On the complexity of robust transshipment under consistent flow constraints

Christina Büsing*
RWTH Aachen University,
Combinatorial Optimization
Aachen, Germany
buesing@combi.rwth-aachen.de

Arie M.C.A. Koster
RWTH Aachen University,
Lehrstuhl II für Mathematik
Aachen, Germany
koster@math2.rwth-aachen.de

Sabrina Schmitz†
RWTH Aachen University,
Combinatorial Optimization
Aachen, Germany
schmitz@combi.rwth-aachen.de

ABSTRACT

In this paper, we study the complexity of robust transshipment under consistent flow constraints. We consider demand uncertainty represented by a finite set of scenarios and characterize a subset of arcs as so-called fixed arcs. In each scenario, we require a flow that satisfies the respective balance constraints. In addition, we require on each fixed arc equal flow for all scenarios. The objective is to minimize the maximum cost occurring among all scenarios.

We show that the problem is strongly $\mathcal{NP}$-hard on acyclic digraphs by a reduction from the $(3, B2)$-Sat problem. Further, we prove that the problem is weakly $\mathcal{NP}$-hard on series-parallel digraphs by a reduction from Partition. If in addition the number of scenarios is constant, we suggest a pseudo-polynomial algorithm based on dynamic programming. Finally, we present a special case solvable in polynomial time for series-parallel digraphs.

KEYWORDS

Transshipment Problem, Minimum Cost Flow, Equal Flow Problem, Robust Flows, Demand Uncertainty, Series-Parallel Digraphs

1 INTRODUCTION

In this paper, we consider the *robust transshipment problem under consistent flow constraints* (RobT≡). The problem is motivated by long-term decisions on transshipment that have to be made despite uncertainties in demand. For instance, in logistic applications the transshipment is often agreed in advance by long-term contracts with subcontractors. A solution to the RobT≡ problem facilitates cost-efficient decision-making which is robust against demand uncertainty.

The RobT≡ problem is the uncapacitated version of the robust minimum cost flow problem under consistent flow constraints (RobMCF≡), introduced in our previous work [4]. The complexity results for the RobMCF≡ problem rely on the existence of (tight) capacities. Hence, the question raises whether the problem is solvable in polynomial time if the capacity restrictions are neglected. This is the case for instance for the integral multi-commodity flow problem, which is $\mathcal{NP}$-hard but becomes polynomially solvable for uncapacitated networks.

*Supported by the Freigeist-Fellowship of the Volkswagen Stiftung and by the German research council (DFG) Research Training Group 2236 UnRAVeL.
†Supported by the Freigeist-Fellowship of the Volkswagen Stiftung.

As in the transshipment problem [8], we consider an uncapacitated network in the RobT≡ problem. To represent demand uncertainty, we consider vertex balances for a discrete number of demand scenarios. Furthermore, we characterize a subset of arcs as so-called fixed arcs. In each scenario, we require a flow that satisfies the respective balance constraints. In addition, we require on each fixed arc equal flow for all scenarios. The objective is to minimize the maximum cost that may occur among all demand scenarios.

The main contribution of this paper can be summarized as follows. We prove that finding a feasible solution to the RobT≡ problem is strongly $\mathcal{NP}$-complete on acyclic digraphs, even if only two demand scenarios are considered. On series-parallel (SP) digraphs, we show that the decision version of the RobT≡ problem is weakly $\mathcal{NP}$-complete. We identify the pseudo-polynomial time solvability in the special case of a constant number of scenarios. If all demand scenarios have the same single source and sink in SP digraphs, we propose a polynomial time algorithm.

The outline of this paper is as follows. In Section 2, we provide an overview of related work. In Section 3, we define the problem and introduce the notations of this paper. In Sections 4 and 5, we analyze the complexity of the RobT≡ problem on acyclic and SP digraphs, respectively. In Section 6, we conclude our results.

2 RELATED WORK

In the literature, there are several extensions to the maximum flow (MF) and minimum cost flow (MCF) problem that consider equal flow requirements on specified arc sets. Sahni [11] introduces the integral flow with homologous arcs problem (homIF). In addition to the set-up of the MF problem, the flow value has to be equal on specified arcs. Sahni proves the $\mathcal{NP}$-hardness of the problem by a reduction from the Non-Tautology problem. The MCF version of the homIF problem is known as the (integer) equal flow problem (EF). Using standard techniques, the complexity results can be transformed from the homIF to the EF problem [1].

Meyers and Schulz [10] discuss the transshipment version of the EF problem. For instance, they prove the strong $\mathcal{NP}$-hardness of the problem by a reduction from the Exact Cover by 3-sets problem, even in the case of a single source and sink. Furthermore, they prove the strong $\mathcal{NP}$-hardness for the special case where all sets have cardinality two, which was first investigated for the capacitated version by Ali et al. [2].

Unlike the research referenced above, we do not consider only one demand scenario in the RobT≡ problem. Like in the RobMCF≡ problem, we consider several demand scenarios. We stress that the equal flow requirements are only of importance across more than two different demand scenarios. The flow value of a specified arc has to be equal among all scenarios. In turn, the flow value of two specified arcs may differ in one scenario.

To the best of our knowledge, equal flow requirements and demand uncertainty are combined in our previous study [4] for the first time. Demand uncertainty is frequently studied in the context of (uncapacitated) network design. Three examples are as follows. Gutiérrez et al. [5] present a robustness approach to uncapacitated network design problems. Lien et al. [9] provide an efficient and robust design for transshipment networks by chain configurations. Holmberg and Hellstrand [6] concentrate on finding an optimal solution to the uncapacitated network design problem for commodities with a single source and sink by a Lagrangean heuristic within a branch-and-bound framework.

3 DEFINITION & NOTATIONS

The RobT≡ problem is the uncapacitated version of the RobMCF≡ problem. We define the problem on the basis of our previous work [4]. Let *digraph* $G = (V, A)$ with vertex set V and arc set A be given. The set of arcs A is divided into two disjoint sets A^{fix} and A^{free}, termed *fixed* and *free arcs*, respectively. If not explicitly defined, we specify the sets of vertices, arcs, fixed arcs, and free arcs of a digraph G by $V(G)$, $A(G)$, $A^{\text{fix}}(G)$, and $A^{\text{free}}(G)$, respectively. Let arc *cost* $c : A \to \mathbb{Z}_{\geq 0}$ be given. The demand uncertainty is represented by the finite set of discrete scenarios Λ. For every scenario $\lambda \in \Lambda$, vertex *balances* $b^\lambda : V \to \mathbb{Z}$ with $\sum_{v \in V} b^\lambda(v) = 0$ that define the supply and demand realizations are given, denoted by $\boldsymbol{b} = (b^1, \ldots, b^{|\Lambda|})$. A vertex with a positive or negative balance is termed *source* or *sink*, respectively. In general, the source (sink) vertices do not necessarily have to be the same in every scenario. If all scenarios have only one vertex with a positive (negative) balance and if it is the same vertex in all scenarios, we say that the problem has a *unique source (sink)*. In sum, we obtain the *network* $(G = (V, A = A^{\text{fix}} \cup A^{\text{free}}), c, \boldsymbol{b})$.

For a single scenario $\lambda \in \Lambda$, a b^λ*-flow* in digraph G is defined by a function $f^\lambda : A \to \mathbb{Z}_{\geq 0}$ that satisfies the *flow balance constraints*

$$\sum_{a=(v,w) \in A} f^\lambda(a) - \sum_{a=(w,v) \in A} f^\lambda(a) = b^\lambda(v)$$

at every vertex $v \in V$. The cost of a b^λ-flow f^λ is defined by

$$c(f^\lambda) = \sum_{a \in A} c(a) \cdot f^\lambda(a).$$

For the entire set of scenarios Λ, a *robust* $\boldsymbol{b}$*-flow* $\boldsymbol{f} = (f^1, \ldots, f^{|\Lambda|})$ is defined by a $|\Lambda|$-tuple of b^λ-flows $f^\lambda : A \to \mathbb{Z}_{\geq 0}$ that satisfy the *consistent flow constraints* $f^\lambda(a) = f^{\lambda'}(a)$ on all fixed arcs $a \in A^{\text{fix}}$ for all scenarios $\lambda, \lambda' \in \Lambda$. The cost of a robust $\boldsymbol{b}$-flow $\boldsymbol{f}$ is defined by

$$c(\boldsymbol{f}) = \max_{\lambda \in \Lambda} c(f^\lambda).$$

Finally, the RobT≡ problem is defined as follows.

Definition 3.1 (RobT≡). Given a network $(G = (V, A = A^{\text{fix}} \cup A^{\text{free}}), c, \boldsymbol{b})$, the *robust transshipment problem under consistent flow constraints* aims at computing a robust $\boldsymbol{b}$-flow $\boldsymbol{f} = (f^1, \ldots, f^{|\Lambda|})$ of minimum cost.

We note that in the case of a single scenario, i.e., $|\Lambda| = 1$, the RobT≡ problem corresponds to the transshipment problem [8]. Analogously to the RobMCF≡ problem, we stress that the integral flow property of the transshipment problem (uncapacitated MCF problem) does not hold for the RobT≡ problem. In general, the solution of the continuous relaxation of the RobT≡ problem is not integral.

4 COMPLEXITY FOR ACYCLIC DIGRAPHS

In this section, we investigate the complexity of the RobT≡ problem for networks based on acyclic digraphs. The reduction is performed from the strongly $\mathcal{NP}$-complete $(3, B2)$-Sat problem, introduced by Berman et al. [3]. The $(3, B2)$-Sat problem is a special case of the 3-Sat problem where every literal occurs exactly twice. We use the notation $[n] := \{1, \ldots, n\}$.

Theorem 4.1. *Deciding whether or not a feasible solution exists to the RobT≡ problem for networks based on acyclic digraphs is strongly $\mathcal{NP}$-complete, even if a unique source and a unique sink are given and only two scenarios are considered.*

Proof. The RobT≡ problem is contained in $\mathcal{NP}$ as we can check in polynomial time whether the flow balance and consistent flow constraints are satisfied for every scenario. Let $\{x_1, \ldots, x_n\}$ be the set of variables and $C_1, \ldots, C_m$ be the clauses of the $(3, B2)$-Sat instance $\mathcal{I}$. For a set of two scenarios $\Lambda = \{1, 2\}$, we construct a RobT≡ instance $\widetilde{\mathcal{I}} = (G, c, \boldsymbol{b})$. An example of a RobT≡ instance corresponding to a $(3, B2)$-Sat instance with four clauses and three variables is visualized in Figure 1. In general, the instance is based on a digraph $G = (V, A)$ defined as follows. The vertex set V consists of one vertex v_i per variable x_i, $i \in [n]$, one dummy vertex v_{n+1}, and one vertex u_j per clause C_j, $j \in [m]$. For every literal x_i ($\overline{x}_i$), $i \in [n]$, four auxiliary vertices w_i^ℓ ($\overline{w}_i^\ell$), $\ell \in [4]$ are included. Furthermore, set V consists of one auxiliary vertex t and vertices r_ℓ for $\ell \in [2(2n+2)]$. Arc set A contains arcs that connect two successive variable vertices v_i, v_{i+1}, $i \in [n]$ by two parallel paths p_i and $\overline{p}_i$ defined along the auxiliary vertices, i.e., $p_i = v_i w_i^1 w_i^2 w_i^3 w_i^4 v_{i+1}$ and $\overline{p}_i = v_i \overline{w}_i^1 \overline{w}_i^2 \overline{w}_i^3 \overline{w}_i^4 v_{i+1}$ for $i \in [n]$. Path p_i represents the positive literal x_i and path $\overline{p}_i$ the negative literal $\overline{x}_i$ of instance $\mathcal{I}$. As each literal occurs exactly twice, we identify two arcs of paths p_i and $\overline{p}_i$ each with the literals. More precisely, let x_i^k ($\overline{x}_i^k$) denote literal x_i ($\overline{x}_i$) which occurs the k-th time, $k \in [2]$ in the formula. *Literal arc* (w_i^{2k-1}, w_i^{2k}) ($(\overline{w}_i^{2k-1}, \overline{w}_i^{2k})$) corresponds to literal x_i^k ($\overline{x}_i^k$). Using this correspondence, we add arc (w_i^{2k}, u_j) ($(\overline{w}_i^{2k}, u_j)$) for every literal x_i^k ($\overline{x}_i^k$), $i \in [n]$, $k \in [2]$ included in clause C_j, $j \in [m]$. In the next step, we create a path $\widetilde{p}$ from vertex $v_{n+1} =: r_0$ along vertices r_ℓ, $\ell \in [2(2n+2) - 1]$ to vertex $z := r_{2(2n+2)}$. Before introducing the last arcs, we identify all literal arcs and every second arc of path $\widetilde{p}$ as the only fixed arcs in the network, i.e.,

$$A^{\text{fix}} = \left\{(w_i^\ell, w_i^{\ell+1}), (\overline{w}_i^\ell, \overline{w}_i^{\ell+1}) \mid \ell \in \{1,3\},\ i \in [n]\right\} \\ \cup \left\{(r_\ell, r_{\ell+1}) \mid \ell \in \{1, 3, \ldots, 2(2n+2) - 1\}\right\}.$$

We add arcs that connect vertex v_1 with every literal arc and every literal arc with auxiliary vertex t, i.e., (v_1, w_i^ℓ), $(v_1, \overline{w}_i^\ell)$ for $\ell \in \{1, 3\}$ and (w_i^ℓ, t), $(\overline{w}_i^\ell, t)$ for $\ell \in \{2, 4\}$. The clause vertices are connected with the first m fixed arcs of path $\widetilde{p}$, i.e., (u_j, r_{2j-1}) for all $j \in [m]$. The auxiliary vertex t is connected with the successive $2n - m$ fixed arcs of path $\widetilde{p}$ by (t, r_ℓ), $\ell \in \{2m+1, 2m+3, \ldots, 4n-1\}$. We

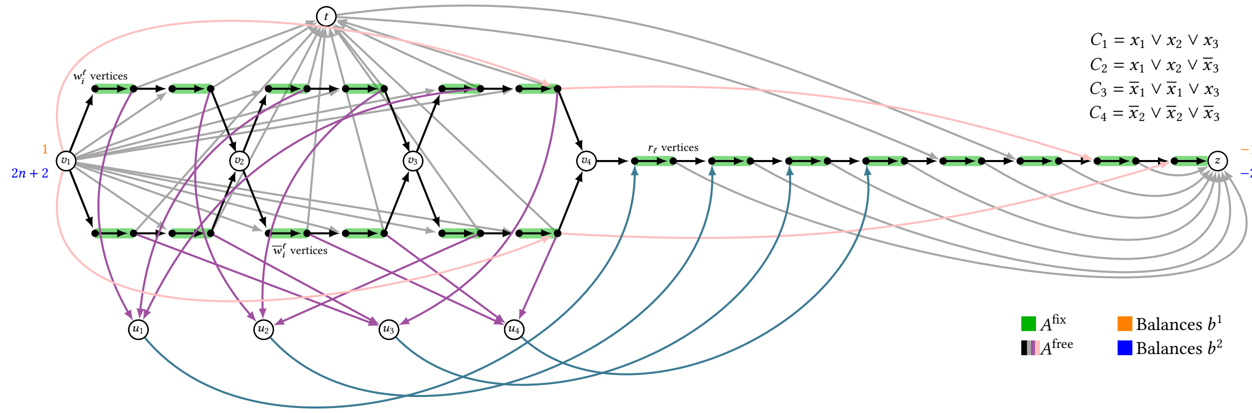

Figure 1: Construction of RobT≡ instance $\widetilde{\mathcal{I}}$.

add arcs (v_1, w_n^4), $(v_1, \overline{w}_n^4)$, (w_n^4, r_{4n+1}), and $(\overline{w}_n^4, r_{4n+3})$. Finally, we connect all $2n+2$ fixed arcs of path $\widetilde{p}$ with vertex z, i.e., (r_ℓ, z) for all $\ell \in \{2, 4, \ldots, 4n+2\}$. We set the cost $c \equiv 0$ and define the balances $\boldsymbol{b} = (b^1, b^2)$ by

$$b^1(v) = \begin{cases} 1 & \text{if } v = v_1, \\ -1 & \text{if } v = z, \\ 0 & \text{otherwise,} \end{cases} \qquad b^2(v) = \begin{cases} 2n+2 & \text{if } v = v_1, \\ -(2n+2) & \text{if } v = z, \\ 0 & \text{otherwise.} \end{cases}$$

Overall, we obtain a feasible RobT≡ instance $\widetilde{\mathcal{I}} = (G, c, \boldsymbol{b})$ that is constructed in polynomial time. Hence, it remains to show that $\mathcal{I}$ is a Yes-instance if and only if for instance $\widetilde{\mathcal{I}}$ a feasible robust $\boldsymbol{b}$-flow exists.

Let $x_1, \ldots, x_n$ be a satisfying truth assignment for instance $\mathcal{I}$. We define the first scenario flow f^1 of instance $\widetilde{\mathcal{I}}$ as follows

$$f^1(a) = \begin{cases} 1 & \text{for all } a \in A(p_i) \text{ if } x_i = \textsc{True}, \\ 1 & \text{for all } a \in A(\overline{p}_i) \text{ if } x_i = \textsc{False}, \\ 1 & \text{for all } a \in A(\widetilde{p}), \\ 0 & \text{otherwise.} \end{cases}$$

Flow f^1 uses either path p_i or $\overline{p}_i$, $i \in [n]$ to send one unit from source v_1 to vertex v_{n+1}. The unit is forwarded from vertex v_{n+1} to sink z along path $\widetilde{p}$. As $x_1, \ldots, x_n$ is a satisfying truth assignment, there exists one designated verifying literal x_i^k or $\overline{x}_i^k$, $k \in [2]$, $i \in [n]$ for each clause C_j, $j \in [m]$. Using this, we define the first part of the second scenario flow f^2 as follows

$$f^2(a) = \begin{cases} 1 & \text{for all } a \in A(q_i^k) \text{ with } q_i^k = v_1 w_i^{2k-1} w_i^{2k} u_j r_{2j-1} r_{2j} z \\ & \text{if } x_i^k \in C_j \text{ is verifying,} \\ 1 & \text{for all } a \in A(q_i^k) \text{ with } q_i^k = v_1 \overline{w}_i^{2k-1} \overline{w}_i^{2k} u_j r_{2j-1} r_{2j} z \\ & \text{if } \overline{x}_i^k \in C_j \text{ is verifying,} \\ 0 & \text{otherwise.} \end{cases}$$

Flow f^2 sends m units from the source v_1 to the clause vertices $u_1, \ldots, u_m$ along the literal arcs corresponding to the verifying literals. The m units are forwarded via the subsequent fixed arc to sink z. Further, we define the second part of the second scenario flow f^2 that sends $2n - m$ units along the remaining literal arcs to vertex t. The flow is forwarded to sink z, i.e., $f^2(a) = 1$ for all

$$\begin{cases} a \in A(p_i^k) & \text{with } p_i^k = v_1 w_i^{2k-1} w_i^{2k} t \text{ if } x_i^k = \textsc{True} \\ & \text{and } x_i^k \text{ is not chosen as a clause-verifying literal,} \\ a \in A(p_i^k) & \text{with } p_i^k = v_1 \overline{w}_i^{2k-1} \overline{w}_i^{2k} t \text{ if } x_i^k = \textsc{False} \\ & \text{and } x_i^k \text{ is not chosen as a clause-verifying literal,} \\ a \in A(p_\ell) & \text{with } p_\ell = t r_\ell r_{\ell+1} z, \ell \in \{2m+1, 2m+3, \ldots, 4n-1\}. \end{cases}$$

Finally, one unit is sent along path $v_1 w_n^4 r_{4n+1} r_{4n+2} z$ and one along path $v_1 \overline{w}_n^4 r_{4n+3} z$. We have constructed a feasible robust $\boldsymbol{b}$-flow $\boldsymbol{f} = (f^1, f^2)$.

Conversely, let $\boldsymbol{f} = (f^1, f^2)$ be a feasible robust $\boldsymbol{b}$-flow. Flows f^1 and f^2 send one and $2n+2$ units from source v_1 to sink z, respectively. By construction of the network, the only option to reach the sink requires the usage of at least two fixed arcs, namely one literal arc and one fixed arc of path $\widetilde{p}$ (except for the two paths $v_1 w_n^4 r_{4n+1} r_{4n+2} z$ and $v_1 \overline{w}_n^4 r_{4n+3} z$ which include only one fixed arc of path $\widetilde{p}$). Due to the integral flow f^1 sending one unit within the acyclic digraph, it holds $f^1(a) = f^2(a) \in \{0, 1\}$ for all fixed arcs $a \in A^{\text{fix}}$. Consequently, flows f^1 and f^2 use at least $4n+2$ fixed arcs in order to meet the demand of flow f^2. To use sufficient fixed arcs in the first scenario, flow f^1 sends the unit along either path p_i or $\overline{p}_i$ (but due to the integrality requirement and the acyclic construction never both simultaneously) for all $i \in [n]$ and subsequently along path $\widetilde{p}$. If flow f^1 sends flow along path p_i, $i \in [n]$, we set $x_i =$ True. If flow f^1 sends flow along path $\overline{p}_i$, $i \in [n]$, our choice is $x_i =$ False. To use sufficient fixed arcs in the second scenario, flow f^2 sends one unit via each clause vertex and $2n - m$ units via vertex t. Accordingly, flow f^2 sends one unit along either path $v_1 w_i^\ell w_i^{\ell+1} u_j r_{2j-1} r_{2j} z$ or $v_1 \overline{w}_i^\ell \overline{w}_i^{\ell+1} u_j r_{2j-1} r_{2j} z$, $\ell \in \{1, 3\}$, $i \in [n]$ for all $j \in [m]$ but never both simultaneously due to the consistent flow constraints. In the former case, clause C_j is verified due to the previous assignment $x_i =$ True induced by flow f^1 and the fact that $x_i \in C_j$ holds. In the latter case, clause C_j is verified due to the previous assignment $x_i =$ False induced by flow f^1 and the fact that $\overline{x}_i \in C_j$ holds. Two extra units are sent along paths $v_1 w_n^4 r_{4n+1} r_{4n+2} z$ and $v_1 \overline{w}_n^4 r_{4n+3} z$ using the last two fixed arcs on path $\widetilde{p}$. The two extra units sent are needed as otherwise, there might exist a feasible robust flow whose second scenario flow sends a unit along path $v_1 w_n^3 w_n^4 v_{n+1} r_1 r_2 z$ or $v_1 \overline{w}_n^3 \overline{w}_n^4 v_{n+1} r_1 r_2 z$ which in turn allows one unsatisfied clause. Overall, $x_1, \ldots, x_n$ is a satisfying truth assignment for instance $\mathcal{I}$. □

We note that the construction of our previous work's reduction [4] exploits (tight) capacities. For this reason, the adjusted construction containing path $\widetilde{p}$ is essential for the proof of Theorem 4.1. Otherwise, without capacities, we cannot control that one flow unit is sent via every clause vertex.

5 ROBT≡ PROBLEM ON SP DIGRAPHS

In this section, we consider the RobT≡ problem on SP digraphs. In Section 5.1, we show the weak $\mathcal{NP}$-completeness of the problem. In Section 5.2, we provide two algorithms running in polynomial time for the special case of networks with a unique source and a unique sink.

We consider SP digraphs based on the edge SP multi-graphs definition of Valdes et al. [12]. In short, SP digraphs are recursively composed serially or in parallel by SP digraphs, where a single arc itself is defined as an SP digraph. The corresponding SP tree represents its individual arcs (L-vertices) and the order of its series (S-vertices) and parallel (P-vertices) compositions by a binary decomposition computable in polynomial time [12].

5.1 Multiple Sources & Sinks Networks

Before discussing the case of networks based on SP digraphs with multiple sources and multiple sinks, we concentrate on the complexity of the RobT≡ problem in case of a unique source. We perform a reduction from the Partition problem, which is known to be weakly $\mathcal{NP}$-complete [7].

Theorem 5.1. *The decision version of the RobT≡ problem on networks based on SP digraphs with a unique source and multiple sinks is weakly $\mathcal{NP}$-complete, even if only two scenarios are considered.*

Proof. Let $\mathcal{I}$ be a Partition instance with positive integers s_i, $i \in [n]$ such that $\sum_{i=1}^{n} s_i = 2w$ holds. For a set of two scenarios $\Lambda = \{1, 2\}$, we construct a RobT≡ instance $\widetilde{\mathcal{I}} = (G, c, \boldsymbol{b})$ as visualized in Figure 2. The network is based on an SP digraph $G = (V, A)$. Vertex set V consists of two auxiliary vertices v_0, t and two vertices t_i, v_i per integer s_i, $i \in [n]$. Arc set A consists of two parallel arcs a_i^1, a_i^2 that connect vertices v_{i-1}, t_i, $i \in [n]$. Furthermore, arcs $a_i^3 = (t_i, v_i)$ and $a_i^4 = (v_{i-1}, v_i)$ for $i \in [n]$ and arc $a_{n+1} = (v_n, t)$ are included. The fixed arcs of set A are defined by arcs $a_i^2 = (v_{i-1}, t_i)$, i.e., $A^{\text{fix}} = \{a_i^2 \mid i \in [n]\}$. All other arcs are included in set A^{free}. The subgraph induced by vertices v_{i-1}, t_i, v_i represents integer s_i, $i \in [n]$. The cost c is given such that the use of arcs a_i^1 and a_i^2 cost two and one times the integer value s_i, $i \in [n]$ per flow unit, respectively. The use of arc a_{n+1} costs $2w$ per flow unit. The use of all other arcs causes zero cost. We define balances $\boldsymbol{b} = (b^1, b^2)$ by

$$b^1(v) = \begin{cases} 1 & \text{if } v = v_0, \\ -1 & \text{if } v = t, \\ 0 & \text{otherwise,} \end{cases} \qquad b^2(v) = \begin{cases} n & \text{if } v = v_0, \\ -1 & \text{if } v = t_i, \ i \in [n], \\ 0 & \text{otherwise.} \end{cases}$$

Overall, we obtain a RobT≡ instance $\widetilde{\mathcal{I}} = (G, c, \boldsymbol{b})$ that is constructed in polynomial time. Hence, it remains to show that $\mathcal{I}$ is a Yes-instance if and only if for instance $\widetilde{\mathcal{I}}$ a robust $\boldsymbol{b}$-flow exists with cost of at most $3w$.

Let S_1 and S_2 be a feasible partition for instance $\mathcal{I}$. We define the first scenario flow f^1 for instance $\widetilde{\mathcal{I}}$ by

$$f^1(a) = \begin{cases} 1 & \text{for arcs } a = a_i^4 \in A, \ i \in [n] \text{ if } s_i \in S_1, \\ 1 & \text{for arcs } a \in \{a_i^2, a_i^3\} \subseteq A, \ i \in [n] \text{ if } s_i \in S_2, \\ 1 & \text{for arc } a = a_{n+1} \in A, \\ 0 & \text{otherwise.} \end{cases}$$

The cost is

$$\begin{aligned} c(f^1) &= \sum_{a \in A^{\text{fix}}} c(a) f^1(a) + \sum_{a \in A^{\text{free}} \setminus \{a_{n+1}\}} c(a) f^1(a) + c(a_{n+1}) f^1(a_{n+1}) \\ &= w + 0 + 2w = 3w. \end{aligned}$$

According to flow f^1 and the partition, we define the second scenario flow f^2 by

$$f^2(a) = \begin{cases} 1 & \text{for arcs } a = a_i^1 \in A, \ i \in [n] \text{ if } s_i \in S_1, \\ 1 & \text{for arcs } a = a_i^2 \in A, \ i \in [n] \text{ if } s_i \in S_2, \\ n - i & \text{for arcs } a = a_i^4 \in A, \ i \in [n], \\ 0 & \text{otherwise.} \end{cases}$$

The cost is

$$c(f^2) = \sum_{a \in A^{\text{fix}}} c(a) f^2(a) + \sum_{a \in A^{\text{free}}} c(a) f^2(a) = w + 2w = 3w.$$

Consequently, we have constructed a robust $\boldsymbol{b}$-flow $\boldsymbol{f} = (f^1, f^2)$ with cost $c(\boldsymbol{f}) = 3w$.

Conversely, let $\boldsymbol{f} = (f^1, f^2)$ be a robust $\boldsymbol{b}$-flow with cost $c(\boldsymbol{f}) = \max\{c(f^1), c(f^2)\} \leq 3w$. The first scenario flow f^1 sends one unit from source v_0 to sink t. To reach sink t, arc $a_{n+1} = (v_n, t)$ with cost of $2w$ is used. If all arcs a_i^4, $i \in [n]$ of cost zero were used to reach vertex v_n, no fixed arc could be used in the second scenario due to the consistent flow constraints. This in turn means that flow f^2 would have to use all arcs a_i^1, $i \in [n]$ to send one unit from source v_0 to each of the sinks $t_1, \ldots, t_n$. However, the cost would be

$$\sum_{i=1}^{n} c(a_i^1) = \sum_{i=1}^{n} 2s_i = 4w > 3w \geq \max\{c(f^1), c(f^2)\}.$$

Thus, flow f^2 uses at least as many fixed arcs a_i^2, $i \in [n]$ as cost of w is saved. In return, flow f^1 uses as many fixed arcs a_i^2, $i \in [n]$ as cost of at most w is caused. Consequently, to reach sink t, flow f^1 uses either arc a_i^4 or the two successive arcs a_i^2, a_i^3, $i \in [n]$ of each subgraph but due to the integrality requirement and the acyclic construction never simultaneously. As a result, the sets

$$S_1 := \{s_i \mid f^1(a_i^4) = 1 \text{ for } i \in [n]\},$$
$$S_2 := \{s_i \mid f^1(a_i^2) = 1 \text{ and } f^1(a_i^3) = 1 \text{ for } i \in [n]\}$$

form a feasible partition for instance $\mathcal{I}$. □

Corollary 5.2. *The decision version of the RobT≡ problem on networks based on SP digraphs with multiple sources and multiple sinks is weakly $\mathcal{NP}$-complete, even if only two scenarios are considered.*

In the special case of a constant number of scenarios, we can solve the RobT≡ problem on networks based on SP digraphs with multiple sources and multiple sinks by the pseudo-polynomial algorithm presented in our previous work [4]. As the algorithm is based on dynamic programming, it needs arc capacities as input to limit

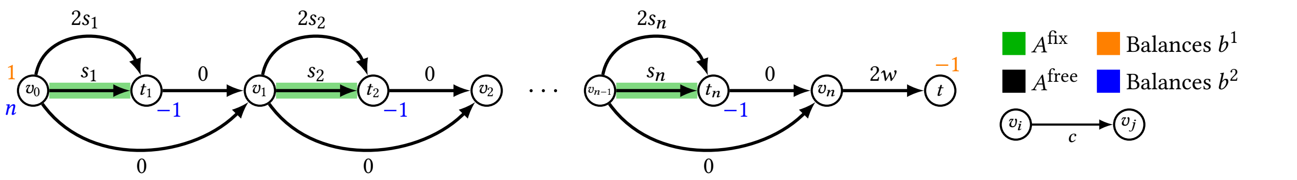

Figure 2: Construction of RobT≡ instance $\widetilde{\mathcal{I}}$.

the number of occurring labels. We can simply set the capacity for every arc to the maximum total demand among all scenarios.

5.2 Unique Source & Unique Sink Networks

For the special case of networks based on SP digraphs with a unique source and a unique sink, we can solve the RobT≡ problem by the polynomial time algorithm presented in our previous work [4]. We set the capacities required for the input to the maximum source's balance among all scenarios. The algorithm reduces to the computation of two shortest paths – one in digraph G and one in digraph $G - A^{\text{fix}}$.

In the following, we discuss an alternative polynomial algorithm which provides further insight into the RobT≡ problem on SP digraphs. Exploiting the SP structure of the digraph, we introduce two shrinking procedures, the parallel- and the series-shrinking procedure. Applying these procedures, we only need to solve the RobT≡ problem on a resulting digraph consisting of one multi-arc.

Without loss of generality, we assume a digraph G with multi-arcs of the form $a = (a^1, \ldots, a^{r(a)}, a^{r(a)+1}, \ldots, a^{k(a)})$ with $r(a) \in \mathbb{Z}_{\geq 0}$ fixed and $k(a) - r(a) \in \mathbb{Z}_{\geq 0}$ free arcs, i.e., $a^1, \ldots, a^{r(a)} \in A^{\text{fix}}(G)$ and $a^{r(a)+1}, \ldots, a^{k(a)} \in A^{\text{free}}(G)$, ordered by their costs $c(a^1) \leq \ldots \leq c(a^{r(a)})$ and $c(a^{r(a)+1}) \leq \ldots \leq c(a^{k(a)})$. We define the *parallel-shrinking procedure* as visualized in Case 1 of Figure 3. The procedure shrinks a multi-arc to a multi-arc consisting of at most one fixed and one free arc.

Require: SP digraph $G = (V, A)$, cost c, multi-arc a
Ensure: Reduced SP digraph $\widetilde{G} = (V, A \setminus \{a\} \cup \{\tilde{a}\})$
procedure Parallel-Shrinking(G, c, a)
  if $k(a) = r(a)$ **then**
    Set $\tilde{a} := (a^1)$
  else if $r(a) = 0$ or $c(a^{r(a)+1}) \leq c(a^1)$ **then**
    Set $\tilde{a} := (a^{r(a)+1})$
  else
    Set $\tilde{a} := (a^1, a^{r(a)+1})$
return $\widetilde{G} := (V, A \setminus \{a\} \cup \{\tilde{a}\})$

Applying the parallel-shrinking procedure, we obtain the following result.

Lemma 5.3 (Parallel-Shrinking). *Let $\mathcal{I} = (G, c, \boldsymbol{b})$ be a RobT≡ instance where G is an SP digraph. Further, let $\widetilde{\mathcal{I}} = (\widetilde{G}, c, \boldsymbol{b})$ be the RobT≡ instance where digraph $\widetilde{G}$ results from applying the parallel-shrinking procedure on arc $a \in A(G)$. The problem of finding an optimal robust $\boldsymbol{b}$-flow for instance $\mathcal{I}$ can be reduced to the problem of finding an optimal robust $\boldsymbol{b}$-flow for instance $\widetilde{\mathcal{I}}$.*

Proof. Let $\tilde{f}$ be an optimal robust $\boldsymbol{b}$-flow for instance $\widetilde{\mathcal{I}}$. Assume flow $\tilde{f}$ is not an optimal robust $\boldsymbol{b}$-flow for instance $\mathcal{I}$. There exists a $\boldsymbol{b}$-flow f with less cost, i.e., $c(f) < c(\tilde{f})$. Flow f uses at least one arc of set $\hat{A} := A(G) \setminus A(\widetilde{G})$. If $f^\lambda(a) > 0$ holds for a fixed or free arc $a \in \hat{A} = \{a^2, \ldots, a^{r(a)}, a^{r(a)+2}, \ldots, a^{k(a)}\}$ in one scenario $\lambda \in \Lambda$, we shift the flow to arc a^1 or arc $a^{r(a)+1}$ (provided they exist), respectively. This results in a feasible robust $\boldsymbol{b}$-flow $\hat{f}$ for instance $\widetilde{\mathcal{I}}$ with cost $c(\hat{f}) \leq c(f) < c(\tilde{f})$, which contradicts the assumption. □

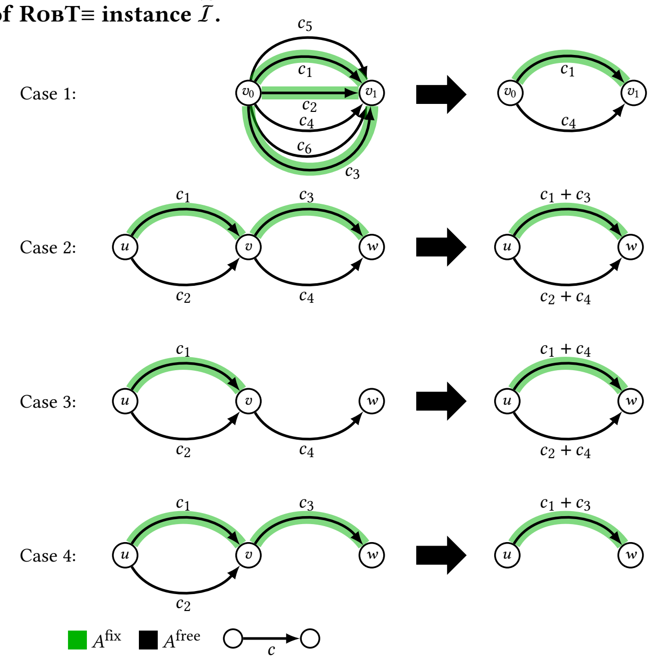

Figure 3: Parallel- and series-shrinking.

In the next step, we define the *series-shrinking procedure*. Let $a_{uv} \in A(G)$ denote a multi-arc directed from vertex u to vertex v with $u, v \in V(G)$. Further, let a_{uv} and a_{vw} be multi-arcs that consist of at most one fixed and one free arc, indicated by the labels 'fix' and 'free'. By the parallel-shrinking procedure, we assume without loss of generality that multi-arcs are of the form $a_{uv} = (a_{uv}^{\text{fix}}, a_{uv}^{\text{free}})$ with $c(a_{uv}^{\text{fix}}) < c(a_{uv}^{\text{free}})$. The series-shrinking procedure reduces a series composition of multi-arcs a_{uv} and a_{vw} associated with an S-vertex in the corresponding SP tree to a single multi-arc $\tilde{a}_{uw}$. Depending on whether multi-arc a_{vw} consists of a fixed and/or a free arc, the series-shrinking procedure is visualized in Cases 2 – 4 of Figure 3.

Require: SP digraph $G = (V, A)$, cost c, SP tree T, S-vertex $s \in V(T)$ with associated subgraph $G_s = (\{u, v, w\}, \{a_{uv}, a_{vw}\})$
Ensure: Reduced SP digraph $\widetilde{G} = (V \setminus \{v\}, A \setminus \{a_{uv}, a_{vw}\} \cup \{\tilde{a}_{uw}\})$
procedure Series-Shrinking(G, c, a_{uv}, a_{vw})
  if $a_{vw} = (a_{vw}^{\text{fix}}, a_{vw}^{\text{free}})$ **then**

Set $\tilde{a}_{uw} := (\tilde{a}^{\text{fix}}_{uw}, \tilde{a}^{\text{free}}_{uw})$
Set $c(\tilde{a}^{\text{fix}}_{uw}) = c(a^{\text{fix}}_{uv}) + c(a^{\text{fix}}_{vw})$
and $c(\tilde{a}^{\text{free}}_{uw}) = c(a^{\text{free}}_{uv}) + c(a^{\text{free}}_{vw})$
else if $a_{vw} = (a^{\text{free}}_{vw})$ **then**
Set $\tilde{a}_{uw} := (\tilde{a}^{\text{fix}}_{uw}, \tilde{a}^{\text{free}}_{uw})$
Set $c(\tilde{a}^{\text{fix}}_{uw}) = c(a^{\text{fix}}_{uv}) + c(a^{\text{free}}_{vw})$
and $c(\tilde{a}^{\text{free}}_{uw}) = c(a^{\text{free}}_{uv}) + c(a^{\text{free}}_{vw})$
else
Set $\tilde{a}_{uw} := (\tilde{a}^{\text{fix}}_{uw})$
Set $c(\tilde{a}^{\text{fix}}_{uw}) = c(a^{\text{fix}}_{uv}) + c(a^{\text{fix}}_{vw})$
return $\widetilde{G} := (V \setminus \{v\}, A \setminus \{a_{uv}, a_{vw}\} \cup \{\tilde{a}_{uw}\})$

Lemma 5.4 (Series-Shrinking). *Let $\mathcal{I} = (G, c, \boldsymbol{b})$ be a RobT≡ instance where G is an SP digraph. Let $a_{uv}, a_{vw} \in A(G)$ be multi-arcs consisting of at most one fixed and one free arc. Further, let $a_{uv}, a_{vw} \in A(G)$ be arcs whose series composition is associated with an S-vertex in the SP tree of digraph G. Let $\widetilde{\mathcal{I}} = (\widetilde{G}, c, \boldsymbol{b})$ be a RobT≡ instance where digraph $\widetilde{G}$ results from applying the series-shrinking procedure on arcs $a_{uv}, a_{vw} \in A(G)$. The problem of finding an optimal robust $\boldsymbol{b}$-flow for instance $\mathcal{I}$ can be reduced to the problem of finding an optimal robust $\boldsymbol{b}$-flow for instance $\widetilde{\mathcal{I}}$.*

Proof. Let $\tilde{a}_{uw} \in A(\widetilde{G})$ denote the shrunk arc. Let $\tilde{\boldsymbol{f}}$ be an optimal robust $\boldsymbol{b}$-flow for instance $\widetilde{\mathcal{I}}$. We define a corresponding robust $\boldsymbol{b}$-flow $\boldsymbol{f}$ for instance $\mathcal{I}$ as follows. For all arcs $a \in A(G) \setminus \{a_{uv}, a_{vw}\}$, we set $\boldsymbol{f}(a) = \tilde{\boldsymbol{f}}(a)$. For multi-arcs $a_{uv}, a_{vw} \in A(G)$, we distinguish between the following three cases, assuming $a_{uv} = (a^{\text{fix}}_{uv}, a^{\text{free}}_{uv})$ without loss of generality.

Case 1: $a_{vw} = (a^{\text{fix}}_{vw}, a^{\text{free}}_{vw})$

$$\boldsymbol{f}(a) = \begin{cases} \tilde{\boldsymbol{f}}(\tilde{a}^{\text{fix}}_{uw}) & \text{for arcs } a \in \{a^{\text{fix}}_{uv}, a^{\text{fix}}_{vw}\}, \\ \tilde{\boldsymbol{f}}(\tilde{a}^{\text{free}}_{uw}) & \text{for arcs } a \in \{a^{\text{free}}_{uv}, a^{\text{free}}_{vw}\}. \end{cases}$$

Case 2: $a_{vw} = (a^{\text{free}}_{vw})$

$$\boldsymbol{f}(a) = \begin{cases} \tilde{\boldsymbol{f}}(\tilde{a}^{\text{fix}}_{uw}) & \text{for arc } a = a^{\text{fix}}_{uv}, \\ \tilde{\boldsymbol{f}}(\tilde{a}^{\text{free}}_{uw}) & \text{for arc } a = a^{\text{free}}_{uv}, \\ \tilde{\boldsymbol{f}}(\tilde{a}^{\text{fix}}_{uw}) + \tilde{\boldsymbol{f}}(\tilde{a}^{\text{free}}_{uw}) & \text{for arc } a = a^{\text{free}}_{vw}. \end{cases}$$

Case 3: $a_{vw} = (a^{\text{fix}}_{vw})$

$$\boldsymbol{f}(a) = \begin{cases} \tilde{\boldsymbol{f}}(\tilde{a}^{\text{fix}}_{uw}) & \text{for arcs } a \in \{a^{\text{fix}}_{uv}, a^{\text{fix}}_{vw}\}, \\ 0 & \text{for arc } a = a^{\text{free}}_{uv}. \end{cases}$$

As the flow balance and consistent flow constraints are still satisfied, flow $\boldsymbol{f}$ is a feasible robust $\boldsymbol{b}$-flow for instance $\mathcal{I}$. Furthermore, flow $\boldsymbol{f}$ causes in every scenario $\lambda \in \Lambda$ the same cost as flow $\tilde{\boldsymbol{f}}$. Assume flow $\boldsymbol{f}$ is not an optimal $\boldsymbol{b}$-flow for instance $\mathcal{I}$. There exists a robust $\boldsymbol{b}$-flow $\hat{\boldsymbol{f}}$ with less cost, i.e., $c(\hat{\boldsymbol{f}}) < c(\boldsymbol{f}) = c(\tilde{\boldsymbol{f}})$. We can transform flow $\hat{\boldsymbol{f}}$ to a feasible flow $\boldsymbol{f}'$ for instance $\widetilde{\mathcal{I}}$. By definition of the shrinking procedure, flow $\boldsymbol{f}'$ causes the same cost as flow $\hat{\boldsymbol{f}}$, i.e., $c(\boldsymbol{f}') = c(\hat{\boldsymbol{f}}) < c(\boldsymbol{f}) = c(\tilde{\boldsymbol{f}})$, which contradicts to the assumption that $\tilde{\boldsymbol{f}}$ is an optimal flow for instance $\widetilde{\mathcal{I}}$. □

Using the shrinking procedures, we obtain the following result.

Theorem 5.5. *The RobT≡ problem can be solved in polynomial time on networks based on SP digraphs with a unique source and a unique sink.*

Proof. We construct SP digraph G with its unique source $o \in V(G)$ by instructions from the SP tree. If we apply the series- and the parallel-shrinking procedure after each parallel and each series composition of two subgraphs, we obtain a reduced digraph $\widetilde{G} = (\{\tilde{u}, \tilde{v}\}, \{\tilde{a}\})$. We consider the RobT≡ problem on digraph $\widetilde{G}$. For determining a robust $\boldsymbol{b}$-flow $\tilde{\boldsymbol{f}}$, we distinguish between three cases.

If $\tilde{a} = (\tilde{a}^{\text{fix}})$, there exists no feasible solution to the RobT≡ problem as the consistent flow constraints cannot be satisfied (unless the demand of all scenarios is equal). If $\tilde{a} = (\tilde{a}^{\text{free}})$, we set $\tilde{f}^{\lambda}(\tilde{a}) = b^{\lambda}(o)$ for all scenarios $\lambda \in \Lambda$. If $\tilde{a} = (\tilde{a}^{\text{fix}}, \tilde{a}^{\text{free}})$, we set

$$\tilde{f}^{\lambda}(a) = \begin{cases} \min_{\lambda \in \Lambda} b^{\lambda}(o) & \text{for arc } a = \tilde{a}^{\text{fix}}, \\ b^{\lambda}(o) - \min_{\lambda \in \Lambda} b^{\lambda}(o) & \text{for arc } a = \tilde{a}^{\text{free}}, \end{cases}$$

for scenario $\lambda \in \Lambda$. The retransformation of the reduced digraph $\widetilde{G}$ to digraph G and the analog transformation of flow $\tilde{\boldsymbol{f}}$ (cf. proof of Lemmas 5.3 and 5.4) result in a robust minimum cost $\boldsymbol{b}$-flow $\boldsymbol{f}$ for digraph G. Considering the runtime, we can construct an SP tree in $O(|A(G)|)$ time. The series- and parallel-shrinking procedures as well as the (re-)transformation are done in $O(1)$ time for all vertices of the SP tree. In total, the algorithm runs in $O(|A(G)|)$ time. □

6 CONCLUSION

In this paper, we considered the RobT≡ problem. On acyclic networks, we proved that finding a feasible solution is strongly $\mathcal{NP}$-complete, even if a unique source and sink are given and only two scenarios are considered. On networks based on SP digraphs, we proved the weak $\mathcal{NP}$-completeness, even if a unique source is given and only two scenarios are considered. For the special case of a constant number of scenarios, we showed how to solve the problem in pseudo-polynomial time. For the special case of a unique source and sink, we presented two algorithms running in polynomial time.

For the future work, we will study the complexity of the RobT≡ problem on networks based on SP digraphs with a unique source and one sink per scenario (but not a unique sink). Furthermore, we will consider the case if the number of scenarios is part of the input.

REFERENCES

[1] R K Ahuja, T L Magnanti, and J B Orlin. 1988. Network flows. (1988).
[2] A I Ali, J Kennington, and B Shetty. 1988. The equal flow problem. *European Journal of Operational Research* 36, 1 (1988), 107–115.
[3] P Berman, M Karpinski, and A D Scott. 2004. *Approximation hardness of short symmetric instances of MAX-3SAT.* Technical Report.
[4] C Büsing, A MCA Koster, and S Schmitz. 2021. Robust minimum cost flow problem under consistent flow constraints. *Annals of Operations Research* (2021), 1–32.
[5] G J Gutiérrez, P Kouvelis, and A A Kurawarwala. 1996. A robustness approach to uncapacitated network design problems. *European Journal of Operational Research* 94, 2 (1996), 362–376.
[6] K Holmberg and J Hellstrand. 1998. Solving the uncapacitated network design problem by a Lagrangean heuristic and branch-and-bound. *Operations research* 46, 2 (1998), 247–259.
[7] D S Johnson and M R Garey. 1979. *Computers and intractability: A guide to the theory of NP-completeness.* WH Freeman.
[8] B H Korte and J Vygen. 2011. *Combinatorial optimization.* Vol. 1. Springer.
[9] R W Lien, S MR Iravani, K Smilowitz, and M Tzur. 2011. An efficient and robust design for transshipment networks. *Production and Operations Management* 20, 5 (2011), 699–713.
[10] C A Meyers and A S Schulz. 2009. Integer equal flows. *Operations Research Letters* 37, 4 (2009), 245–249.
[11] S Sahni. 1974. Computationally related problems. *SIAM J. Comput.* 3, 4 (1974), 262–279.
[12] J Valdes, R E Tarjan, and E L Lawler. 1982. The recognition of series parallel digraphs. *SIAM J. Comput.* 11, 2 (1982), 298–313.