A Note on the Quickest Minimum Cost Transshipment Problem

Martin Skutella

Abstract

Klinz and Woeginger (1995) prove that the minimum cost quickest flow problem is NP-hard. On the other hand, the quickest minimum cost flow problem can be solved efficiently via a straightforward reduction to the quickest flow problem without costs. More generally, we show how the quickest minimum cost transshipment problem can be reduced to the efficiently solvable quickest transshipment problem, thus adding another mosaic tile to the rich complexity landscape of flows over time.

Keywords: flow over time, transshipment, transportation problem, complexity

1. Introduction

Network flows over time generalize the classical concept of static network flows by incorporating the temporal development of flow progressing through a network with transit times on the arcs. They have numerous applications in various areas such as traffic and transport, production, finance, evacuation, and communication; see, e.g., the classical surveys by Aronson [1] and by Powell et al. [2].

Historically, flows over time have been introduced by Ford and Fulkerson [3] (see also their classical textbook [4]). For single-source single-sink networks with capacities and transit times on the arcs, they show how to efficiently compute a maximum flow over time with given time horizon. If, instead of fixing the time horizon, we fix the flow value and ask for a flow over time with minimum time horizon, we arrive at the quickest flow problem. For this problem, Saho and Shigeno [5] present the currently fastest known algorithm with strongly polynomial running time $O(m^2n \log^2 n)$, where $m$ denotes the number of arcs and $n$ the number of nodes.

Somewhat surprisingly, the quickest transshipment problem in networks containing several sources with given supplies and several sinks with given demands...
seems to require much more advanced algorithmic techniques than the single
source single-sink case. The first algorithm due to Hoppe and Tardos \cite{6}, as well
as all efficient algorithms developed subsequently, rely on parametric submod-
ular function minimization; see Schlöter et al. \cite{7} for the latest and currently
fastest such algorithm.

Maybe even more surprisingly, Klinz and Woeginger \cite{8,9} reveal a sig
nificant complexity gap between static flows and flows over time. They show that it is
NP-hard to compute minimum cost \( s-t \)-flows over time in networks with capac-
ities, transit times, and cost coefficients on the arcs. In particular, they prove
that the \textit{minimum cost quickest flow problem}, that is, finding among all flows
over time with minimum time horizon one that has minimum cost, is NP-hard.

Reversing the order of the two objectives time and cost yields the \textit{quickest
minimum cost flow problem} which, to the best of our knowledge, is considered
for the first time in this paper, and constitutes a special case of the problem in
the main focus of this paper. Here, the primary objective is to minimize the
total cost of the flow over time, and the secondary objective is to minimize its
time horizon. It is easy to observe that, in this case, flow may be sent along
cheapest source-sink-paths only. The quickest minimum cost flow problem thus
polynomially reduces to the efficiently solvable quickest flow problem on the
subnetwork formed by all arcs that are contained in some cheapest path. This
rather straightforward observation is discussed in some more detail in Section \(2\).

In Section \(3\) we present a generalization of this polynomial reduction to the
\textit{quickest minimum cost transshipment problem}. We prove that also a quickest
minimum cost transshipment can be obtained by computing a quickest trans-
shipment in a suitably chosen subnetwork. Determining this subnetwork, how-
ever, turns out to be nontrivial as it requires an optimal dual solution to a static
transportation problem on a bipartite graph.

\textit{Preliminaries and notation}

We consider a directed graph \( D = (V,A) \) with node set \( V \) and arc set \( A \). We
use \( n \) := \( |V| \) to denote the number of nodes, and \( m := |A| \) to denote the number
of arcs. Every arc \( a \in A \) has a positive capacity \( u_a > 0 \), a non-negative transit
time \( \tau_a \geq 0 \), and a cost coefficient \( c_a \). For a path \( P \) in \( D \), we denote its
total transit time by \( \tau(P) := \sum_{a \in P} \tau_a \), and its total cost by \( c(P) := \sum_{a \in P} c_a \).
We assume throughout that arc costs are \textit{conservative}, that is, there is no negative
cost cycle in \( D \); otherwise, there might not exist a minimum cost flow over time
since cost can be decreased arbitrarily by repeatedly sending flow through a
negative cost cycle.

Among the nodes in \( D \), there is a subset of \textit{sources} \( S^+ \subset V \) and a subset of
\textit{sinks} \( S^- \subset V \setminus S^+ \) with given \textit{supplies} \( b_s > 0 \), for \( s \in S^+ \), and \textit{demands} \( -b_t > 0 \),
for \( t \in S^- \). As usual, we require that the total supply equals the total demand,
that is, \( \sum_{s \in S^+} b_s + \sum_{t \in S^-} b_t = 0 \).

For the purpose of this note, only a rudimentary understanding of flows over
time is necessary. A thorough introduction to the area can be found in the
survey \cite{10}. A flow over time \( f \) with time horizon \( \theta \) specifies for each arc \( a \in A \)
a Lebesgue-measurable function \( f_a : [0, \theta - \tau_a) \rightarrow \mathbb{R}_{\geq 0} \), where \( f_a(\theta') \) is the flow rate entering arc \( a \) at its tail at time \( \theta' \). Thus, the total amount of flow entering and traversing arc \( a \) is \( \int_{0}^{\theta - \tau_a} f_a(\theta') \, d\theta' \), and the cost incurred on arc \( a \) is \( c_a \) times this amount.

The capacity \( u_a \) of arc \( a \) provides an upper bound on its inflow rate at all points in time: \( f_a(\theta') \leq u_a \), for all \( \theta' \in [0, \theta - \tau_a) \). The transit time \( \tau_a \) is the time it takes to traverse arc \( a \). In particular, the outflow rate at the head of arc \( a \) at time \( \theta' \in [\tau_a, \theta) \) is \( f_a(\theta' - \tau_a) \). Flow conservation states that, with the exception of sources in \( S^+ \), no flow deficit may occur at nodes at any point in time. Moreover, with the exception of sinks in \( S^- \), no flow surplus must remain at nodes at time \( \theta \). Depending on the specifics of the considered flow model, it might or might not be allowed to temporarily store flow at intermediate nodes. But Fleischer and Skutella [11] prove that the possibility of temporarily storing flow at intermediate nodes does not lead to cheaper transshipments over time.

We conclude these introductory remarks on flows over time with a small, illustrative example: Sending \( b > 0 \) units of flow from node \( s \) to node \( t \) along a given \( s-t \)-path \( P \) with bottleneck capacity \( u := \min_{a \in P} u_a \) can, for example, be achieved within time horizon \( b/u + \tau(P) \) by sending flow at constant rate \( u \) into \( P \) during the time interval \([0, b/u)\) such that the last flow particle reaches \( t \) before time \( b/u + \tau(P) \).

2. Quickest minimum cost \( s-t \)-flows

As a warm-up and easy exercise, we consider the case of a single source node \( s \) and a single sink node \( t \), with supply/demand \( b_s = -b_t \). We want to find a quickest minimum cost \( s-t \)-flow in \( D \), that is, we are looking for an \( s-t \)-flow over time that sends \( b_t \) units of flow from \( s \) to \( t \) in a cheapest possible way and, among all such minimum cost solutions, has minimum time horizon \( \theta \).

If we do not restrict the time horizon \( \theta \), a cheapest way of sending \( b_s = -b_t \) units of flow from source \( s \) to sink \( t \) is to send the entire flow along a cheapest \( s-t \)-path \( P \), resulting in overall cost \( b_t \cdot c(P) \). As long as time is not an issue, capacity values and transit times of arcs on path \( P \) do not play a role. They only come into play when we ask for a minimum cost \( s-t \)-flow over time with bounded time horizon, or even a quickest minimum cost \( s-t \)-flow. In that case, it might be beneficial to use more than only one cheapest path.

Notice that an \( s-t \)-flow over time has minimum cost if and only if all flow (but a null set) is sent along cheapest \( s-t \)-paths. That is, all flow is sent through the cheapest-paths network \( D' = (V, A') \), which consists of node set \( V \) and arc subset

\[
A' := \{ a \in A \mid a \text{ lies on some cheapest } s-t \text{-path in } D \}.
\]

This immediately yields the following observation.

**Observation 1.** A quickest minimum cost \( s-t \)-flow can be obtained by finding a quickest \( s-t \)-flow in the cheapest-paths network \( D' = (V, A') \).
The currently best-known running time for computing a quickest s-t-flow is \( O(m^2n\log^2 n) \) due to Saho and Shigeno [5]. It dominates the time required to compute the cheapest-paths network \( D' = (V, A') \) defined above, which can easily be obtained as follows: For all \( v \in V \), compute the cost \( \alpha_v \) of a cheapest s-v-path as well as the cost \( \beta_v \) of a cheapest v-t-path in \( D \). Then, the cheapest-paths network’s set of arcs is \( A' = \{(v, w) \in A \mid \alpha_v + c_{(v,w)} + \beta_w = \alpha_t\} \).

3. Quickest minimum cost transshipments

When we turn to the quickest minimum cost transshipment problem in networks with several sources and sinks, it is still true that an optimum solution must only use arcs that lie on a cheapest s-t-path for some source \( s \in S^+ \) and sink \( t \in S^- \).

In contrast to the special case of s-t-flows, however, this necessary condition is no longer sufficient. Instead, as we argue below, a quickest minimum cost transshipment can be found by computing a quickest transshipment in a more carefully chosen subnetwork. Before going into details, we provide an illustrating example.

**Example 1.** We consider the network \( D = (V, A) \) depicted in Figure 1 with two source nodes \( s_1, s_2 \) of unit supply \( b_{s_1} = b_{s_2} = 1 \) and two sink nodes \( t_1, t_2 \) of unit demand \( -b_{t_1} = -b_{t_2} = 1 \). Notice that the network is constructed such that every arc \( a \in A \) lies on a cheapest \( s_i-t_j \)-path for some pair \( i, j \in \{1, 2\} \).

A quickest transshipment in \( D \) sends flow at rate 1 between times 0 and 1 through each zero transit time arc. It has time horizon \( \theta = 1 \) and total cost 1, caused by sending one unit of flow through the only ‘expensive’ arc \( s_2v \) with cost coefficient \( c_{s_2v} = 1 \). A quickest minimum cost transshipment, however, may not use the costly arc \( s_2v \). Instead, it needs to send one unit of flow from \( s_2 \) to \( t_2 \) via the direct but more time-consuming arc \( s_2t_2 \) with transit time \( \tau_{s_2t_2} = 1 \), and another unit from \( s_1 \) to \( t_1 \) via node \( v \). It thus requires time horizon \( \theta = 2 \), total cost 0, and is a quickest transshipment in a subnetwork \( D' = (V, A') \) where \( A' \) is obtained from \( A \) by deleting arc \( s_2v \) or arc \( vt_2 \) (or both).

In general, the correct choice of subnetwork \( D' = (V, A') \) such that any quickest transshipment in \( D' \) is a quickest minimum cost transshipment in the
original network $D$ not only depends on the arc costs, but also on the given supplies and demands at the sources and sinks, respectively. This can be illustrated again using the network depicted in Figure 1.

If we increase the supply of source $s_2$ to $b_{s_2} = 3/2$ and the demand of sink $t_1$ to $-b_{t_1} = 3/2$, any transshipment over time must send half a unit of flow through the costly arc $s_2v$. In this case, a quickest minimum cost transshipment still has time horizon $\theta = 2$, total cost $1/2$, and is a quickest transshipment in the subnetwork obtained by only deleting arc $vt_2$, that is, $A' = A \setminus \{vt_2\}$.

On the other hand, if instead we increase the supply of source $s_1$ to $b_{s_1} = 3/2$ and the demand of sink $t_2$ to $-b_{t_2} = 3/2$ in Figure 1, a minimum cost transshipment over time may no longer use the costly arc $s_2v$, but must send half a unit of flow through arc $vt_2$. Thus, a quickest minimum cost transshipment has time horizon $\theta = 2$ again, total cost 0, and is a quickest transshipment in the subnetwork obtained by deleting arc $s_2v$ only, i.e., $A' = A \setminus \{s_2v\}$.

For an arbitrary instance of the quickest minimum cost transshipment problem, we would like to determine which arcs of the given network $D = (V, A)$ may be used by a minimum cost transshipment over time. As a first step, we ask whether a particular source node $s \in S^+$ may serve a particular sink node $t \in S^-$ in a minimum cost transshipment over time.

To this end, we construct a static transportation problem on a bipartite network $\tilde{D} = (S^+ \cup S^-, \tilde{A})$ with $\tilde{A} := \{st \in S^+ \times S^- \mid \exists s\text{-}t\text{-path in } D\}$. An arc $st \in \tilde{A}$ thus represents the option to send flow from source $s$ to sink $t$ in a transshipment over time. Since a transshipment over time in $D$ with arbitrarily large time horizon may send an arbitrary amount of flow along an $s\text{-}t$-path, all arcs in $\tilde{A}$ get infinite capacity. Moreover, the cost $\tilde{c}_{st}$ of arc $st \in \tilde{A}$ is set to the cost of a cheapest $s\text{-}t$-path in $D$, since, as already discussed above, a minimum cost transshipment over time may send flow along cheapest paths only. Finally, supplies and demands at the nodes of the bipartite network $\tilde{D}$ match those given in the original network. In Figure 2 we illustrate the transportation problem corresponding to Example 1, where the supply of source $s_2$ and the demand of sink $t_1$ have been increased to $b_{s_2} = -b_{t_1} = 3/2$.

Observation 2. The static transportation problem on $\tilde{D}$ has a feasible solution

![Figure 2: Transportation problem corresponding to the quickest minimum cost transshipment instance depicted in Fig. 1 (with increased supply/demand $b_{s_2} = -b_{t_1} = 3/2$), together with an optimum dual solution $y^*$](image-url)
if and only if there exists a transshipment over time in $D$. A cheapest solution to the static transportation problem on $\bar{D}$ has the same cost as a minimum cost transshipment over time in $D$. Moreover, there is a minimum cost transshipment over time in $D$ that sends a positive amount of flow from source $s \in S^+$ to sink $t \in S^-$ if and only if $st \in \bar{A}$ and there is an optimal solution to the static transportation problem with positive flow value on arc $st \in \bar{A}$.

**Proof.** Given a transshipment over time in $D$, a solution to the transportation problem on $\bar{D}$ can be obtained as follows: assign to each arc $st \in \bar{A}$ the amount of flow that is being sent from $s$ to $t$ in the given transshipment over time in $D$. By definition of $\bar{c}_{st}$, the cost for sending this flow through arc $st \in \bar{A}$ is a lower bound on the respective cost incurred by the transshipment over time in $D$.

Vice versa, given a solution to the transportation problem on $\bar{D}$, one can construct an equally expensive transshipment over time in $D$ as follows: for each arc $st \in \bar{A}$, send the amount of flow carried by $st$ along a cheapest $s$-$t$-path in $D$. Since the time horizon $\theta$ may be chosen arbitrarily, the entire transport of flow through $D$ can be scheduled such that no arc capacities are violated. $\square$

In order to exploit this observation, we consider the natural linear programming formulation of the static transportation problem on $\bar{D}$. For every $st \in \bar{A}$, the variable $x_{st}$ gives the amount of flow sent through arc $st$ in $\bar{D}$:

$$\min \sum_{st \in \bar{A}} \bar{c}_{st} \cdot x_{st}$$

subject to:

$$\sum_{t:st \in \bar{A}} x_{st} = b_s \quad \text{for all } s \in S^+,$$

$$\sum_{s:st \in \bar{A}} -x_{st} = b_t \quad \text{for all } t \in S^-,$$

$$x_{st} \geq 0 \quad \text{for all } st \in \bar{A}.$$  

The corresponding dual linear program is:

$$\max \sum_{s \in S^+} b_s \cdot y_s + \sum_{t \in S^-} b_t \cdot y_t$$

subject to:

$$y_s - y_t \leq \bar{c}_{st} \quad \text{for all } st \in \bar{A}. \tag{1}$$

In view of Observation 2, we assume that there exists a transshipment over time in $D$ and thus a feasible solution to our static transportation problem. Since we assume that cost coefficients in $D$ are conservative, Observation 2 also implies the existence of minimum cost solutions to both problems. In particular, the dual linear program above has an optimum solution which we denote by $y^*$. For $s \in S^+$ and $t \in S^-$, we say that the source-sink pair $s, t$ is active, if $st \in \bar{A}$ and the corresponding dual constraint (1) is tight, i.e., $y_s^* - y_t^* = \bar{c}_{st}$. Moreover, an $s$-$t$-path $P$ in $D$ is called admissible if $s, t$ is an active pair and $P$ is a cheapest $s$-$t$-path, i.e., $c(P) = \bar{c}_{st}$.
Observation 3. A transshipment over time in $D$ has minimum cost if and only if flow is being sent from $s \in S^+$ to $t \in S^-$ along admissible $s$-$t$-paths only.

Proof. The observation is a direct consequence of Observation 2 and the complementary slackness theorem of linear programming.

Next we show that there is a sub-network $D' = (V, A')$ of $D$ that contains all admissible $s$-$t$-paths, and such that any $s$-$t$-path in $D'$ is admissible, for $s \in S^+$ and $t \in S^-$. To this end, starting from $D$, we first construct an extended network $\hat{D}$ by adding a super-source $\hat{s}$ and a super-sink $\hat{t}$ to node set $V$. The super-source $\hat{s}$ is connect to all sources $s \in S^+$ by arcs $\hat{ss}$; and all sinks $t \in S^-$ are connected to the super-sink $\hat{t}$ by arcs $\hat{tt}$. The new arcs’ cost coefficients are set to $c_{\hat{ss}} := -y^*_s$, for $s \in S^+$, and $c_{\hat{tt}} := y^*_t$, for $t \in S^-$. Notice that, by construction, $\hat{D}$ does not contain negative cost cycles since every cycle in $D$ is also contained in $D$.

**Lemma 1.** Consider the subnetwork $D' = (V, A')$ of $D$ with

$$A' := \{ a \in A \mid a \text{ lies on cheapest } \hat{s}\hat{t}\text{-path in } \hat{D}\}. $$

Then, for $s \in S^+$ and $t \in S^-$, an $s$-$t$-path in $D$ is admissible if and only if it is contained in subnetwork $D'$, that is, if and only if it uses edges in $A'$ only.

Proof. Any $s$-$t$-path $P$ in $D$ naturally corresponds to an $\hat{s}\hat{t}$-path $\hat{P}$ in the extended network $\hat{D}$, where $\hat{P}$ is obtained from $P$ by adding arcs $\hat{ss}$ and $\hat{tt}$ in the beginning and end, respectively.

We start off by showing that $P$ is admissible if and only if $c(\hat{P}) = 0$. Notice that the cost of $\hat{P}$ is equal to $c_{\hat{ss}} + c(P) + c_{\hat{tt}} = -y^*_s + c(P) + y^*_t$. Thus, $c(\hat{P}) = 0$ if and only if $c(P) = y^*_s - y^*_t$. The latter condition is equivalent to the definition of $P$ being admissible.

It remains to prove that the $\hat{s}\hat{t}$-paths of cost zero are exactly the cheapest $\hat{s}\hat{t}$-path in $\hat{D}$. Together with the result of the previous paragraph this then implies that an $s$-$t$-path $P$ is admissible if and only if $\hat{P}$ is a cheapest $\hat{s}\hat{t}$-path, which, by definition of $A'$, is equivalent to the statement of the lemma.

We thus need to show that an arbitrary $\hat{s}\hat{t}$-path $\hat{P}$ in $\hat{D}$ has non-negative cost. The first and last arcs of $\hat{P}$ are $\hat{ss}$, with $s \in S^+$, and $\hat{tt}$, with $t \in S^-$, respectively. Inbetween lies an $s$-$t$-path $P$ whose cost is at least the cost of a cheapest $s$-$t$-path $\bar{c}_{st}$. Thus,

$$c(\hat{P}) = c_{\hat{ss}} + c(P) + c_{\hat{tt}} \geq -y^*_s + \bar{c}_{st} + y^*_t \geq 0$$

where the last inequality immediately follows by dual feasibility [1]. This concludes the proof.

Summarizing, we can state the following algorithm which reduces the solution of the quickest minimum cost transshipment problem to computing a quickest transshipment on a suitably chosen subnetwork.
Algorithm 1. Quickest Minimum Cost Transshipment

1. for all $st \in S^+ \times S^-$, compute cost $\bar{c}_{st}$ of cheapest s-t-path in $D$
2. determine optimal dual solution $y^*$ to transportation problem on $\tilde{D}$
3. find arc set $A'$ via cheapest-paths subnetwork of extended network $\tilde{D}$
4. compute quickest transshipment on $D' = (V, A')$

With respect to the cheapest-paths subnetwork of $\tilde{D}$ in Step 3 we refer to our short discussion of how to compute a cheapest-paths subnetwork at the end of Section 2.

Theorem 1. Algorithm 1 correctly solves the quickest minimum cost transshipment problem. Its running time is dominated by the quickest transshipment computation in its final step.

Proof. By Lemma 1 and Observation 3 any transshipment over time in the subnetwork $D' = (V, A')$ yields a minimum cost transshipment over time in $D$, and any minimum cost transshipment over time in $D$ must only use arcs in $A'$ and thus lives in $D'$. In particular, a quickest transshipment in $D'$ yields a quickest minimum cost transshipment in $D$.

The currently fastest known quickest transshipment algorithm is due to Schlöter et al. [7] and achieves a running time of $\tilde{O}(m^2k^5 + m^3k^3 + m^3n)$, where $k := |S^+ \cup S^-|$ and the $\tilde{O}$-notation omits all poly-logarithmic terms.

4. Concluding Remark

Flows over time with costs can be seen as a bicriteria problem, with time horizon and total cost as two conflicting objectives. The efficiently solvable quickest minimum cost flow problem asks for a solution at one end of the corresponding tradeoff curve, while the NP-hard minimum cost quickest flow problem targets the other end. As Klinz and Woeginger [8, 9] point out, computing minimum cost flows over time with bounded time horizon is NP-hard. This essentially means that solutions corresponding to intermediate points on the tradeoff curve are NP-hard to find. In view of this, the fact that the minimum cost quickest flow/transshipment problem can be solved efficiently thus reveals a kind of singularity in the complexity landscape of minimum cost flows over time.

Acknowledgement

This note is dedicated to the memory of my friend and colleague Gerhard Woeginger, in thankful admiration for his outstanding scientific contributions, his fine taste of problems, his unfailingly inspiring lectures, and, last but not least, for his great sense of humor.
References

[1] J. E. Aronson, A survey of dynamic network flows, Annals of Operations Research 20 (1989) 1–66. doi:10.1007/BF02216922

[2] W. B. Powell, P. Jaillet, A. Odoni, Stochastic and dynamic networks and routing, in: M. O. Ball, T. L. Magnanti, C. L. Monma, G. L. Nemhauser (Eds.), Network Routing, volume 8 of Handbooks in Operations Research and Management Science, North–Holland, Amsterdam, The Netherlands, 1995, pp. 141–295. doi:10.1016/S0927-0507(05)80107-0

[3] L. R. Ford, D. R. Fulkerson, Constructing maximal dynamic flows from static flows, Operations Research 6 (1958) 419–433. doi:10.1287/opre.6.3.419

[4] L. R. Ford, D. R. Fulkerson, Flows in Networks, Princeton University Press, 1962. doi:10.1515/9781400875184

[5] M. Saho, M. Shigeno, Cancel-and-tighten algorithm for quickest flow problems, Networks 69 (2017) 179–188. doi:10.1002/net.21726

[6] B. Hoppe, E. Tardos, The quickest transshipment problem, Mathematics of Operations Research 25 (2000) 36–62. doi:10.1287/moor.25.1.36.15211

[7] M. Schlöter, M. Skutella, K. V. Tran, A faster algorithm for quickest transshipments via an extended discrete newton method, in: Proceedings of the 33rd ACM-SIAM Symposium on Discrete Algorithms, 2022, pp. 90–102. doi:10.1137/1.9781611977073.5

[8] B. Klinz, G. J. Woeginger, Minimum cost dynamic flows: The series-parallel case, in: E. Balas, J. Clausen (Eds.), Integer Programming and Combinatorial Optimization, 4th International IPCO Conference, volume 920 of Lecture Notes in Computer Science, Springer, 1995, pp. 329–343. doi:10.1007/3-540-59408-6\_62

[9] B. Klinz, G. J. Woeginger, Minimum-cost dynamic flows: The series-parallel case, Networks 43 (2004) 153–162. doi:10.1002/net.10112

[10] M. Skutella, An introduction to network flows over time, in: W. Cook, L. Lovász, J. Vygen (Eds.), Research Trends in Combinatorial Optimization, Springer, 2009, pp. 451–482. doi:10.1007/978-3-540-76796-1\_21

[11] L. Fleischer, M. Skutella, Quickest flows over time, SIAM J. Comput. 36 (2007) 1600–1630. doi:10.1137/S0097539703427215