An FPTAS for Quickest Multicommodity Flows with Inflow-Dependent Transit Times[⋆]

Alex Hall[1], Katharina Langkau[2], and Martin Skutella[3]

[1] Institut TIK, Gloriastrasse 35, ETH Zentrum, 8092 Zurich, Switzerland, hall@tik.ee.ethz.ch
[2] Institut für Mathematik, TU Berlin, Straße des 17. Juni 136, 10623 Berlin, Germany, langkau@math.tu-berlin.de
[3] Max-Planck Institut für Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany, skutella@mpi-sb.mpg.de

Abstract. Given a network with capacities and transit times on the arcs, the quickest flow problem asks for a 'flow over time' that satisfies given demands within minimal time. In the setting of flows over time, flow on arcs may vary over time and the transit time of an arc is the time it takes for flow to travel through this arc. In most real-world applications (such as, e.g., road traffic, communication networks, production systems, etc.), transit times are not fixed but depend on the current flow situation in the network. We consider the model where the transit time of an arc is given as a nondecreasing function of the rate of inflow into the arc. We prove that the quickest s-t-flow problem is NP-hard in this setting and give various approximation results, including an FPTAS for the quickest multicommodity flow problem with bounded cost.

1 Introduction

Flows over time have been introduced more than forty years ago by Ford and Fulkerson [6, 7]. Given a directed graph with capacities and transit times on the arcs, a source node s, a sink node t, and a time horizon T, they consider the problem of sending the maximum possible amount of flow from s to t within T time units. A flow over time specifies a flow rate for each arc at each point in time. The capacity of an arc is an upper bound on this flow rate, i.e., on the amount of flow that can be sent into the arc during each unit of time. Flow on an arc progresses at a constant speed which is determined by its transit time.

Known results for flows over time with constant transit times. Ford and Fulkerson show that the maximum s-t-flow over time problem can be solved by essentially one static min-cost flow computation in the given network, where transit times are interpreted as costs. An arbitrary path decomposition of such a static min-cost flow can be turned into

[⋆] Extended abstract; information on the full version of the paper can be obtained via the authors' WWW-pages. This work was supported in part by the joint Berlin/Zurich graduate program Combinatorics, Geometry, and Computation (CGC) financed by ETH Zurich and the German Science Foundation grant GRK 588/2 and by the EU Thematic Network APPOL II, Approximation and Online Algorithms, IST-2001-30012.

a flow over time by sending flow at the given flow rate into each path as long as there is enough time left for the flow on a path to arrive at the sink before time T. A flow featuring this structure is called 'temporally repeated'.

A problem closely related to the maximum s-t-flow over time problem is the quickest s-t-flow problem. Here, the flow value (or 'demand') is fixed and the task is to find a flow over time with minimal time horizon T. Clearly, this problem can be solved in polynomial time by incorporating the algorithm of Ford and Fulkerson into a binary search framework. Burkard, Dlaska, and Klinz [2] give a strongly polynomial algorithm for the quickest s-t-flow problem which is based on the parametric search method of Megiddo [15]. Hoppe and Tardos [10, 11] study the quickest transshipment problem which, given supplies and demands at the nodes, asks for a flow over time that zeroes all supplies and demands within minimal time. They give a polynomial time algorithm which is, however, based on a submodular function minimization routine.

The latter fact already indicates that flow over time problems are, in general, considerably harder than their static counterparts in classical network flow theory. The best evidence for this allegation is maybe provided by a surprising result of Klinz and Woeginger [12]. They show that computing a quickest s-t-flow of minimum cost in a network with cost coefficients on the arcs is already NP-hard in series-parallel networks. Moreover, it is even strongly NP-hard to find a quickest temporally repeated s-t-flow of minimum cost. Only recently, Hall, Hippler, and Skutella [8] showed that computing quickest multicommodity flows is NP-hard, even on series-parallel networks.

On the other hand, Ford and Fulkerson [6, 7] introduce the concept of time-expanded networks which allows to solve many flow over time problems in pseudopolynomial time. The node set of a time-expanded network consists of several copies of the node set of the underlying graph building a 'time layer'. The number of time layers is equal to the integral time horizon T and thus pseudopolynomial in the input size. Copies of an arc of the underlying graph join copies of its end-nodes in time layers whose distances equal the transit time of that arc. Ford and Fulkerson observe that a flow over time in the given graph corresponds to a static flow in the time-expanded network, and vice versa. Thus, many flow over time problems can be solved by static flow computations in the time-expanded network.

Fleischer and Skutella [4] come up with so-called 'condensed' time-expanded networks which are of polynomial size and can be used to compute provably good multicommodity flows over time with costs in polynomial time. In particular, they present a fully polynomial time approximation scheme (FPTAS) for the quickest multicommodity flow problem with bounded cost [4, 5]. Using completely different techniques, they also show that 2-approximate temporally repeated flows can be obtained from a static, length-bounded flow computation in the given graph [4]. The advantage of the latter solutions is that they have a very simple structure and also do not use storage of flow at intermediate nodes.

Flow-dependent transit times. So far we have considered the setting of flows over time where transit times of arcs are fixed. In many practical applications, however, the latter assumption is not realistic since transit times vary with the flow situation on an arc. We refer to [1, 16, 17] for an overview and further references. Usually, the correlation of the transit time and the flow situation on an arc is highly complex. It is a major challenge to come up with a mathematical model that, on the one hand, captures the real behavior as realistically as possible and, on the other hand, can be solved efficiently even on large networks.

Köhler and Skutella [14] consider a model where, at any moment in time, the actual speed of flow on an arc depends on the current amount of flow on the arc. Under this assumption, they give a 2-approximation algorithm for the quickest s-t-flow problem and show that no polynomial time approximation scheme (PTAS) exists, unless P=NP. A simpler model is studied by Carey and Subrahmanian [3]. They assume that the transit time on an arc only depends on the current rate of inflow into the arc and propose a time-expanded network whose arcs somehow to reflect this behavior. Köhler, Langkau, and Skutella [13] give a 2-approximation algorithm for the quickest s-t-flow problem in the setting of inflow-dependent transit times. The algorithm uses the algorithm of Ford and Fulkerson [6, 7] on a so-called 'bow graph' with fixed transit times on the arcs. In the bow graph, every arc of the original graph is replaced by a bunch of arcs corresponding to different transit times. The quickest flow problem in the bow graph is a relaxation of the quickest flow problem with inflow-dependent transit times.

Contribution of this paper. While, for the special case of constant transit times, quickest s-t-flows can be computed in polynomial time [2, 6, 7], we show in Section 6 that the problem becomes NP-hard if we allow inflow-dependent transit times. In Section 4, we generalize the 2-approximation result given in [13] to the setting with costs and multiple commodities. Our approach is based on a new and stronger relaxation of the quickest flow problem, which we introduce in Section 3. This relaxation is defined in a bow graph similar to the one introduced in [13], but it uses additional 'coupling constraints' between flow values on different copies of one arc in the original graph. In particular, this relaxation can no longer be solved by standard network flow algorithms but requires general linear programming techniques. Nevertheless, as shown in Section 4, the approximation technique based on length-bounded static flows presented in [4] can be generalized to yield provably good solutions to our bow graph relaxation. Moreover, we prove that such a solution to the relaxation can be turned into a feasible multicommodity flow over time with inflow-dependent transit times and bounded cost.

The main result of this paper is a fully polynomial time approximation scheme for the quickest multicommodity flow problem with bounded cost and inflow-dependent transit times (see Section 5). It again uses the new bow graph relaxation introduced in Section 3 and generalizes the approach based on condensed time-expanded networks from [5]. Interestingly, the time-expanded version of our bow graph relaxation essentially coincides with the modified time-expanded graph considered by Carey and Subrahmanian [3].

Due to space limitations, we omit most proofs in this extended abstract.

2 Preliminaries

We are considering network flow problems in a directed graph $G = (V, E)$ with $n := |V|$ nodes and $m := |E|$ arcs. Each arc $e \in E$ has associated with it a positive capacity u_e and a nonnegative, nondecreasing transit time function $\tau_e : [0, u_e] \rightarrow \mathbb{R}^+$. There is a set of commodities $K = \{1, \ldots, k\}$; every commodity $i \in K$ is defined by a source-sink pair[4] $(s_i, t_i) \in V \times V$. The objective is to send a prespecified amount of flow $d_i > 0$, called the demand, from s_i to t_i. Finally, each arc e has associated cost coefficients $c_{e,i}$, for $i \in K$, where $c_{e,i}$ is interpreted as the cost (per flow unit) for sending flow of commodity i through the arc. For an arc $e = (v, w) \in E$, we use the notation head$(e) := w$ and tail$(e) := v$.

Flows over time with constant transit times. A *(multicommodity) flow over time f* in G with time horizon T is given by Lebesgue-measurable functions $f_{e,i} : [0, T) \to \mathbb{R}^+$, where $f_{e,i}(\theta)$ is the rate of flow (per time unit) of commodity i entering arc e at time θ. In order to simplify notation, we sometimes use $f_{e,i}(\theta)$ for $\theta \notin [0, T)$, implicitly assuming that $f_{e,i}(\theta) = 0$ in this case. The capacity u_e is an upper bound on the rate of flow entering arc e at any moment of time, i.e., $f_e(\theta) \leq u_e$ for all $\theta \in [0, T)$ and $e \in E$. Here, $f_e(\theta) := \sum_{i \in K} f_{e,i}(\theta)$ is the total rate at which flow is entering arc e at time θ.

In the original setting of flows over time, the transit time function τ_e of arc e is assumed to be constant. Then, the flow $f_{e,i}(\theta)$ of commodity i entering arc e at time θ arrives at head(e) at time $\theta + \tau_e$. All arcs must be empty from time T on, i.e., $f_{e,i}(\theta) = 0$ for $\theta \geq T - \tau_e$. To generalize the notion of *flow conservation*, we define $D^-_{v,i}(\xi) := \sum_{e \in \delta^-(v)} \int_{\tau_e}^{\xi} f_{e,i}(\theta - \tau_e)\, d\theta$ to be the total inflow of commodity $i \in K$ into node v until time $\xi \in [0, T]$. Similarly, $D^+_{v,i}(\xi) := \sum_{e \in \delta^+(v)} \int_0^{\xi} f_{e,i}(\theta)\, d\theta$ is the corresponding outflow. We consider the model with storage of flow at intermediate nodes. That is, flow entering a node can be held back for some time before it is sent onward. To rule out deficit at any node, we require $D^-_{v,i}(\xi) - D^+_{v,i}(\xi) \geq 0$, for all $\xi \in [0, T), i \in K$, and $v \in V \backslash \{s_i\}$. Moreover, flow must not remain in any node other than the sinks at time T. Therefore, we require that equality holds for every $i \in K$, $v \in V \backslash \{s_i, t_i\}$, at time $\xi = T$. The flow over time f satisfies the multicommodity demands if $D^-_{t_i,i}(T) - D^+_{t_i,i}(T) = d_i$, for any commodity $i \in K$. The cost of a flow over time f is defined as $c(f) := \sum_{e \in E} \sum_{i \in K} c_{e,i} \int_0^T f_{e,i}(\theta) d\theta$.

Time-expanded graphs. Many flow over time problems can be solved by static flow algorithms in time-expanded graphs [6, 7]. Given a graph $G = (V, E)$ with integral transit times on the arcs and an integral time horizon T, the *T-time-expanded graph* of G, denoted G^T, is obtained by creating T copies of V, labeled V_0 through V_{T-1}, with the θ^{th} copy of node v denoted $v(\theta)$, $\theta = 0, \ldots, T - 1$. For every arc $e = (v, w) \in E$ and $\theta = 0, \ldots, T - 1 - \tau_e$, there is an arc $e(\theta)$ from $v(\theta)$ to $w(\theta + \tau_e)$ with the same capacity and costs as arc e. In addition, there is an infinite capacity *holdover arc* from $v(\theta)$ to $v(\theta + 1)$, for all $v \in V$ and $\theta = 0, \ldots, T - 2$, which models the possibility to hold flow at node v during the time interval $[\theta, \theta + 1)$.

Any static flow in this time-expanded network corresponds to a flow over time of equal cost: interpret the flow on arc $e(\theta)$ as the flow through arc $e = (v, w)$ that starts at node v in the time interval $[\theta, \theta + 1)$. Similarly, any flow over time completing by time T corresponds to a static flow in G^T of the same value and cost obtained by mapping the total flow starting on e in time interval $[\theta, \theta + 1)$ to flow on arc $e(\theta)$. Thus, we may

[4] To simplify notation, we restrict to the case of only one source and one sink for each commodity. However, our results can be directly generalized to the case of several sources and sinks with given supplies and demands for each commodity.

solve a flow over time problem by solving the corresponding static flow problem in the time-expanded network.

One drawback of this approach is that the size of G^T depends linearly on T, so that if T is not bounded by a polynomial in the input size, this is not a polynomial-time method. However, the following useful observation can be found in [4]: If all transit times are multiples of some large number $\Delta > 0$, then instead of using the T-time-expanded graph, we may rescale time and use a *Δ-condensed time-expanded graph* that contains only $\lceil T/\Delta \rceil$ copies of V. Since in this setting every arc corresponds to a time interval of length Δ, capacities are multiplied by Δ. For more details we refer to [4].

Flows with inflow-dependent transit times. In the original setting of flows over time discussed above, it is assumed that transit times are fixed throughout, so that flow on arc e progresses at a uniform speed. In the following, we will consider the more general model of *inflow-dependent* transit times. Here, the transit time of an arc may vary with the current amount of flow using this arc. Each arc e has an associated non-negative transit time function τ_e, which determines the time it takes for flow to traverse arc e. Flow of commodity i entering arc e at time θ at rate $f_{e,i}(\theta)$ arrives at head(e) at time $\theta + \tau_e(f_e(\theta))$. We will later need the following simple observation which follows from the fact that flow can be stored at intermediate nodes.

Observation 1. *For every arc $e \in E$, let $\tau_e : [0, u_e] \to \mathbb{R}^+$ and $\tau'_e : [0, u_e] \to \mathbb{R}^+$ be transit time functions on arc e such that $\tau'_e(x) \le \tau_e(x)$ for all $x \in [0, u_e]$. Then, a flow over time with inflow-dependent transit times $(\tau_e)_{e\in E}$ and time horizon T also yields a flow over time with inflow-dependent transit times $(\tau'_e)_{e\in E}$ and time horizon T.*

3 The Bow Graph

In this section, we will define a so-called bow graph that is very similar to the one defined in [13]. Let us for the moment assume that all transit time functions are piecewise constant, non-decreasing, and left-continuous. This transit time function of arc e is denoted by τ^s_e. It is given by breakpoints $0 = x_0 < x_1 < \cdots < x_\ell$ and corresponding transit times $\tau_1 < \cdots < \tau_\ell$. Flow entering arc e at rate $x \in (x_{i-1}, x_i]$ needs τ_i time to traverse arc e. Later we will use the fact that general transit time functions can be approximated by such step functions within arbitrary precision.

The bow graph, denoted $G^B = (V^B, E^B)$, is defined on the same node set as G, i.e., $V^B := V$, and is obtained by creating several copies of an arc, one for every possible transit time on this arc. Thus, arc e is replaced by ℓ parallel *bow arcs* $a_1, \dots, a_\ell$. The transit time of bow arc a_i is τ_i and its capacity is x_i, $i = 1, \dots, \ell$. We will denote the set of bow arcs corresponding to arc $e \in E$ by E^B_e, and refer to E^B_e as the *expansion* of arc e. The cost coefficients of every arc $a \in E^B_e$ are identical to those of e, i.e., $c_{a,i} := c_{e,i}$, for $i \in K$.

3.1 A Relaxation of Inflow-Dependent Transit Times

We will now discuss the relationship between flows over time with inflow-dependent transit times in G and flows over time in the bow graph G^B. Any flow over time f in G

with inflow-dependent transit times $(\tau_e^s)_{e\in E}$ and time horizon T can be interpreted as a flow over time f^B in G^B (with constant transit times) with the same time horizon T: If flow is entering arc $e \in E$ at time θ with flow rate $f_e(\theta)$, then, in the bow graph, this flow is sent onto the bow arc $a \in E_e^B$ representing the transit time $\tau_e^s(f_e(\theta))$.

Unfortunately, an arbitrary flow over time f^B in G^B does not correspond to a flow over time f with inflow-dependent transit times $(\tau_e^s)_{e\in E}$ in G. In addition, we have to require the following property: For every original arc $e \in E$ and at every point in time θ, the flow f^B sends flow into at most one bow arc $a \in E_e^B$. A flow over time in G^B fulfilling this property is called *inflow-preserving*.

Observation 2. *Every inflow-preserving flow over time f^B in G^B with time horizon T corresponds to a flow over time f in G with inflow-dependent transit times $(\tau_e^s)_{e\in E}$ and time horizon T, and vice versa.*

Notice that the set of inflow-preserving flows over time is not convex. In particular, it is difficult to compute inflow-preserving flows directly. Therefore, we also consider a relaxed notion which can be interpreted as a convexification of inflow-preserving flows: For any arc $a \in E^B$, let $\lambda_a(\theta) := f_a^B(\theta)/u_a$ denote the *per capacity inflow rate* into arc a at time θ. Then, a flow over time f^B in G^B with time horizon T is called *weakly inflow-preserving* if $\sum_{a\in E_e^B} \lambda_a(\theta) \le 1$ for all $e \in E$ and $\theta \in [0,T)$. Since every inflow-preserving flow over time is also weakly inflow-preserving, it follows from Observations 1 and 2 that weakly inflow-preserving flows over time in G^B constitute a relaxation of flows over time with inflow-dependent transit times in G:

Observation 3. *For every arc $e \in E$, let $\tau_e^s : [0,u_e] \to \mathbb{R}^+$ and $\tau_e : [0,u_e] \to \mathbb{R}^+$ be transit time functions on arc e such that τ_e^s is a step function with $\tau_e^s(x) \le \tau_e(x)$ for all $x \in [0,u_e]$. Then, every flow over time with inflow-dependent transit times $(\tau_e)_{e\in E}$ and time horizon T in G yields a (weakly) inflow-preserving flow over time with time horizon T in G^B.*

The basic idea of the approximation algorithms presented in this paper is to compute weakly inflow-preserving flows over time in an appropriate bow graph and turn these into flows over time in G with inflow-dependent transit times. The following lemma and its corollary make this approach work. Consider the expansion of a single arc $e \in E$ to bow arcs $E_e^B = \{a_1,\dots,a_\ell\}$.

Lemma 1. *Let f^B be a weakly inflow-preserving flow over time with time horizon T in E_e^B and $\delta > 0$. Then, f^B can be turned into an inflow-preserving flow over time $\hat{f}^B$ in E_e^B such that every (infinitesimal) unit of flow in $\hat{f}^B$ reaches head(e) at most δ time units later than it does in f^B.*

Proof. For every bow arc a_i, $i = 1,\dots,\ell$, we set up a buffer b_i in tail(e) for temporary storage of flow. The buffer b_i is collecting all flow in f^B which is about to be shipped through bow arc a_i. It can output this flow in a first-in-first-out manner, i.e., flow units must enter and leave the buffer in the same order. Buffer b_i has only two output modes. Either it is *closed*, then no flow is leaving the buffer, or it is *open* and flow is leaving the buffer at constant rate u_{a_i}, immediately entering arc a_i. In our modified solution $\hat{f}^B$, at every point in time at most one of the buffers b_i, $i = 1,\dots,\ell$, will be open. This guaranties that $\hat{f}^B$ is inflow-preserving.

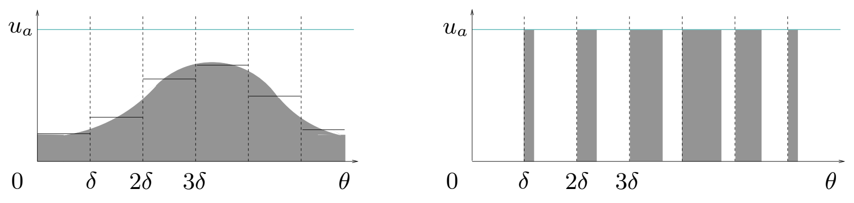

Fig. 1. Original flow rate on bow arc a and modified flow rate produced by buffering in tail(a).

As above, let $\lambda_a(\theta) := f_a^B(\theta)/u_a$ be the per capacity inflow rate of f^B on arc $a \in E_e^B$ at time θ. We partition the time horizon into intervals of length $\tilde{\delta}$, where $\tilde{\delta} := \delta/2$. Let $\lambda_{a,j}$ be the average per capacity inflow rate on arc $a \in E_e^B$ during time interval $[(j-1)\,\tilde{\delta}, j\,\tilde{\delta})$, $j = 1, \ldots, \lceil T/\tilde{\delta} \rceil$. We define the modified flow $\hat{f}^B$ as follows: During the first $\tilde{\delta}$-round, all buffers are closed. During each following $\tilde{\delta}$-round, we open the buffers in a 'round robin' fashion. More precisely, during time interval $[j\,\tilde{\delta}, (j+1)\,\tilde{\delta})$, we first open buffer b_1 for $\lambda_{a_1,j}\tilde{\delta}$ time, then buffer b_2 for $\lambda_{a_2,j}\tilde{\delta}$ time, and so on. Since f^B is weakly inflow-preserving, $\sum_{i=1}^{\ell} \lambda_{a_i,j} \leq 1$ holds and the last buffer is closed again before the end of this $\tilde{\delta}$-round. Figure 1 illustrates how the buffer changes the original inflow rate of a single bow arc a.

We show that the buffers are never empty while they are open. Consider bow arc a_i. During the interval $[(j-1)\,\tilde{\delta}, j\,\tilde{\delta})$, the flow f^B sends $\tilde{\delta}\lambda_{a_i,j}u_{a_i}$ units of flow into bow arc a_i. This is exactly the amount of flow that the corresponding buffer b_i is sending out during the succeeding interval $[j\,\tilde{\delta}, (j+1)\,\tilde{\delta})$. Hence buffer b_i is never emptied and, in particular, every unit of flow is delayed for at most $2\tilde{\delta} = \delta$ time. Note that throughout these modifications no flow is rerouted. We only make use of storage in nodes. Therefore, the cost of f^B remains unchanged. $\square$

For $\delta > 0$, we call a flow over time f^B in G^B *δ-resting* if, for every node $v \in V \backslash \{s_1, \ldots, s_k\}$, all flow arriving at v is stored there for at least δ time units before it moves on. A weakly inflow-preserving flow over time f^B in G^B which is δ-resting can easily be interpreted as an inflow-preserving flow over time $\hat{f}^B$: Consider a single arc $e \in E$ and its expansion E_e^B. Applying Lemma 1, the flow over time f^B restricted to E_e^B can be modified to an inflow-preserving flow over time such that every unit of flow is delayed by at most δ. The resting property of f^B makes up for this delay and ensures that every such flow unit can continue its way on time. Applying Observation 2, the flow $\hat{f}^B$ can then be interpreted as a flow over time f in G with inflow-dependent transit times $(\tau_e^s)_{e \in E}$.

Corollary 1. *Let f^B be a weakly inflow-preserving flow over time in G^B with time horizon T which is δ-resting. Then, f^B can be turned into a flow over time f in G with inflow-dependent transit times $(\tau_e^s)_{e \in E}$ and with the same time horizon and the same cost as f^B. Moreover, the flow over time f is given by piecewise constant functions $(f_e)_{e \in E}$ such that the number of breakpoints of f_e is bounded by $2\,|E_e^B|\,\lceil T/\delta \rceil$.*

4 A $(2+\varepsilon)$-Approximation Algorithm for Quickest Flows

In this section we present a fairly simple $(2+\varepsilon)$-approximation algorithm for the quickest multicommodity flow problem with inflow-dependent transit times. The algorithm consists of the following three main steps. First, the original transit times $(\tau_e)_{e\in E}$ are replaced by lower step functions $(\tau_e^s)_{e\in E}$ and the corresponding bow graph G^B is constructed. Then, an appropriately modified version of the $(2+\varepsilon)$-approximation algorithm presented in [4] is applied yielding a weakly inflow-preserving flow over time in G^B. Finally, the output is turned into a feasible solution to the original problem. The bow graph G^B is defined in the first step according to step functions fulfilling the requirements stated in the following observation. We will later specify the parameters $\delta, \eta > 0$ such that the size of the resulting bow graph is polynomial in the input size and $1/\varepsilon$.

Observation 4. *Let $\delta, \eta > 0$. For every non-negative, non-decreasing, and left-continuous function $\tau : [0,u] \to \mathbb{R}^+$, there exists a step function $\tau^s : [0,u] \to \mathbb{R}^+$, with*

(i) $\tau^s(x) \leq \tau(x) \leq (1+\eta)\,\tau^s(x) + \delta$ *for every $x \in [0,u]$,*
(ii) *the number of breakpoints of τ^s is bounded by* $\lceil \log_{1+\eta}(\tau(u)/\delta)\rceil + 1$.

4.1 $(2+\varepsilon)$-Approximate Quickest Weakly Inflow-Preserving Flows

Fleischer and Skutella [4] propose a $(2+\varepsilon)$-approximation algorithm for the quickest multicommodity flow problem with bounded cost and constant transit times. The method is based on an approximate length-bounded static flow computation. The same approach can be applied to the problem of finding a quickest weakly inflow-preserving multicommodity flow over time with bounded cost in the bow graph.

Let f^B be an optimal solution to this problem with minimal time horizon T. As suggested in [4], we consider the static multicommodity flow x^B in G^B which results from averaging the flow f^B on every arc $a \in E^B$ over the time interval $[0,T)$. As proven in [4], this static flow (i) satisfies a fraction of $1/T$ of the demands covered by the flow over time f^B, (ii) has cost $c(x^B) = c(f^B)/T$, and (iii) is T-length-bounded. The latter property means that the flow of every commodity $i \in K$ can be decomposed into a sum of flows on s_i-t_i-paths such that the length $\tau(P) := \sum_{a\in P}\tau_a$ of any such path P is at most T. Since f^B is weakly inflow-preserving, so is x^B, i.e., its *per capacity flow values* $\lambda_a := x_a^B/u_a$, $a \in E^B$, satisfy $\sum_{a\in E_e^B}\lambda_a \leq 1$ for every arc $e \in E$. We refer to this property as property (iv).

Any static flow x in G^B meeting requirements (i) – (iv) can be turned into a weakly inflow-preserving flow over time g in G^B meeting the same demands at the same cost as f^B within time $2T$: Send flow into every s_i-t_i-path P given by the length-bounded path decomposition of x at the corresponding flow rate $x_{P,i}$ for exactly T time units; wait for at most another T time units until all flow has arrived at its destination. Since $g_a(\theta)/u_a$ is always upper-bounded by x_a/u_a, it follows from property (iv) that g is weakly-inflow preserving. Thus, g is a 2-approximate solution to the problem under consideration.

Unfortunately, computing the T-length-bounded flow x is NP-hard, even for the special case of a single commodity [9]. Yet, as discussed in [4], the T-length-bounded multicommodity flow problem can be approximated within arbitrary precision in polynomial time by slightly relaxing the length bound T. It is easy to generalize this observation to length-bounded, weakly inflow-preserving flows. This finally yields a $(2+\varepsilon)$-approximate solution.

Lemma 2. *Assume that there exists a weakly inflow-preserving multicommodity flow over time with time horizon T and cost at most C. Then, for every $\varepsilon > 0$, a weakly inflow-preserving multicommodity flow over time with time horizon at most $(2+\varepsilon)T$ and cost at most C can be computed in time polynomial in the input size and $1/\varepsilon$.*

If all transit time functions τ_e are constant, the $(2+\varepsilon)$-approximation algorithm in Lemma 2 and the one presented in [4] basically coincide. In [4], an example is given which shows that the performance guarantee of both algorithms is not better than 2.

4.2 $(2+\varepsilon)$-Approximate Quickest Flows with Inflow-Dependent Transit Times

So far, we have presented an algorithm to compute a $(2+\varepsilon)$-approximate solution to the quickest multicommodity flow problem in the relaxed model of weakly inflow-preserving flows over time. Such a solution has a simple structure, namely it is generated from a path decomposition of a static flow in the bow graph. We will use this property to turn such a flow into a solution to the original problem. Throughout this modification we will make sure that the time horizon only increases by a small factor.

Let f^B be a weakly inflow-preserving multicommodity flow over time with time horizon T^B in G^B, which is generated from a static flow x^B as described in the last section. In particular, x^B is weakly inflow-preserving and has a length-bounded path decomposition. Let $\mathcal{P}_i$ denote the set of s_i-t_i-paths from the length-bounded path decomposition of x^B and $\mathcal{P} := \cup_{i=1}^k \mathcal{P}_i$.

Lemma 3. *The flow over time f^B can be turned into a flow over time f in G with inflow-dependent transit times $(\tau_e)_{e\in E}$ and time horizon T, where T is bounded from above by $(1+\eta)T^B + 2n\delta$.*

We are now ready to state the main result of this section.

Theorem 1. *For the quickest multicommodity flow problem with inflow-dependent transit times and bounded cost, there exists a polynomial time algorithm that, for any $\varepsilon > 0$, finds a solution of the same cost as optimal with time horizon at most $2+\varepsilon$ times the optimal time horizon T^*.*

5 An FPTAS for Quickest Flows

In this section we present an FPTAS for the quickest multicommodity flow problem with inflow-dependent transit times and bounded cost. We use ideas similar to the ones employed in [5] for the problem with fixed transit times. The FPTAS is based on a static weakly inflow-preserving flow computation in a condensed time-expanded bow graph.

Theorem 2. *There is an FPTAS for the quickest multicommodity flow problem with inflow-dependent transit times and bounded cost.*

5.1 The Algorithm

To state our algorithm and prove its correctness we define the following two bow graphs: Given $G = (V, E)$ with transit time functions $(\tau_e)_{e \in E}$ and a time horizon T, let $G^{\downarrow}$ denote the *lower bow graph* constructed from the lower step functions $\tau_e^{\downarrow}(x) := \lfloor \tau_e(x)/\Delta \rfloor \, \Delta$, for $e \in E$, $x \in [0, u_e]$. Here, $\Delta := \varepsilon^2 T/n$ for a given small constant $\varepsilon > 0$ (we assume that n/ε^2 is integral such that T is a multiple of Δ). That is, $\tau_e(x)$ is rounded down to the nearest multiple of Δ. By choice of Δ, the size of $G^{\downarrow}$ is polynomially bounded since we can delete all arcs with transit times greater than T. The second graph is the 2Δ*-lengthened bow graph*, denoted by $G^{\uparrow\uparrow}$, which is constructed from $G^{\downarrow}$ by lengthening the transit time of each arc by 2Δ. The corresponding transit time step functions are given by $\tau_e^{\uparrow\uparrow}(x) := \tau_e^{\downarrow}(x) + 2\Delta$, for $e \in E$, $x \in [0, u_e]$.

Let the *fan graph* $G^F = (V^F, E^F)$ be the Δ-condensed time-expansion of $G^{\uparrow\uparrow}$ for time horizon T (see Section 2). Each arc $e \in E$ is represented in the bow graph $G^{\uparrow\uparrow}$ by its expansion $E_e^{\uparrow\uparrow}$. Thus, the fan graph contains, for each time $\theta \in S := \{0, \Delta, \ldots, T - \Delta\}$, a 'fan' of arcs $E_e^F(\theta) := \{a(\theta) : a \in E_e^{\uparrow\uparrow},\ \theta + \tau_a \in S\}$, where $a(\theta) = (v(\theta), w(\theta + \tau_a))$. For a static flow x in G^F, we define $\lambda_a(\theta) := x_{a(\theta)}/u_{a(\theta)}$ to be the per capacity inflow value on arc $a(\theta) \in E^F$. With these definitions, the concept of (weakly) inflow-preserving flows directly carries over to static flows in G^F. Moreover, the problem of computing a weakly inflow-preserving static flow in G^F can easily be formulated as a linear program. Take a standard network flow formulation and add an extra constraint for each fan in G^F. In particular, such a flow can be computed in polynomial time. Note that any (weakly) inflow-preserving *static flow* in G^F directly corresponds to a (weakly) inflow-preserving *flow over time* in $G^{\uparrow\uparrow}$, as described in Section 1.

Let T^* denote the time horizon of a quickest flow with inflow-dependent transit times in G. We can now give an overview of our algorithm:

FPTAS FOR QUICKEST FLOWS WITH INFLOW-DEPENDENT TRANSIT TIMES

1. Guess T such that $T^* \leq T \leq (1 + \mathcal{O}(\varepsilon))T^*$. This is done via geometric mean binary search, starting with good upper and lower bounds, obtained, e.g., with help of the $(2 + \varepsilon)$-approximation in Section 4.
2. Construct the fan graph G^F for time horizon T and compute a weakly inflow-preserving *static* multicommodity flow satisfying all demands at minimum cost.
3. Interpret this static flow as a weakly inflow-preserving *flow over time* in $G^{\uparrow\uparrow}$. Modify this flow to make it inflow-preserving and, from this, derive a flow over time in G with inflow-dependent transit times and time horizon at most T.

We now proceed as follows: First we discuss issues related to the running time of the algorithm and detail how step 3 is implemented. Then, in the next section, we prove that a static flow in G^F with the properties claimed in step 2 actually exists.

The upper and lower bounds obtained from the $(2 + \varepsilon)$-approximation in step 1 are within a constant factor of each other. Thus, the estimate T can be found within $\mathcal{O}(\log(1/\varepsilon))$ geometric mean binary search steps. The fan graph G^F constructed in step 2 contains $\mathcal{O}(n^2/\varepsilon^2)$ nodes and $\mathcal{O}(mn^2/\varepsilon^4)$ arcs; note that each fan contains

$\mathcal{O}(n/\varepsilon^2)$ arcs, potentially one for each layer of G^F. Therefore, the static flow in G^F can be computed in polynomial time. We now go into the details of step 3. As mentioned before, interpreting the static flow in G^F as a weakly inflow-preserving flow over time in $G^{\uparrow\uparrow}$ is done in the canonical way, as described in Section 1. If we now shorten all arcs of $G^{\uparrow\uparrow}$ by Δ (we refer to the resulting bow graph as $G^{\uparrow}$), we obtain a weakly inflow-preserving flow over time in $G^{\uparrow}$which is Δ-resting. Applying Corollary 1, we derive an inflow-preserving flow over time in $G^{\uparrow}$. Finally, by Observation 1, we get a flow over time in G with inflow-dependent transit times $(\tau_e)_{e\in E}$ with time horizon at most T. Clearly, step 3 can be done in polynomial time.

5.2 Transforming a Flow over Time in G to a Static Flow in G^F

In this section we prove that our algorithm actually is an FPTAS by showing that a feasible flow as claimed in step 2 exists. To this end, we transform a quickest flow in G with inflow-dependent transit times to a weakly inflow-preserving static flow in G^F and thereby lengthen the time horizon by at most a factor of $1 + \mathcal{O}(\varepsilon)$. This transformation is done in several steps which are illustrated in the following diagram:

$$\begin{tikzcd} \boxed{\begin{array}{l}\textit{infl.-dep.}\text{ flow}\\ \text{over time in}\\ G\text{, time}\\ \text{horizon } T^*\end{array}} \arrow[r, "❶"] & \boxed{\begin{array}{l}\textit{infl.-pres.}\text{ flow}\\ \text{over time}\\ \text{in } G^{\downarrow}\text{, time}\\ \text{horizon } T^*\end{array}} \arrow[r, "❷"] & \boxed{\begin{array}{l}\textit{weakly}\text{ infl.-pres.}\\ \text{flow over time}\\ \text{in } G^{\uparrow\uparrow}\text{, time}\\ \text{horizon} \leq T\end{array}} \arrow[r, "❸"] & \boxed{\begin{array}{l}\text{weakly infl.-pres.}\\ \textit{static}\text{ flow in}\\ G^F\text{, time}\\ \text{horizon} \leq T\end{array}} \end{tikzcd}$$

With Observation 3, step ❶ is easy to see. For step ❸, flow in $G^{\uparrow\uparrow}$ is mapped to G^F as described in Section 1: the total flow entering arc $a \in E^{\uparrow\uparrow}$ in the interval $[\theta, \theta + \Delta)$ is assigned to $a(\theta) \in E^F$, for $\theta \in S$. Clearly, if the flow was (weakly) inflow-preserving in $G^{\uparrow\uparrow}$, it will be weakly inflow-preserving in G^F, too. Step ❷ is the most interesting but also the most intricate one. It is done similarly to [5] by carefully averaging flow to derive an 'almost feasible' flow, then subsequently sending less to obtain a feasible flow and finally increasing the time horizon to meet the demands (we refer to [5] for details). We can adopt this method since the transit times in bow graphs $G^{\downarrow}$ and $G^{\uparrow\uparrow}$ are constant. However, in contrast to [5], our flows must have the additional property of being weakly inflow-preserving.

Lemma 4. *A (weakly) inflow-preserving flow over time f in $G^{\downarrow}$ with time horizon T^* can be transformed into a* weakly *inflow-preserving flow over time in $G^{\uparrow\uparrow}$ with time horizon at most $T := (1 + \mathcal{O}(\varepsilon))T^*$ and the same cost as f.*

This concludes the proof of Theorem 2.

6 Complexity

Theorem 3. *The quickest s-t-flow problem with inflow-dependent transit times, with or without storage of flow at intermediate nodes, is NP-hard in the strong sense.*

The proof uses a reduction from the well-known NP-complete problem 3-PARTITION.

References

[1] J. E. Aronson. A survey of dynamic network flows. *Annals of Operations Research*, 20:1–66, 1989.

[2] R. E. Burkard, K. Dlaska, and B. Klinz. The quickest flow problem. *ZOR — Methods and Models of Operations Research*, 37:31–58, 1993.

[3] M. Carey and E. Subrahmanian. An approach to modelling time-varying flows on congested networks. *Transportation Research B*, 34:157–183, 2000.

[4] L. Fleischer and M. Skutella. The quickest multicommodity flow problem. In W. J. Cook and A. S. Schulz, editors, *Integer Programming and Combinatorial Optimization*, volume 2337 of *Lecture Notes in Computer Science*, pages 36–53. Springer, Berlin, 2002.

[5] L. Fleischer and M. Skutella. Minimum cost flows over time without intermediate storage. In *Proceedings of the 14th Annual ACM–SIAM Symposium on Discrete Algorithms*, pages 66–75, Baltimore, MD, 2003.

[6] L. R. Ford and D. R. Fulkerson. Constructing maximal dynamic flows from static flows. *Operations Research*, 6:419–433, 1958.

[7] L. R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, Princeton, NJ, 1962.

[8] A. Hall, S. Hippler, and M. Skutella. Multicommodity flows over time: Efficient algorithms and complexity. In *Proceedings of the 30th International Colloquium on Automata, Languages and Programming (ICALP)*, Eindhoven, The Netherlands, 2003. To appear.

[9] G. Handler and I. Zang. A dual algorithm for the constrained shortest path problem. *Networks*, 10:293–310, 1980.

[10] B. Hoppe. *Efficient dynamic network flow algorithms*. PhD thesis, Cornell University, 1995.

[11] B. Hoppe and É. Tardos. The quickest transshipment problem. *Mathematics of Operations Research*, 25:36–62, 2000.

[12] B. Klinz and G. J. Woeginger. Minimum cost dynamic flows: The series-parallel case. In E. Balas and J. Clausen, editors, *Integer Programming and Combinatorial Optimization*, volume 920 of *Lecture Notes in Computer Science*, pages 329–343. Springer, Berlin, 1995.

[13] E. Köhler, K. Langkau, and M. Skutella. Time-expanded graphs for flow-dependent transit times. In *Proceedings of the 10th Annual European Symposium on Algorithms (ESA)*, volume 2461 of *Lecture Notes in Computer Science*, pages 599–611. Springer, Berlin, 2002.

[14] E. Köhler and M. Skutella. Flows over time with load-dependent transit times. In *Proceedings of the 13th Annual ACM–SIAM Symposium on Discrete Algorithms*, pages 174–183, San Francisco, CA, 2002.

[15] N. Megiddo. Combinatorial optimization with rational objective functions. *Mathematics of Operations Research*, 4:414–424, 1979.

[16] W. B. Powell, P. Jaillet, and A. Odoni. Stochastic and dynamic networks and routing. In M. O. Ball, T. L. Magnanti, C. L. Monma, and G. L. Nemhauser, editors, *Network Routing*, volume 8 of *Handbooks in Operations Research and Management Science*, chapter 3, pages 141–295. North–Holland, Amsterdam, The Netherlands, 1995.

[17] B. Ran and D. E. Boyce. *Modelling Dynamic Transportation Networks*. Springer, Berlin, 1996.