in Figure 2 when $j^\star = 14$. For each node $i^{\langle\beta\rangle}$, the values of i and β are labeled using the faint red lines that intersect the node. Dotted red arcs (⋯→) denote arcs that were not included in $\bar{G}^{(14)}$ because the corresponding edge was not reachable by an AAF($14_{[+]}$) walk in $G^{(14)}$. Thick blue arcs (⟶) indicate a directed walk that begins at node $14^{\langle 2\rangle}$ in $\bar{G}^{(14)}$—such a walk must always correspond to an AAF($14_{[+]}$) walk in G.

intersects W only at the switch node j (and whose last two nodes are therefore $N_1(j)$ and j).

PROOF Suppose $y_{j^\star} > \gamma_{j^\star}$ in an optimal solution to $\text{LP}_{j^\star}$. By Theorem 2, the network cannot be devoid of AAF cycles or else $y_{j^\star} = \gamma_{j^\star}$ in every optimal solution. Therefore, the network contains an AAF cycle $\hat{W}$ with nodes $i(0)$–$i(1)$–$\cdots$–$i(m)$, and all of the edges in $\hat{W}$ must be reachable via an AAF(${j^\star}_{[+]}$) walk due to Assumption 4. Let $\hat{j} = i(0) = i(m)$, which must be a switch node by Assumption 3. Due to Lemma 9, $\{[i(m-1), \hat{j}], [\hat{j}, i(1)]\}$ must be an acute angle, or else $\text{LP}_{j^\star}$ would be infeasible.

By contradiction, suppose every AAF(${j^\star}_{[+]}$) walk that ends at switch node $\hat{j}$ includes some edge in $\hat{W}$. For the remainder of this proof, the phrase "traverses edge (i', j')" conveys that i' appears immediately before j' on an AAF(${j^\star}_{[+]}, \hat{j}$) walk.

Let $h' \in \{1, \ldots, m-1\}$ denote the largest value for which there exists an AAF(${j^\star}_{[+]}, \hat{j}$) walk that traverses $(i(h'), i(h'-1))$. (If no such value of h' exists, the result is proved by contradiction because edge $[\hat{j}, i(1)]$ must be unreachable.) Similarly, let $h'' \in \{1, \ldots, m\}$ denote the smallest value for which there exists an AAF(${j^\star}_{[+]}, \hat{j}$) walk that traverses $(i(h''), i(h''+1))$. (Similarly, the value of h'' must be defined or else $[i(m-1), \hat{j}]$ is unreachable.) By definition of h', we have the following results: No AAF(${j^\star}_{[+]}, \hat{j}$) walk traverses both $(i(h'), i(h'-1))$ and $(i(h'+1), i(h'))$. And by definition of h'' we have the following: No AAF(${j^\star}_{[+]}, \hat{j}$) walk traverses both $(i(h''), i(h''+1))$ and $(i(h''-1), i(h''))$. We now prove result (b) in three cases: (I) $h' = h''$, (II) $h' < h''$, and (III) $h' > h''$.

Proof of (Case I): Let $h = h' = h''$ and consider two AAF(${j^\star}_{[+]}, \hat{j}$) walks: W', which traverses $(i(h), i(h-1))$ but not $(i(h+1), i(h))$; and W'', which traverses $(i(h), i(h+1))$ but not $(i(h-1), i(h))$. By Property 1, both of these walks must arrive to node $i(h)$ from a common neighbor, $i' \in \bar{N}^{(j^\star)}$. We have identified the three edges $\{[i', i(h)], [i(h), i(h-1)], [i(h), i(h+1)]\}$ as adjacent to node $i(h)$; however, no pair of edges from this subset can be an acute angle. The pairs $\{[i', i(h)], [i(h), i(h-1)]\}$ and $\{[i', i(h)], [i(h), i(h+1)]\}$ cannot be an acute angle because they are respectively traversed in the AAF(${j^\star}_{[+]}, \hat{j}$) walks W' and W''. The pair $\{[i(h-1), i(h)], [i(h), i(h+1)]\}$ cannot be an acute angle because $i(h)$ is an internal node in the AAF cycle W. This establishes a contradiction and proves (Case I).

Proof of (Case II): If $h' < h''$, we show the edge $[i(h'), i(h'+1)]$ cannot be reachable from node $j^\star$, which contradicts Assumption 4. By contradiction, suppose there exists an AAF(${j^\star}_{[+]}$) walk that traverses either (Case II:A) $(i(h'), i(h'+1))$ or (Case II:B) $(i(h'+1), i(h'))$. In (Case II:A), we can construct an AAF(${j^\star}_{[+]}, \hat{j}$) walk as the union of an AAF(${j^\star}_{[+]}$) walk to node $i(h')$ with the

AAF walk $i(h')$–$i(h'+1)$–$\cdots$–$i(m)$ $(=\hat{j})$. Because $h' < h''$, this contradicts our selection of h'' as the smallest value for which there exists an AAF$({j^\star}_{[+]}, \hat{j})$ walk that traverses $(i(h''), i(h''+1))$. In (Case II:B), we can construct an AAF$({j^\star}_{[+]}, \hat{j})$ walk as the union of an AAF$({j^\star}_{[+]})$ walk to node $i(h')$ with the AAF walk $i(h')$–$i(h'-1)$–$\cdots$–$i(0)$ $(=\hat{j})$. Because $h' < h''$, this contradicts our selection of h' as the largest value for which there exists an AAF$({j^\star}_{[+]}, \hat{j})$ walk that traverses $(i(h'), i(h'-1))$. This completes the proof of (Case II).

Proof of (Case III): If $h' > h''$, the set of AAF$({j^\star}_{[+]}, \hat{j})$ walks includes at least one walk that traverses $(i(h'), i(h'-1))$ and at least one walk that traverses $(i(h'-1), i(h'))$. We prove the result in two subcases: (Case III:A), there exists an AAF$({j^\star}_{[+]}, \hat{j})$ walk that traverses both $(i(h'), i(h'-1))$ and $(i(h'-1), i(h'))$; and (Case III:B), the set of AAF$({j^\star}_{[+]}, \hat{j})$ walks that traverse $(i(h'), i(h'-1))$ does not intersect the set of AAF$({j^\star}_{[+]}, \hat{j})$ walks that traverse $(i(h'-1), i(h'))$.

Proof of (Case III:A): From the AAF$({j^\star}_{[+]}, \hat{j})$ walk that traverses both $(i(h'), i(h'-1))$ and $(i(h'-1), i(h'))$, extract an AAF (sub)walk $j(0)$–$j(1)$–$\cdots$–$j(n)$ such that $j(0) = j(n) = i(h')$ and $j(1) = j(n-1) = i(h'-1)$. (The proof of the case $j(0) = j(n) = i(h'-1)$ and $j(1) = j(n-1) = i(h')$ is symmetric, so this restriction is made without loss of generality.) From $j(0)$–$j(1)$–$\cdots$–$j(n)$, extract an AAF (sub)walk $j(\ell)$–$j(\ell+1)$–$\cdots$–$j(\ell')$ (with $\ell' > \ell$) such that $j(\ell) = j(\ell')$ and all of the nodes $\{j(\ell), j(\ell+1), \ldots, j(\ell'-1)\}$ are distinct. (If no internal nodes in $j(1)$–$j(1)$–$\cdots$–$j(n-1)$ are repeated, then $\ell = 1$ and $\ell' = n-1$.) By definition of ℓ and ℓ', the AAF walk $j(\ell)$–$j(\ell+1)$–$\cdots$–$j(\ell')$ is a SAAF walk that begins and at the same node, i.e., an AAF cycle. Due to Lemma 9 and the assumption that LP$_{j^\star}$ is feasible, this AAF cycle cannot also be a closed AAF walk. Therefore, the edge pair $\{[j(\ell), j(\ell+1)], [j(\ell), j(\ell'-1)]\}$ is an acute angle; furthermore, because the AAF cycle was extracted from an AAF$({j^\star}_{[+]})$ walk, node $j(\ell)$ must be reachable via an AAF$({j^\star}_{[+]})$ walk. This case is proved after defining $j = j(\ell)$ and W as the AAF cycle $j(\ell)$–$j(\ell+1)$–$\cdots$–$j(\ell')$.

Proof of (Case III:B): Consider AAF$({j^\star}_{[+]}, \hat{j})$ walks W' and W'', such that W' traverses $(i(h'), i(h'-1))$ but not $(i(h'-1), i(h'))$ and W'' traverses $(i(h'-1), i(h'))$ but not $(i(h'), i(h'-1))$. After removing all edges after $(i(h'), i(h'-1))$ (respectively, $(i(h'-1), i(h'))$) is first traversed, let W' have nodes $(j^\star =)$ $j(0)$–$j(1)$–$\cdots$–$j(n-1)$–$j(n)$ (where $j(n-1) = i(h')$ and $j(n) = i(h'-1)$), and let W'' have nodes $(j^\star =)$ $j'(0)$–$j'(1)$–$\cdots$–$j'(n'-1)$–$j'(n')$ (where $j'(n'-1) = i(h'-1)$ and $j'(n') = i(h')$). Reversing the order of W'' and merging with W' yields the AAF walk $(j^\star =)j(0)$–$j(1)$–$\cdots$–$j(n-1)$–$j'(n'-1)$–$j'(n'-2)$–$\cdots$–$j'(0)(= j^\star)$. Analogous to the proof of (Case III:A), any AAF walk that begins and ends at the same node must contain as a subwalk an AAF cycle whose first and last edges (under Lemma 9 and the assumption that LP$_{j^\star}$ is feasible) form an acute angle.

Furthermore, because this AAF cycle is a subwalk of the AAF($j^\star{}_{[+]}$) walk ($j^\star =$)$j(0)$–$j(1)$–$\cdots$–$j(n-1)$–$j'(n'-1)$–$j'(n'-2)$–$\cdots$–$j'(0)(= j^\star)$, the first node in the AAF cycle must be reachable via an AAF($j^\star{}_{[+]}$) walk. This case is therefore proved.

To summarize, we first established the existence of an AAF cycle, $\hat{W}$, whose first node is $\hat{j}$. Although result (a) in the lemma's statement is clearly satisfied for $\hat{W}$, result (b) is not necessarily satisfied. In order to derive a contradiction, we assumed the complement of result (b) and proved (Case I)–(Case III), at least one of which describes the set of AAF($j^\star{}_{[+]}, \hat{j}$) walks. In (Case I)–(Case II), we arrived at the contradiction in order to prove that results (a) and (b) must both be satisfied for $\hat{W}$ and $\hat{j}$. In (Case III), results (a) and (b) are not satisfied for $\hat{W}$ and $\hat{j}$, but we establish the existence of another AAF cycle W and node j for which (a) and (b) are both satisfied. This completes the proof. ■

We now prove Theorem 3.

Theorem 3 *For BCL networks, the acute angle at switch node $j^\star \in S^{(j^\star)}$ is feasible if and only if (a) $LP_{j^\star}$ is infeasible or (b) $y_{j^\star} \geq L$ in an optimal solution of $LP_{j^\star}$.*

Proof

Proof of ($\rightarrow$): Suppose the acute angle at switch node $j^\star$ is feasible, i.e., $\gamma_{j^\star} \geq L$. If $\text{LP}_{j^\star}$ is feasible, Lemma 8 establishes that $y_{j^\star} \geq \gamma_{j^\star}$ ($\geq L$). This completes the proof of ($\rightarrow$).

Proof of ($\leftarrow$, Case I): Suppose $\text{LP}_{j^\star}$ is infeasible. By Lemma 9, there exists a closed AAF walk W in $G^{(j^\star)}$. By Assumption 4, all of the edges in W must be reachable via an AAF($j^\star{}_{[+]}$) walk. Therefore (reaching one such edge and then traversing all of the edges of W in sequence), there exists an AAF($j^\star{}_{[+]}$) walk ($j^\star =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$ such that W is defined by the nodes $i(n)$–$i(n+1)$–$\cdots$–$i(m)$ for some $n < m$. Within this walk, let ℓ denote the smallest value in $\{1, \ldots, m\}$ such that $i(\ell) \in \{i(0), i(1), \ldots, i(\ell-1)\}$, where $\ell \leq m$ because $i(n) = i(m)$. By our selection of ℓ, the walk ($j^\star =$) $i(0)$–$i(1)$–$\cdots$–$i(\ell)$ is a SAAF($j^\star{}_{[+]}$) walk; moreover, because the walk visits $i(\ell)$ twice, Assumption 2 implies its length must be at least L. The acute angle at switch node $j^\star \in N^{(j^\star)}$ is therefore feasible.

Proof of ($\leftarrow$, Case II): Suppose $\text{LP}_{j^\star}$ is feasible and $y_{j^\star} \geq L$ in an optimal solution of $\text{LP}_{j^\star}$. We prove the result in two subcases: ($\leftarrow$, Case II:A) $y_{j^\star} = \gamma_{j^\star}$ in an optimal solution of $\text{LP}_{j^\star}$ or ($\leftarrow$, Case II:B) $y_{j^\star} > \gamma_{j^\star}$ in an optimal solution of $\text{LP}_{j^\star}$. In ($\leftarrow$, Case II:A), the result is immediate due to Lemma 8 because ($\gamma_{j^\star} =$) $y_{j^\star} \geq L$. In ($\leftarrow$, Case II:B), Lemma 10 implies the existence of an AAF cycle W with nodes ($j =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$ ($= j$) such that (a) $\{[i(m-1), j], [j, i(1)]\}$ is

an acute angle and (b) j is reachable via an AAF($j^{\star}{}_{[+]}$) walk that intersects W only at j. If all of the nodes in the AAF($j^{\star}{}_{[+]}$) walk are unique, the union of this AAF($j^{\star}{}_{[+]}$) walk with W therefore yields a SAAF($j^{\star}{}_{[+]}$) walk in which only node j is repeated; if some node in the AAF($j^{\star}{}_{[+]}$) walk is repeated, consider instead its subwalk leading up to the first repeated node. By Assumption 2, this walk must have length at least L. Therefore, the acute angle at switch node $j^{\star} \in N^{(j^{\star})}$ is feasible.■

Remark 1 Suppose $G^{(j^{\star})} = (N^{(j^{\star})}, E^{(j^{\star})})$ has been created without removing edges that are not reachable via an AAF($j^{\star}{}_{[+]}$) walk. From $G^{(j^{\star})}$, we can apply the transformation described in the proof of Lemma 9 to obtain the directed network $\bar{G}^{(j^{\star})} = (\bar{N}^{(j^{\star})}, \bar{A}^{(j^{\star})})$. As demonstrated in the Lemma 9 proof, directed walks in $\bar{G}^{(j^{\star})}$ correspond to (possibly non-simple) AAF walks in $G^{(j^{\star})}$. Therefore, we may identify edges $[i,j] \in E^{(j^{\star})}$ that are unreachable by performing a traditional breadth first search (see, e.g., [2, p.76]) over $\bar{G}^{(j^{\star})}$, beginning at node $j^{\star\langle 2\rangle} \in \bar{N}^{(j^{\star})}$, in order to identify the subset of directed arcs $\hat{A} \subseteq \bar{A}^{(j^{\star})}$ that exist in at least one directed walk that begins at node $j^{\star\langle 2\rangle}$. We then remove each edge $[i,j]$ from $E^{(j^{\star})}$ if $\hat{A}$ contains no arcs with one endpoint in each of the node subsets $\{i^{\langle 1\rangle}, i^{\langle 2\rangle}\} \subseteq \bar{N}^{(j^{\star})}$ and $\{j^{\langle 1\rangle}, j^{\langle 2\rangle}\} \subseteq \bar{N}^{(j^{\star})}$. ■

Remark 2 The directed network $\bar{G}^{(j^{\star})} = (\bar{N}^{(j^{\star})}, \bar{A}^{(j^{\star})})$ from the proof of Lemma 9 can also be used to determine a minimum-length AAF cycle in $G^{(j^{\star})}$, which can then be used to determine whether a yard network is BCL. In $\bar{G}^{(j^{\star})}$, the directed arcs $(i^{\langle 3-b(i,j)\rangle}, j^{\langle b(j,i)\rangle})$ and $(j^{\langle 3-b(j,i)\rangle}, i^{\langle b(i,j)\rangle})$ were generated due to edge $e = [i,j] \in E^{(j^{\star})}$, where $b(i,j)$ and $b(j,i)$ are defined in Equation (5). Let each of these directed arcs have length c_e. As demonstrated in the Lemma 9 proof, AAF walks in $G^{(j^{\star})}$ correspond to directed walks in $\bar{G}^{(j^{\star})}$, and by assignment of the directed arc lengths, the lengths of these corresponding walks are identical. In $\bar{G}^{(j^{\star})}$, solve the all-pairs shortest path problem (using, e.g., the Floyd-Warshall algorithm) in order to compute the shortest path length $\delta(i^{\langle\beta\rangle}, j^{\langle\beta'\rangle})$ in $\bar{G}^{(j^{\star})}$ from each node $i^{\langle\beta\rangle} \in \bar{N}^{(j^{\star})}$ to each node $j^{\langle\beta'\rangle} \in \bar{N}^{(j^{\star})}$. If no $i^{\langle\beta\rangle}$-$j^{\langle\beta'\rangle}$ path exists, let $\delta(i^{\langle\beta\rangle}, j^{\langle\beta'\rangle})$ equal the sum of the lengths of all arcs in $\bar{A}^{(j^{\star})}$.

Now suppose there exists a minimum-length AAF cycle W with length $c(W)$ in $G^{(j^{\star})}$ such that the first and last node of W is $\hat{j} \in N^{(j^{\star})}$. Let $j(1) = N_1(\hat{j})$, $j(2) = N_2(\hat{j})$, and $j(3) = N_3(\hat{j})$ such that $b(\hat{j}, j(1)) = 1$ and $b(\hat{j}, j(2)) = b(\hat{j}, j(2)) = 2$. It is easy to show that W is also a minimum-length AAF walk among AAF walks ($\hat{j} =$) $i(0)$–$i(1)$–$\cdots$–$i(m)$ ($= \hat{j}$) such that $i(1)$ and $i(m-1)$ are distinct nodes from the set $\{j(1), j(2), j(3)\}$. Using the correspondence between AAF walks in $G^{(j^{\star})}$ and directed walks in $\bar{G}^{(j^{\star})}$, the length of W can be determined in the following three cases.

Case 1: If $i(1) = j(1)$ and $i(m-1) = j(2)$, then by Equation (27) W corresponds to a shortest directed walk in $\bar{G}^{(j^\star)}$ that traverses arc $(\hat{j}^{\langle 2\rangle}, j(1)^{\langle b(j(1),\hat{j})\rangle})$ first and arc $(j(2)^{\langle 3-b(j(2),\hat{j})\rangle}, \hat{j}^{\langle 2\rangle})$ last. In this case,

$$c(W) = c_{\hat{j},j(1)} + \delta(j(1)^{\langle b(j(1),\hat{j})\rangle}, j(2)^{\langle 3-b(j(2),\hat{j})\rangle}) + c_{j(2),\hat{j}}. \tag{31}$$

Case 2: If $i(1) = j(1)$ and $i(m-1) = j(3)$, W corresponds to a shortest directed walk in $\bar{G}^{(j^\star)}$ that traverses arc $(\hat{j}^{\langle 2\rangle}, j(1)^{\langle b(j(1),\hat{j})\rangle})$ first and arc $(j(3)^{\langle 3-b(j(3),\hat{j})\rangle}, \hat{j}^2)$ last. In this case,

$$c(W) = c_{\hat{j},j(1)} + \delta(j(1)^{\langle b(j(1),\hat{j})\rangle}, j(3)^{\langle 3-b(j(3),\hat{j})\rangle}) + c_{j(2),\hat{j}}. \tag{32}$$

Case 3: If $i(1) = j(2)$ and $i(m-1) = j(3)$, W corresponds to a shortest directed walk in $\bar{G}^{(j^\star)}$ that traverses arc $(\hat{j}^{\langle 1\rangle}, j(2)^{\langle b(j(2),\hat{j})\rangle})$ first and arc $(j(2)^{\langle 3-b(j(3),\hat{j})\rangle}, \hat{j}^{\langle 2\rangle})$ last. In this case,

$$c(W) = c_{\hat{j},j(2)} + \delta(j(2)^{\langle b(j(2),\hat{j})\rangle}, j(2)^{\langle 3-b(j(3),\hat{j})\rangle}) + c_{j(3),\hat{j}}. \tag{33}$$

Summarizing the above, the minimum AAF cycle length can be computed as

$$\begin{aligned} c(W) = \min_{\hat{j}\in N^{(j^\star)}} \min \Big\{ & c_{\hat{j},j(1)} + \delta(j(1)^{\langle b(j(1),\hat{j})\rangle}, j(3)^{\langle 3-b(j(3),\hat{j})\rangle}) + c_{j(2),\hat{j}}, \\ & c_{\hat{j},j(1)} + \delta(j(1)^{\langle b(j(1),\hat{j})\rangle}, j(3)^{\langle 3-b(j(3),\hat{j})\rangle}) + c_{j(2),\hat{j}}, \\ & c_{\hat{j},j(2)} + \delta(j(2)^{\langle b(j(2),\hat{j})\rangle}, j(2)^{\langle 3-b(j(3),\hat{j})\rangle}) + c_{j(3),\hat{j}} \Big\}. \end{aligned} \tag{34}$$

■