An Efficient Algorithm for the Evacuation Problem in a Certain
Class of Networks with Uniform Path-Lengths

Naoyuki Kamiyama∗ Naoki Katoh† Atsushi Takizawa‡

October 13, 2008

Abstract

In this paper, we consider the evacuation problem in a network which consists of a directed
graph with capacities and transit times on its arcs. This problem can be solved by the
algorithm of Hoppe and Tardos [8] in polynomial time. However their running time is high-
order polynomial, and hence is not practical in general. Thus, it is necessary to devise a
faster algorithm for a tractable and practically useful subclass of this problem. In this paper,
we consider a network with a sink $s$ such that (i) for each vertex $v \neq s$ the sum of the transit
times of arcs on any path from $v$ to $s$ takes the same value, and (ii) for each vertex $v \neq s$ the
minimum $v$-$s$ cut is determined by the arcs incident to $s$ whose tails are reachable from
$v$. This class of networks is a generalization of grid networks studied in the paper [11]. We
propose an efficient algorithm for this network problem.

1 Introduction

In December 2004, the Sumatra-Andaman earthquake occurred. It triggered tsunamis, and
tragedy fell upon many people. Furthermore, the Sichuan earthquake recently occurred in
China. Not only earthquakes but also diverse disasters occurred and caused serious damages in
many countries. Therefore, it is very important to establish crisis management systems against
large-scale disasters such as big earthquakes, conflagrations and tsunamis to secure evacuation
pathways and to effectively guide residents to a safe place. The problem of finding the most
effective plan to evacuate people to a safe place has been modelled as the evacuation problem
by using dynamic flows [7]. In the evacuation problem, we are given a directed graph $D = (V, A)$
which consists of a vertex set $V$ with the supply $b(v)$ on each $v \in V$, an arc set $A$ with the
capacity $c(e)$ and the transit time $\tau(e)$ on each $e \in A$ and a sink $s \in V$. If we consider urban
evacuation, vertices model buildings, rooms, exits and so on, and arcs model pathways or roads.
For each $v \in V$, the supply $b(v)$ represents the number of people which exist at $v$. For each $e \in A$, the capacity $c(e)$ represents the number of people which can enter $e$ per unit time, and
the transit time $\tau(e)$ represents the time required to traverse $e$. Then, the evacuation problem
asks to find the minimum time required to send all the supplies to the sink and a dynamic flow
which attains the optimal evacuation time.

Given time horizon $T$, the decision problem of whether we can send all the supplies to the
sink within $T$ can be transformed to the maximum flow problem in the time-expanded network
which is an ordinary network introduced by Ford and Fulkerson [3]. However, the time-expanded
network consists of $O(T)$ copies of original vertices and arcs and hence does not directly lead us to an efficient algorithm. Furthermore, when we consider evacuation planning in practical situations, we need to consider memory requirement in addition to running time. Kamiyama et al. [12] showed that we can not solve a large instance with appropriate granularity of unit time by using the algorithm based on the time-expanded network through numerical examples modelling Kyoto city in Japan. More precisely, even when the input dynamic network is a $20 \times 20$ grid network in which a single supply is set to four persons and unit time is set to five seconds, the time required is about forty minutes and the number of vertices of the time-expanded network becomes more than four hundred million. (These experiments were done on a PC with Athlon64, 2.20GHz with 1.00GB memory.) If we adopt finer granularity, we can not solve this problem by using the time-expanded network on an ordinary PC.

A polynomial time algorithm for the evacuation problem was proposed by Hoppe and Tar-dos [8] for the first time. However, it requires to use a submodular function minimization algorithm as a subroutine. Their algorithm requires $O(\log(nT + B))$ submodular function minimizations for computing the optimal evacuation time where we denote by $n$ the number of vertices and $T = \max_{e \in A} \tau(e)$ and $B = \sum_{v \in V} b(v)$. Hence, the running time is high-order polynomial, and the algorithm is not practical in general. Therefore, it is necessary to devise a faster algorithm for a tractable and practically useful subclass of this problem. We should remark that Fleischer and Skutella [1] presented an FPTAS for the evacuation problem by using a maximum flow computation in a transformed time-expanded network\(^1\).

As a special case, Mamada et al. [13] gave an $O(n \log^2 n)$ algorithm for tree networks. Hall et al. [6] studied the case called uniform path-lengths where for each vertex $v \neq s$ the sum of transit times of arcs on any path from $v$ to $s$ takes the same value. They showed that in this case the time-expanded network can be condensed to the so-called condensed time-expanded network whose size is polynomial in the input size. Kamiyama et al. [11] have shown that we can compute in $O(n \log n)$ time the optimal evacuation time for a $\sqrt{n} \times \sqrt{n}$ grid network with uniform arc capacity and uniform transit time in which the arcs are directed so that the uniform path-lengths condition holds. Their algorithm is constructed by using the uniform path-lengths condition and the underlying special structure of grid networks in terms of the local arc-connectivity, and this algorithm does not explicitly use the time-expanded network. In this paper, we will generalize the class of networks to which the ideas developed in [11] can be applied. More precisely, we consider dynamic networks satisfying the following two conditions.

1. For each vertex $v \neq s$, the sum of transit times of arcs on any path from $v$ to $s$ takes the same value.
2. For each vertex $v \neq s$, the minimum $v$-$s$ cut is determined by the arcs incident to $s$ whose tails are reachable from $v$.

We call Conditions 1 and 2 uniform path-lengths and fully connected, respectively (see Figure 1).

The algorithm of [11] reduced the evacuation problem to the min-max resource allocation problem under network constraints [9], but in this paper we reduce the evacuation problem to the parametric flow problem defined on an ordinary network. Although it is known [6] that the evacuation problem in the case of uniform path-lengths can be reduced to the parametric flow problem in which the capacity of a subset of arcs is a linear function of time horizon $T$, we prove that in our case the evacuation problem can be reduced to the special case of the parametric flow

\(^1\)Precisely speaking, the paper [1] considered continuous dynamic flows such that the time step is continuous, while in this paper we use discrete dynamic flows such that the time step is discretized. The papers [13, 6] also used continuous dynamic flows. However, Mamada et al. [13] explicitly announced that their algorithm can be applied to discrete dynamic flows. For the relation between the continuous version and the discrete one, see [2].
problem studied by [4] which can be solved more efficiently than that considered in [6]. Thus, under Conditions 1 and 2, our algorithm is faster than using the condensed time-expanded network. In particular, our algorithm becomes much faster when the in-degree of the sink is small or considered to be a constant which is often the case with road networks.

The rest of this paper is organized as follows. Section 2 introduces notations and basic results. Section 3 shows the main result of this paper, i.e., an algorithm for the evacuation problem in fully connected networks with uniform path-lengths. In Section 4, we consider the case of grid networks. Section 5 concludes this paper with a further result.

2 Preliminaries

We denote by \( \mathbb{R}_+ \) and \( \mathbb{Z}_+ \) the sets of nonnegative reals and nonnegative integers, respectively. Throughout this paper, we may not distinguish between a singleton \( \{x\} \) and its element \( x \). For a set \( X \) and a singleton \( \{x\} \), the union of \( X \) and \( \{x\} \) is abbreviated by \( X + x \) while \( X \setminus \{x\} \) by \( X - x \). Given a function \( f: X \to \mathbb{R}_+ \) on a set \( X \), we use the abbreviation \( f(Y) = \sum_{x \in Y} f(x) \) for each \( Y \subseteq X \).

2.1 Directed graphs

Let \( D = (V, A) \) be a directed graph which may have parallel arcs. A vertex \( v \) is said to be reachable from a vertex \( u \) when there is a directed path from \( u \) to \( v \). We denote by \( e = uv \) an arc \( e \) whose tail and head are \( u \) and \( v \), respectively. If \( e = uv \) has no parallel arc, we may simply write \( uv \). Furthermore, for each \( e \in A \) we denote by \( t(e) \) and \( h(e) \) the tail and the head of \( e \), respectively. For each \( X \subseteq V \), let \( \delta_D(X) \) (resp. \( \varrho_D(X) \)) be the set of arcs \( e \in A \) with \( t(e) \in X \) and \( h(e) \notin X \) (resp. \( t(e) \notin X \) and \( h(e) \in X \)). For each distinct \( u, v \in V \), we denote by \( \lambda(u, v; D) \) the local arc-connectivity from \( u \) to \( v \) in \( D \), i.e., \( \lambda(u, v; D) = \min \{|\varrho_D(W)|: v \in W \subseteq V - u\} \). By Menger’s theorem, \( \lambda(u, v; D) \) is equal to the maximum number of the arc-disjoint directed paths from \( u \) to \( v \) in \( D \) (see [16, Corollary 9.1b]). For each \( X \subseteq V \), let \( D[X] \) denote the directed subgraph of \( D \) induced by \( X \).

2.2 Dynamic networks

Let \( N = (D = (V, A), c, \tau, b, s) \) be a dynamic network which consists of a directed graph \( D = (V, A) \), a capacity function \( c: A \to \mathbb{R}_+ \) which represents the upper bound for the rate of flow that enters each arc per unit time, a transit time function \( \tau: A \to \mathbb{Z}_+ \) which represents the time required to traverse each arc, a supply function \( b: V \to \mathbb{R}_+ \) which represents the supply of each vertex and a sink \( s \in V \). We define the length of a directed path \( P \) by \( \sum_{e \in P} \tau(e) \). Since we...
consider evacuation to \( s \), we assume that \( \varrho_D(s) = \emptyset \) and \( b(s) = 0 \), and \( s \) is reachable from every vertex.

We define a dynamic flow \( f: A \times \mathbb{Z}_+ \rightarrow \mathbb{R}_+ \) as follows. For each \( e \in A \) and \( \theta \in \mathbb{Z}_+ \), we denote by \( f(e, \theta) \) the flow rate entering \( e \) at the time step \( \theta \) which arrives at \( h(e) \) at time step \( \theta + \tau(e) \). We call \( f \) feasible if it satisfies the following three conditions capacity constraint, flow conservation, and demand constraint.

- **Capacity constraint.** For every \( e \in A \) and \( \theta \in \mathbb{Z}_+ \),
  \[
  0 \leq f(e, \theta) \leq c(e).
  \]

- **Flow conservation.** For every \( v \in V \) and \( \Theta \in \mathbb{Z}_+ \),
  \[
  \sum_{e \in \delta_D(v)} \sum_{\theta=0}^{\Theta} f(e, \theta) - \sum_{e \in \varrho_D(v)} \sum_{\theta=0}^{\Theta-\tau(e)} f(e, \theta) \leq b(v).
  \]

- **Demand constraint.** There exists \( \Theta \in \mathbb{Z}_+ \) with
  \[
  \sum_{e \in \varrho_D(s)} \sum_{\theta=0}^{\Theta-\tau(e)} f(e, \theta) = b(V). \tag{1}
  \]

For a feasible dynamic flow \( f \), we define the evacuation time of \( f \) by the minimum time step \( \Theta \) satisfying (1), which is denoted by \( \Theta(f) \). Then, the evacuation problem asks to find the minimum evacuation time among all feasible dynamic flows (denoted by \( \Theta(N) \)) and a feasible dynamic flow which attains the minimum evacuation time. Given time horizon \( T \), the decision version of the evacuation problem with time horizon \( T \) asks if there exists a feasible dynamic flow \( f \) with \( \Theta(f) \leq T \).

Throughout this paper, we denote \(|V|\) and \(|A|\) by \( n \) and \( m \), respectively. In a figure of a dynamic network, the pair of numbers attached to each arc indicates its capacity and transit time. Moreover, the number attached to each vertex indicates its supply (see Figure 2(a)).

### 2.3 Static networks

Let \( N_s = (D_s = (V_s, A_s), c_s, b_s, S_s) \) be a static network which consists of a directed graph \( D_s = (V_s, A_s) \), a capacity function \( c_s: A_s \rightarrow \mathbb{R}_+ \), a supply function \( b_s: V_s \rightarrow \mathbb{R}_+ \), and a set of sinks \( S_s \subseteq V_s \). We call \( f_s: A_s \rightarrow \mathbb{R}_+ \) a feasible flow if it satisfies the following two conditions capacity constraint and flow conservation.

- **Capacity constraint.** For every \( e \in A_s \),
  \[
  0 \leq f_s(e) \leq c_s(e).
  \]

- **Flow conservation.** For every \( v \in V_s \setminus S_s \),
  \[
  f_s(\delta_{D_s}(v)) - f_s(\varrho_{D_s}(v)) = b_s(v).
  \]

Throughout this paper, we denote \(|V_s|\) and \(|A_s|\) by \( n_s \) and \( m_s \), respectively. In a figure of a static network, the number attached to each arc and each vertex indicates its capacity and supply, respectively. If no number is attached to an arc, it means that its capacity is infinite. Moreover, if no number is attached to a vertex, its supply is equal to zero (see Figure 2(b)).
2.4 Time-expanded networks

In order to solve the decision version of the evacuation problem with time horizon $T$ in $\mathcal{N}$, Ford and Fulkerson [3] introduced the time-expanded network $\mathcal{N}(T)$ which is a static network in which for each $v \in V$ and $i \in \{0, \ldots, T\}$ there exists a vertex $v_i$, and for each $e = uv \in A$ and $i \in \{0, \ldots, T - \tau(e)\}$ there exists an arc $e_i = u_i v_i + \tau(e)$ whose capacity is $c(e)$, and for each $v \in V$ and $i \in \{0, \ldots, T - 1\}$ there exists a holdover arc $v_i v_{i+1}$ with infinite capacity. For each $v \in V$, the supply of $v_0$ is set to $b(v)$ and the supplies of all the other vertices $v_i$ ($i \in \{1, \ldots, T\}$) are set to zero. Let $s_T$ be a single sink of $\mathcal{N}(T)$ (see Figure 2).

![Dynamic network $\mathcal{N}$](image1)

![Time-expanded network $\mathcal{N}(7)$](image2)

Figure 2: (a) Dynamic network $\mathcal{N}$. (b) Time-expanded network $\mathcal{N}(7)$.

It is known [3] that there exists a feasible dynamic flow $f$ with $\Theta(f) \leq T$ if and only if there exists a feasible flow in $\mathcal{N}(T)$. Although we can decide if there exists a feasible flow in $\mathcal{N}(T)$ by solving the maximum flow problem, the running time is pseudo-polynomial because the size of the time-expanded network is proportional to $T$ and thus is pseudo-polynomial in the input size. This fact motivates the condensed time-expanded network which is a static network whose size is polynomial of the input size.

2.5 Dynamic networks with uniform path-lengths

In this paper, we assume that $\mathcal{N}$ satisfies uniform path-lengths condition. In a dynamic network with uniform path-lengths, if there exists a directed cycle, its length is zero. The zero length cycle is not useful in modelling real road networks. Thus, in this paper we assume that $D$ has no directed cycle. (We will discuss about the case where $D$ is allowed to have directed cycles in Section 5.) Furthermore, to avoid complicated argument, we assume $\tau(e) > 0$ for every $e \in A$. The case where there exists an arc whose transit time is zero can be similarly treated.

In this subsection, we give notations concerning dynamic networks with uniform path-lengths, and then we review the result of Hall et al. [6]. They proved that the evacuation problem in dynamic networks with uniform path-lengths can be solved by solving the parametric flow problem defined on the condensed time-expanded network.

2.5.1 Notation

For each $v \in V$, we define $l_v$ as the length of a directed path from $v$ to $s$. Notice that since $\mathcal{N}$ satisfies the uniform path-length condition, $l_v$ is well-defined for each $v \in V$. Let us arrange the distinct values of $l_v$ ($v \in V$) as $L_1 < \cdots < L_k$ where $L_1 = 0$ and $k$ is the number of the distinct path-lengths. Notice that $k = \mathcal{O}(n)$ holds. We say $v \in V$ is at level $i$ when $l_v = L_i$, which is denoted by $\text{lev}(v) = i$. Without loss of generality, we assume that $b(v) > 0$ holds for at least
one vertex \( v \in V \) with \( \text{lev}(v) = k \). For each \( i \in \{1, \ldots, k\} \), let \( V_{\leq i} \) be the set of vertices \( v \in V \) with \( \text{lev}(v) \leq i \). Letting \( L_{k+1} = T + 1 \), we partition interval \([0, T]\) into disjoint subintervals \( I_1, I_2, \ldots, I_k \) such that \( I_i = (L_i, L_i + 1, \ldots, L_{i+1} - 1) \) holds for each \( i \in \{1, \ldots, k\} \). For example, for the dynamic network illustrated in Figure 2(a) with \( T = 7 \), \((l_8, l_9, l_{10}, l_{11}) = (0, 1, 3, 6) \). Thus, we have \( k = 4 \) and \( I_1 = (0, I_2 = (1, 2), I_3 = (3, 4, 5), I_4 = (6, 7) \).

### 2.5.2 Condensed time-expanded networks

Let \( \mathcal{N}_c = (D_c = (V_c, A_c), c_c, b_c, \alpha_c) \) be the condensed time-expanded network of \( \mathcal{N} \) with time horizon \( T \) defined as follows (see Figure 3(a)). Let \( V_c = \{ v_i : v \in V \text{ and } i \in \{\text{lev}(v), \ldots, k\} \} \) and \( A_c = A^1_c \cup A^2_c \) where \( A^1_c \) contains an arc \( e_i = (u_i, v_i) \) for each \( e = uv \in A \text{ and } i \in \{\text{lev}(u), \ldots, k\} \), and \( A^2_c \) contains an arc \( v_i v_{i+1} \) for each \( v \in V \text{ and } i \in \{\text{lev}(v), \ldots, k - 1\} \). Each arc \( e_i \in A^1_c \) has the capacity \( |I_i| \cdot c(e) \), and each arc in \( A^2_c \) is a holdover arc whose capacity is infinity. For each \( v \in V \), the supply of \( v_{\text{lev}(v)} \) is set to \( b(v) \), and the supplies of all the other vertices \( v_i \ (i \in \{\text{lev}(v) + 1, \ldots, k\}) \) are set to zero. Let \( s_k \) be a single sink of \( \mathcal{N}_c \).

![Figure 3: (a) \( \mathcal{N}_c \) for \( \mathcal{N} \) in Figure 2(a) with \( T = 7 \). (b) Components in \( \mathcal{N}_c \) in (a).](image)

For each \( i \in \{1, \ldots, k\} \), let \( V_{c,i} \) be the set of \( v_i \in V_c \) with \( v \in V_{\leq i} \). Then, the following structural characterizations of \( \mathcal{N}_c \) are used in the subsequent discussion (see Figure 3(b)).

1. The condensed time-expanded network \( \mathcal{N}_c \) consists of \( k \) components such that the \( i \)-th component is \( D_{c}[V_{c,i}] \). Moreover, the \( i \)-th component \( D_{c}[V_{c,i}] \) is isomorphic to \( D[V_{\leq i}] \) and the capacity of an arc \( e_i \) in \( D_{c}[V_{c,i}] \) is \( |I_i| \cdot c(e) \).
2. Consecutive components are connected by holdover arcs.

### 2.5.3 The Evacuation problem in dynamic networks with uniform path-lengths

Hall et al. [6] showed the following theorem.

**Theorem 2.1** (Hall et al. [6]). There exists a feasible dynamic flow \( f \) with \( \Theta(f) \leq T \) in \( \mathcal{N} \) if and only if there exists a feasible flow in \( \mathcal{N}_c \) with time horizon \( T \). Furthermore, given a feasible flow \( f_c \) in \( \mathcal{N}_c \), we can construct a feasible dynamic flow \( f \) in \( \mathcal{N} \) by setting \( f(e, \theta - t_{\ell(e)}) = f_c(e_i)/|I_i| \text{ for each } i \in \{1, \ldots, k\}, \theta \in I_i \text{ and } e \in A \text{ with } \text{lev}(t(e)) \leq i \).

Thus, the evacuation problem in \( \mathcal{N} \) can be solved by computing the minimum \( T \) such that there exists a feasible flow in \( \mathcal{N}_c \), and hence by regarding \( T \) as a parameter we can solve the evacuation problem in \( \mathcal{N} \) by solving the parametric flow problem defined as follows.

<table>
<thead>
<tr>
<th>Problem</th>
<th>The parametric flow problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>a static network ( \mathcal{N}_s = (D_s = (V_s, A_s), c_s, b_s, S_s) ) in which the capacity of ( e \in A_s ) is represented by ( a_e + g_e \xi ) where ( a_e ) is a real constant, ( g_e ) is a nonnegative constant, and ( \xi ) is a nonnegative parameter;</td>
</tr>
<tr>
<td>Output</td>
<td>the minimum value of ( \xi ) such that there exists a feasible flow in ( \mathcal{N}_s ).</td>
</tr>
</tbody>
</table>
It is known [15] that the parametric flow problem can be solved by solving the maximum flow problem $O(m_s)$ times in $N_s$. Since for each $e \in A$ the capacity of $e_k$ is equal to $|I_k| \cdot c(e) = (T - L_k + 1) \cdot c(e)$ by $L_{k+1} = T + 1$, the evacuation problem in $N$ can be solved by solving the parametric flow problem in $N_c$. Strictly speaking, recalling that what we want to compute is the minimum nonnegative integer $T$ such that there exists a feasible flow in $N_c$, the evacuation problem in $N$ and the parametric flow problem in $N_c$ are not equivalent. This is because the optimal value of the parametric flow problem may not be an integer. However, letting $T^{\ast}$ be the optimal value of the parametric flow problem in $N_c$, $\Theta(N) = [T^{\ast}]$ holds since a feasible dynamic flow $f$ with $\Theta(f) \leq T$ exists in $N$ if and only if there exists a feasible flow in $N_c$ with time horizon $T$.

By $|V_c| = O(kn)$ and $|A_c| = O(km)$, the result of [6] implies that the evacuation problem in $N$ can be solved in $O(k^3m^2n\log(kn^2/m))$ time by the algorithm of [5] solving the maximum flow problem in a static network with $n_s$ vertices and $m_s$ arcs in $O(m_sn_s\log(n_s^2/m_s))$ time.

3 The Evacuation Problem in Fully Connected Networks

In this section, we assume that $N$ not only satisfies the uniform path-lengths condition but also satisfies the fully connected condition. For each $v \in V$, let $R_v$ be the set of vertices $w \in V$ such that there exists $e \in g_D(s)$ with $t(e) = w$ and $w$ is reachable from $v$. Recall that $N = (D = (V, A), c, \tau, b, s)$ is called fully connected if for each $v \in V - s$ the minimum $v$-$s$ cut is determined by the arcs incident to $s$ whose tails are reachable from $v$. That is, the value of the minimum $v$-$s$ cut is equal to $\sum\{c(e) : e \in g_D(s) \text{ with } t(e) \in R_v\}$. In the subsequent discussion, we concentrate on the unit capacity case, i.e., the capacity of every arc is equal to one. (We will show that our algorithm can be applied to the integral capacity case in Sections 3.7.) In the unit capacity case, $N$ is fully connected if and only if for each $v \in V - s$

$$\lambda(v, s; D) = |\{e \in g_D(s) : t(e) \in R_v\}|.$$  

(2)

We will prove that the evacuation problem in $N$ can be solved by solving the restricted parametric flow problem defined in the following subsection.

3.1 The restricted parametric flow problem

Given a static network $N_s = (D_s = (V_s, A_s), c_s, b_s, S_s)$ in which for each $e \in A_s$

$$c_s(e) = \begin{cases} \alpha_e + g_e \xi & \text{where } \alpha_e \text{ is a constant, } g_e \text{ is a nonnegative constant and } \xi \text{ is a nonnegative parameter,} \\ \text{constant,} & \text{if } e \in g_{D_s}(S_s), \\ \text{constant,} & \text{otherwise}, \end{cases}$$

the restricted parametric flow problem asks for finding the minimum value of $\xi$ such that there exists a feasible flow in $N_s$. This problem can be transformed into the parametric maximum flow problem studied in [4] by introducing a super source vertex $q$ and arcs from $q$ to every vertex $v$ with $b_s(v) > 0$ such that the capacity of an arc $qv$ is set to $b_s(v)$. It is then easy to see that there exists a feasible flow in $N_s$ for a fixed $\xi$ if and only if the maximum flow value from $q$ to $S_s$ in the transformed problem is equal to $b_s(V_s)$.

Lemma 3.1 (Gallo et al. [4]). The maximum flow value from $q$ to $S_s$ in the transformed network is a non-decreasing piecewise linear concave function in $\xi$ (which is denoted by $\kappa(\xi)$), and the largest breakpoint of $\kappa(\xi)$ can be found in $O(m_sn_s\log(n_s^2/m_s))$ time.
Lemma 3.2. We can determine whether there exists \( \xi \) such that there exists a feasible flow in \( \mathcal{N}_s \), and the minimum such value can be found if one exists in \( O(m_s n_s \log(n_s^2/m_s)) \) time.

Proof. By the above discussion, there exists a feasible flow in \( \mathcal{N}_s \) when there exists \( \xi \) such that maximum flow value in the transformed problem is equal to \( b_s(V_s) \). On the other hand, the maximum flow value in the transformed problem can not exceed \( b_s(V_s) \). Thus when \( \xi \) is larger than the largest breakpoint the slope of \( \kappa(\xi) \) is zero and \( \kappa(\xi) \) is less than or equal to \( b_s(V_s) \).

Checking whether there exists \( \xi \) such that there exists a feasible flow in \( \mathcal{N}_s \), reduces to computing the largest breakpoint of \( \kappa(\xi) \). Moreover, if there exists \( \xi \) such that there exists a feasible flow in \( \mathcal{N}_s \), the minimum value of such \( \xi \) is equal to the largest breakpoint of \( \kappa(\xi) \). Thus, the lemma follows from Lemma 3.1.

As was defined in Section 2.5, the capacity of \( e_k \in A^1 \) in \( \mathcal{N}_c \) contains the parameter \( T \), i.e., it is a linear function of \( T \). In Figure 3(a), regarding \( T \) as the parameter, we have \( c_c(u_4 s_4) = 2(T-5) \), \( c_c(w_1 s_4) = 7(T-5) \), \( c_c(v_4 u_4) = 6(T-5) \) and \( c_c(v_4 w_4) = 4(T-5) \). Thus, the arcs which are not incident to the sink have the parametric capacities. Therefore, we can not in general reduce the evacuation problem to the restricted parametric flow problem.

3.2 Property of fully connected networks

In this subsection, we introduce a property of a fully connected network which plays a crucial role in the sequel. For each \( e \in g_D(s) \), let \( V^e = \{v \in V: t(e) \text{ is reachable from } v\} \cup \{s\} \).

Lemma 3.3. There exist arc-disjoint in-trees \( D^e = (V^e, A^e) \) \( (e \in g_D(s)) \) rooted at \( s \) such that \( D^e \) spans \( V^e \) if and only if (2) holds, i.e., \( \mathcal{N} \) is fully connected.

For example, a directed graph \( D \) in Figure 4(a) contains two arc-disjoint in-trees \( D^e \) and \( D^f \) which satisfy the above condition (see Figures 4(b) and (c)).

![Diagram](a)  \quad ![Diagram](b)  \quad ![Diagram](c)

Figure 4: (a) \( D = (V, A) \). (b) \( D^e = (V^e, A^e) \). (c) \( D^f = (V^f, A^f) \).

Proof. It is not difficult to see that the “only if-part” holds. We then prove the “if-part”. We prove that there exist in-trees satisfying the lemma statement for \( D[V_{\leq t}] \) by induction on \( i \in \{2, \ldots, k\} \). For \( i = 2 \), this lemma clearly holds.

Assuming that the lemma holds for \( D[V_{\leq t}] \) with \( t \geq 2 \), we will prove the lemma also holds for \( D[V_{\leq t+1}] \). For an arbitrary \( v \in V_{\leq t+1} \setminus V_{\leq t} \), we define a bipartite graph \( G = (V^+, V^-, E) \) as follows. Let \( V^+ \) and \( V^- \) represent arcs of \( \delta_D(v) \) and arcs \( e \in g_D(s) \) with \( t(e) \in R_v \), respectively. For each \( e \in \delta_D(v) \) (resp. \( e \in g_D(s) \) with \( t(e) \in R_v \), respectively), we denote by \( v^+(e) \) (resp. \( v^-(e) \)) the vertex in \( V^+ \) (resp. \( V^- \)) corresponding to \( e \). For each \( e \in \delta_D(v) \) with \( h(e) \neq s \) (resp. \( h(e) = s \)) and \( e' \in g_D(s) \) with \( t(e') \in R_v \), \( v^+(e) \) and \( v^-(e') \) are joined by an edge in \( E \) if and only if \( t(e') \) is reachable from \( h(e) \) (resp. \( e = e' \)) (see Figure 5). Notice that we assume \( \tau(e) > 0 \) for every \( e \in A \), all heads of arcs in \( \delta_D(v) \) belong to \( V_{\leq t} \).

In order to prove that the lemma holds for \( D[V_{\leq t} \cup \{v\}] \), it is sufficient to prove that there exists a matching \( \mathcal{M} \) which saturates \( V^- \) in \( G \) because if there exists such \( \mathcal{M} \), by the induction
hypothesis we can extend arc-disjoint in-trees rooted at \( v \) satisfying the lemma in \( D[V\leq t] \) to the ones in \( D[V\leq t \cup \{v\}] \), and thus the lemma follows. By Hall’s theorem \([14]\), it is known that there exists a matching which saturates all vertices of \( V^- \) if and only if \( |W| \leq |Ne(W)| \) holds for every \( W \subseteq V^- \) where \( Ne(W) \) is the set of vertices adjacent to some element of \( W \). Thus, what remains is to prove that if there exists \( W \subseteq V^- \) with \( |W| > |Ne(W)| \), this contradicts (2). A directed path from \( v \) to \( s \) containing \( e \in g_D(s) \) with \( t(e) \in R_v \) and \( v^{-}(e) \in W \) has to contain \( e' \in \delta_D(v) \) with \( v^{+}(e') \in Ne(W) \), and thus there can not exist \( \{e \in g_D(s) : t(e) \in R_v\} \) arc-disjoint directed paths by \( |W| > |Ne(W)| \). This completes the proof. \[\square\]

As was seen in the proof of Lemma 3.3, we can obtain \( D^e \) \((e \in g_D(s))\) by computing a matching in the bipartite graph \( G \) for all \( v \in \mathcal{V} \). However, the time required is very expensive. Thus, in our algorithm, we do not explicitly compute \( D^e \) \((e \in g_D(s))\).

### 3.3 Optimality of disjoint flows

Now we return to the evacuation problem in \( \mathcal{N} \). Let us first consider the case where the amount of the supply of \( v \) which reaches \( s \) through \( e \) is fixed (which is denoted by \( b^e(v) \)) for each \( v \in \mathcal{V} \) and \( e \in g_D(s) \). More formally, let us fix \( b^e : V \rightarrow \mathbb{R}_+ \) \((e \in g_D(s))\) satisfying the following two conditions.

1. For each \( v \in \mathcal{V} \), \( \sum \{b^e(v) : e \in g_D(s)\} = b(v) \) holds,
2. For each \( v \in \mathcal{V} \) and \( e \in g_D(s) \) with \( v \notin V^e \), \( b^e(v) = 0 \) holds.

Intuitively speaking, \( b^e(v) \) represents the assignment of the amount of the supply of \( v \) to \( D^e \) which is an in-tree in Lemma 3.3 for each \( e \in g_D(s) \). For each \( e \in g_D(s) \), let \( \mathcal{N}^e = (D^e, (V^e, A^e), \epsilon^e, \tau^e, b^e, s) \) where \( \epsilon^e \) and \( \tau^e \) respectively stand for \( e \) and \( \tau \) whose domain is restricted to \( A^e \). Since \( A^e(e \in g_D(e)) \) are arc-disjoint, the dynamic flow obtained by combining all optimal dynamic flows in the evacuation problem in \( \mathcal{N}^e \) for each \( e \in g_D(s) \) is feasible in \( \mathcal{N} \). Furthermore, the following theorem says that in this case the flow obtained by combining all optimal dynamic flows in \( \mathcal{N}^e \) \((e \in g_D(s))\) is optimal for the evacuation problem in \( \mathcal{N} \).

**Theorem 3.4.** Under the constraint that the amount of \( b(v) \) which reaches \( s \) through \( e \) is fixed to \( b^e(v) \) for each \( v \in \mathcal{V} \) and \( e \in g_D(s) \), \( \Theta(\mathcal{N}) \) is equal to \( \max \{\Theta(\mathcal{N}^e) : e \in g_D(s)\} \).

**Proof.** Suppose that there exists a feasible dynamic flow \( f \) in \( \mathcal{N} \) with \( \Theta(f) < \max \{\Theta(\mathcal{N}^e) : e \in g_D(s)\} \). Let us decompose \( f \) into \( \tilde{f}^e (e \in g_D(s)) \) such that \( \tilde{f}^e \) represents a dynamic flow which enters into \( s \) through \( e \). Notice that such decomposition may not be unique. We choose an arbitrary one. For each \( e \in g_D(s) \), let \( D^{\tilde{f}^e} = (V^{\tilde{f}^e}, A^{\tilde{f}^e}) \) denote a subgraph such that \( A^{\tilde{f}^e} \subseteq A \) is the set of arcs \( e' \) which \( \tilde{f}^e \) uses, i.e., \( \tilde{f}^e(e', \theta) > 0 \) for some \( \theta \in \mathbb{Z}_+ \). For each \( e \in g_D(s) \), let
\( \bar{N}^e = (\bar{D}^e = (V^e, \bar{A}^e), \bar{c}^e, \bar{b}^e, \bar{\tau}^e, s) \) where \( \bar{c}^e \) and \( \bar{\tau}^e \) respectively represent \( c \) and \( \tau \) whose domain is restricted to \( \bar{A}^e \). In order to prove the theorem, we show that for each \( e \in \rho_D(s) \)

\[
\Theta(\bar{N}^e) = \Theta(N^e).
\] (3)

In order to prove (3), we need the following new dynamic network \( P^e \) (see Figure 6). We first show how the underlying graph \( D_p = (V_p, A_p) \) of \( P^e \) is constructed. Combining vertices \( v \in V^e \) with \( \text{lev}(v) = i \) into a single vertex \( p^i \) and arranging vertices \( p^k, \ldots, p^1 \) along the directed path in this order, we obtain \( D_p = (V_p, A_p) \) where \( V_p = \{ p^i : i \in \{1, \ldots, k\} \} \), \( A_p = \{ p^{i+1}p^i : i \in \{1, \ldots, k-1\} \} \) and \( p^1 \) is the sink of \( P^e \). The supply of \( p^i \) is the sum of supplies \( b^e(v) \) over \( v \in V^e \) with \( \text{lev}(v) = i \). Notice that even if there exists no vertex \( v \in V^e \) with \( \text{lev}(v) = i \), we prepare \( p^i \) whose supply is zero. The capacity of \( p^2p^1 \) is one, and the capacities of all the other arcs is set to infinity. The transit time of \( p^{i+1}p^i \) is determined in such a way that the length of the subpath from \( p^i \) to \( p^1 \) is the same as that from \( v \in V \) with \( \text{lev}(v) = i \) to the sink \( s \) in \( \bar{N}^e \).

Figure 6: (a) Input network \( N \) where the numbers attached to each arc represent the transit time. For each \( v \in V \), let \( b^e(v) = b(v) \). (b) \( \bar{N}^e \). (c) \( P^e \).

Then, we will show that \( \bar{N}^e, N^e \) and \( P^e \) are equivalent in terms of the minimum evacuation time, which implies (3).

**Lemma 3.5.** \( \Theta(\bar{N}^e) = \Theta(N^e) = \Theta(P^e) \).

**Proof.** The justification behind the lemma is intuitively explained as follows. Since the capacity of every arc in \( N^e \) is equal to one and \( D^e \) is an in-tree, the bottleneck lies in the arc \( e \) (the unique arc incident to \( s \) in \( D^e \)). This implies that even if we increase the capacity of all the other arcs to infinity, the minimum evacuation time remains the same. Let \( \bar{N}^e \) stand for the dynamic network such that the capacities of all arcs except \( e \) are increased to infinity. Then, we have \( \Theta(N^e) = \Theta(\bar{N}^e) \) (we give a formal proof later). Since \( P^e \) and \( \bar{N}^e \) are essentially the same, it is not difficult to see that \( \Theta(P^e) = \Theta(\bar{N}^e) \), and thus we have \( \Theta(P^e) = \Theta(N^e) \). Furthermore, we can apply the same argument to \( \bar{N}^e \) by considering an arbitrary spanning in-tree network in \( \bar{N}^e \). Thus, the lemma follows.

Now we will give a more formal proof of \( \Theta(N^e) = \Theta(\bar{N}^e) \). In order to prove \( \Theta(N^e) = \Theta(\bar{N}^e) \), it is sufficient to show \( \Theta(N^e) \leq \Theta(\bar{N}^e) \) since the other direction is obvious. Let us consider a dynamic flow \( f_\infty \) in \( N^e \) which attains \( \Theta(\bar{N}^e) \), and we show that we can construct a feasible dynamic flow \( f \) in \( N^e \) which attains \( \Theta(N^e) \). Now let us focus on \( f_\infty \) that enters into the arc \( e \) in \( N^e \). Since the capacity of \( e \) is equal to one, at most one unit of the supply enters into \( e \) at one time. Let us assume that we know from which vertex the supply originally comes. Let us consider \( f_\infty(e, \theta) \) \( (\theta \in \{0, \ldots, \Theta(\bar{N}^e) - \tau(e)\}) \), and decompose it into \( f_\infty(e, \theta; v) \) \((v \in V^e)\) such that \( f_\infty(e, \theta; v) \) is equal to the amount of the supply of \( f_\infty(e, \theta) \) which originally comes from \( v \). We will construct a feasible flow \( f \) in \( N^e \) such that \( f(e, \theta) = f_\infty(e, \theta) \) holds for every \( \theta \). More precisely, we will first construct the subflow \( f(\cdot, v) \) of \( f \) for every \( v \in V^e \) that represents the flow of supplies which originate from \( v \) in such a way that \( f(e, \theta; v) = f_\infty(e, \theta; v) \) holds for every \( \theta \). If the arc \( e' \) is on the unique path in \( D^e \) from \( v \) to \( t(e) \), \( f(e', \theta; v) \) is obtained by simply
translating \( f(e, \theta; v) \) by \(-\alpha\) where \( \alpha \) is the distance from \( t(e') \) to \( t(e) \). Otherwise, \( f(e', \theta; v) \) is set to zero for all \( \theta \). The feasible flow \( f \) in \( \mathcal{N}^e \) is then defined as \( f(e', \theta) = \sum_{v \in V^e} f(e', \theta; v) \) for every \( e' \in A^e \) and \( \theta \). Since the capacity of \( e \) in \( \mathcal{N}_\infty^e \) is equal to one, \( \sum_{v \in V^e} f(e', \theta; v) \leq 1 \) holds for every \( e' \in A^e \) and every \( \theta \). Notice that \( \sum_{v \in V^e} f(e, \theta; v) \leq 1 \) holds for any \( \theta \) since the capacity of \( e \) is equal to one in \( \mathcal{N}_\infty^e \), and \( f \) is constructed by translating \( f_\infty(e, \theta) \). This implies that the unit capacity constraint is satisfied for \( f \) in \( \mathcal{N}^e \). It is not difficult to see that \( f \) satisfies the other constraints.

Now we return to the proof of the theorem. For each \( e \in g_D(s) \), since \( f^e \) is a feasible dynamic flow in \( \mathcal{N}^e \), \( \Theta(f^e) \geq \Theta(\mathcal{N}^e) \) holds. Hence,

\[
\Theta(f) = \max \{ \Theta(f^e) : e \in g_D(s) \} \geq \max \{ \Theta(\mathcal{N}^e) : e \in g_D(s) \}.
\]

By Lemma 3.5, \( \Theta(\mathcal{N}^e) = \Theta(\mathcal{N}'^e) \) holds for each \( e \in g_D(s) \). Thus, \( \max \{ \Theta(\mathcal{N}^e) : e \in g_D(s) \} = \max \{ \Theta(\mathcal{N}^e) : e \in g_D(s) \} \) holds. This contradicts \( \Theta(f) < \max \{ \Theta(\mathcal{N}^e) : e \in g_D(s) \} \).

By Theorem 3.4, when \( b^e (e \in g_D(s)) \) are fixed, we can compute \( \Theta(\mathcal{N}) \) by computing \( \Theta(\mathcal{N}^e) \) for all \( e \in g_D(s) \). Furthermore, by Lemma 3.5, it is sufficient to compute \( \Theta(\mathcal{P}^e) (e \in g_D(s)) \). For every \( e \in g_D(s) \), the condensed time-expanded network \( \mathcal{P}^e \) for \( \mathcal{P}^e \) becomes as illustrated in Figure 7(a). However, since only arcs \( p_i^2p_i^1 (i \in \{2, \ldots, k\}) \) have finite capacities, it is not difficult to see that we can shrink vertices \( p_i^j (j \in \{2, \ldots, i\}) \) into a single vertex for each \( i \in \{2, \ldots, k\} \) without changing the feasibility of the network. We call the resulting network a gadget and denote by \( \mathcal{G}^e \). For convenience, we denote by \( s_i^j \) the vertex obtained by shrinking vertices \( p_i^j \) for each \( e \in g_D(s) \). Furthermore, for each \( e \in g_D(s) \) and \( i \in \{1, \ldots, k\} \), we denote by \( s_i^j \) the vertex \( p_i^j \) of the gadget \( \mathcal{G}^e \).

Figure 7: (a) The condensed time-expanded network for \( \mathcal{P}^e \) in Figure 6. (b) Gadget \( \mathcal{G}^e \) for \( \mathcal{P}^e \).

### 3.4 Reduction to the restricted parametric flow problem

We are ready to explain our algorithm. In the evacuation problem in \( \mathcal{N} \), \( b^e (e \in g_D(s)) \) are not fixed. Here we explain how to determine \( b^e (e \in g_D(s)) \) which attains the minimum value of

\[
\max \{ \Theta(\mathcal{N}^e) : e \in g_D(s) \}.
\]

To obtain optimal \( b^e (e \in g_D(s)) \), we define the dynamic network \( \mathcal{N}^o \) which is the union of \( \mathcal{P}^e (e \in g_D(s)) \) where the original supply of every vertex is removed, instead a new vertex \( v^o \) whose supply is \( b(v) \) is added for each \( v \in V \) and arcs from each \( v^o \) to \( p^{lev}(v) \) in \( \mathcal{P}^e \) for each \( e \in g_D(s) \) with \( v \in V^e \) are added where capacities and transit times of these arcs are infinity and zero, respectively (see Figure 8).

By Theorem 3.4 and Lemma 3.5, the evacuation problem in \( \mathcal{N} \) and that in \( \mathcal{N}^o \) are equivalent. For each \( v \in V \), the amount of flow through an arc connected from \( v^o \) to \( p^{lev}(v) \) in \( \mathcal{P}^e \) determines
Figure 8: The dynamic network $N^o$ for $N$ in Figure 6.

$b^e(v)$. In order to compute $\Theta(N^o)$, we will use a static network $N_r$ which is composed of $G^e$ ($e \in QD(s)$) as well as source vertices and their incident arcs. Notice that the supplies of vertices in $G^e$ are set to zero.

**Theorem 3.6.** The evacuation problem in $N$ can be solved by solving the restricted parametric flow problem.

**Proof.** Let $N_r = (D_r = (V_r, A_r), c_r, b_r, S_r)$ be the static network to which the evacuation problem in $N$ is reduced. We construct $N_r$ in the following four steps (see Figure 9).

1. Construct $G^e$ ($e \in QD(s)$) with no supply such that the parameter $T$ is common.
2. For each $v \in V$, add a vertex $a_v$ to $V_r$, and set the supply of $a_v$ to $b(v)$.
3. For each $e \in QD(s)$ with $v \in V^e$, add an arc $a_vg^e_{lev(v)}$ with infinite capacity.
4. Let $S_r$ be the set of the sinks of all gadgets.

In Step 2, we add new vertices to the set of gadgets to allocate the supplies. Notice that $a_v$ corresponds to the source vertex $v^o$ in $N^o$. Let us consider to which vertices of gadgets we need to connect $a_v$. Recall that the supply of $v$ in $N$ is given to $p^{lev(v)}$ in $P^e$, and for each $e \in QD(s)$ $g^e_i$ is obtained by contracting the vertices in the $i$-th component in $P^e$. Thus, $a_v$ should be connected to $g^e_{lev(v)}$ for each $e \in QD(s)$ with $v \in V^e$.

Let $T^*$ be the minimum value of $T$ such that there exists a feasible flow in $N_r$. Even if $T$ is not an integer, $\lceil T^* \rceil$ is equal to $\Theta(N)$ (see Section 2.5.3). Notice that from the way of construction of $N_r$, the parameter $T$ is contained only in the capacity of the arc which is incident to $s_k$ in each gadget $G^e$. Thus, all arcs of $A_r$ whose capacity contains the parameter $T$ are incident to the sinks of $S_r$. This completes the proof.

![Figure 9: (a) Gadgets $G^e$ and $G^f$. (b) Vertices and arcs introduced to allocate the supplies.](image)

### 3.5 Time complexity

In this subsection, we evaluate the time complexity of our algorithm. We can reduce the size of $N_r$ without changing the feasibility by performing the following gadget compression and source compression. In the subsequent discussion, let $N$ be the set of vertices $v \in V$ for which there
exists \( e \in g_D(s) \) with \( t(e) = v \).

**Gadget compression.** Since we construct one gadget for each \( e \in g_D(s) \), the number of gadgets in \( \mathcal{N}_r \) is equal to \( |g_D(s)| \). However, we can reduce the number of gadgets to \( |N| \). Consider a subset \( B \) of \( g_D(s) \) for which \( t(e) (e \in B) \) are identical. Since \( t(e) (e \in B) \) are identical, \( V^e (e \in B) \) are identical. Thus, vertices in \( \{a_v : v \in V\} \) which are connected to \( G^e \) are identical for every \( e \in B \). Hence we can combine \( G^e (e \in B) \) into a single gadget by magnifying the capacities of arcs \(|B| \) times.

**Source compression.** We add a new vertex \( a_v \) for each \( v \in V \). However, consider the case where there exist \( u, v \in V \) with \( \text{lev}(u) = \text{lev}(v) \) and \( R_u = R_v \). Then, \( a_u \) and \( a_v \) are connected to the same vertices \( g_i^e \) for each \( i = \text{lev}(u) = \text{lev}(v) \) and \( e \in g_D(s) \) with \( u, v \in V^e \). Furthermore, arcs which connects \( a_u \) and \( a_v \) to gadgets have infinite capacities. Thus, it is sufficient to add only one vertex whose supply is equal to \( b(u) + b(v) \). That is, letting \( V(i, Q) = \{ v \in V : \text{lev}(v) = i \) and \( R_v = Q \}, \) we only prepare one vertex \( a_{i,Q} \) for each nonempty \( V(i, Q) \).

Let \( \mathcal{N}_r \) be the network obtained from \( \mathcal{N}_r \) by applying gadget compression and source compression. Moreover, let \( D_r = (V_r, A_r) \) be the underlying graph of \( \mathcal{N}_r \). Let us analyze the size of \( \mathcal{N}_r \). Let \( \eta \) be the number of distinct combinations of the path-length from a vertex \( v \in V \) to \( s \) and \( R_v \), i.e., \( \eta = |\{(l_v, R_v) : v \in V\}| \). Notice that \( \eta \) is equal to the number of vertices added to allocate the supplies in \( \mathcal{N}_r \), i.e., all \( a_{i,Q} \) defined above, and \( \eta = O(n) \) and \( \eta \geq k \) hold. Thus, we have \(|V_r| = O(k|N| + \eta) \) and \(|A_r| = O(\eta|N|) \).

Next we consider the time required to construct \( \mathcal{N}_r \).

**Lemma 3.7.** We can construct \( \mathcal{N}_r \) in \( O(\min\{m + n^2|N|, m|N|\} + n \log n) \) time.

**Proof.** We construct \( \mathcal{N}_r \) as follows. We first compute \( l_v \) for all \( v \in V \) in \( O(m) \) time by breadth-first search from \( s \) since \( \mathcal{N} \) satisfies the uniform path-length condition. After this, we can compute \( \text{lev}(v) \) for all \( v \in V \) in \( O(n \log n) \) time by sorting \( \{l_v : v \in V\} \). Furthermore, we can construct all gadgets in \( O(|g_D(s)| + k|N|) \) time.

To add the vertices and arcs to allocate the supplies, we compute \( V(i, Q) \). First we obtain \( R_v \) for all \( v \in V \) by depth-first search from all \( u \in N \) in \( O(\min\{m + n^2|N|, m|N|\}) \) time. Then, we partition \( V \) according to \( \text{lev}(v) \) in \( O(n) \) time to obtain the set of vertices \( v \) whose \( \text{lev}(v) \) takes the same value. Next we assign the values \( 2^1, 2^2, \ldots, 2^{\lceil |N| \rceil} \) to each \( u \in N \). Then, for each set \( W \) of vertices \( v \) whose \( \text{lev}(v) \) takes the same value, we compute the sum of the values of \( u \in R_v \) for each \( v \in W \), and sort the vertices \( v \in W \) according to the sum of the assigned values of \( u \in R_v \). Notice that for \( v, w \in V \) with \( R_v \neq R_w \), the sum of the assigned values of the vertices in \( R_v \) can never be equal to that of the vertices in \( R_w \). The time required to complete this operation for all levels \( i \in \{1, \ldots, k\} \) is \( O(n|N| + n \log n) \). Thus, this step can be completed in \( O(\min\{m + n^2|N|, m|N|\} + n \log n) \) time, which completes the proof.

The following theorem holds from the above two lemmas and Theorem 3.2.

**Theorem 3.8.** Given a fully connected dynamic network \( \mathcal{N} \) with uniform path-lengths in which the capacity of every arc is equal to one, \( \Theta(\mathcal{N}) \) can be computed in \( O(\eta|N|(k|N| + \eta) \log n + \min\{m + n^2|N|, m|N|\} + n \log n) \) time.

Let us express the running time given in the above theorem in terms of \( n \) and \( m \). Recall that the number of arcs added to allocate the supplies is \( O(\eta|N|) \). Thus, this number is \( O(n^2) \) by \( \eta = O(n) \) and \( |N| = O(n) \). However, we show that the number of the arcs added to allocate the supplies is \( O(m) \). This number is at most \( \sum_{v \in V \setminus s} |R_v| \) since \( a_{i,R_v} \) is connected to at most \( |R_v| \) gadgets. By \( R_v \subseteq N \) for each \( v \in V \setminus s \), it is clear that

\[
\sum_{v \in V \setminus s} |R_v| \leq \sum_{v \in V \setminus s} \{|e \in g_D(s) : t(e) \in R_v\}.
\]
Furthermore, \(|\delta_D(v)| \leq |\{e \in g_D(s) : t(e) \in R_v\}|\) for each \(v \in V - s\) since \(N\) is fully connected and the capacity of any arc is one. Thus, we have \(\sum_{v \in V - s} |R_v| \leq |\{e \in g_D(s) : t(e) \in R_v\}| = m\). Since the number of arcs in all gadgets is equal to \(O(k|N|)\), \(|A_v| = O(\min\{n^2, kn + m\})\) holds. Thus, the following corollary follows from Lemma 3.2.

**Corollary 3.9.** Given a fully connected dynamic network \(N\) with uniform path-lengths in which the capacity of every arc is equal to one, \(\Theta(N)\) can be computed in \(O(\min\{m + kn^3 \log n, (k^2n^2 + kmn) \log n\})\) time.

If we simply apply the algorithm of [6] to solve the evacuation problem in a fully connected network \(N\) with uniform path-lengths, the time complexity is \(O(k^3m^2n \log (kn^2/m))\). Our algorithm much improves the time bound in the fully connected and uniform path-lengths case. In many practical cases, the in-degree of the sink (i.e., \(|N|\)) can be considered as a constant, and thus the time complexity of our algorithm becomes \(O(m + n^2 \log n)\) in this case by Theorem 3.8.

### 3.6 Computing an optimal flow

The algorithm described in Section 3.4 only computes \(\Theta(N)\), but not an optimal dynamic flow. If we explicitly obtain \(N^e (e \in g_D(s))\), an optimal flow in \(N\) can be computed by separately computing \(\Theta(N^e) (e \in g_D(s))\) by using the algorithm of [13] which can compute an optimal flow in a dynamic tree network. However, the time required to compute \(N^e (e \in g_D(s))\) is very expensive (see Section 3.2). Thus, in this subsection, we present an algorithm to compute an optimal flow without computing \(N^e (e \in g_D(s))\).

Our algorithm presented in Section 3.4 relies on the restricted parametric flow algorithm for \(N^c\). Recall that a gadget \(G^c\) is a static network with no supply which is obtained by shrinking certain vertex sets of \(P^c\). Thus, it is a nontrivial task to obtain an optimal dynamic flow even after we obtain a feasible flow in \(N^c\) with \(T = \Theta(N)\). Notice that if we can find a feasible flow in the condensed time-expanded network \(N^c\) with \(T = \Theta(N)\), we can easily construct a feasible dynamic flow by Theorem 2.1. Thus, since we already know \(\Theta(N)\), we can find an optimal dynamic flow by a single computation of the maximum flow in \(N^c\). This requires \(O(k^2mn \log n)\) time by using the maximum flow algorithm of [5] by \(|V_c| = O(kn)|\) and \(|A_c| = O(km)|\), respectively. However, we want to do it faster. In this subsection, we show that an optimal flow can be computed in \(O(kmn \log n)\) time by using the information of a feasible flow obtained in \(N^c\) with \(T = \Theta(N)\). In our algorithm, we sequentially compute a feasible flow in the \(i\)-th component of \(N^c\) from \(i = 1\) to \(k\) instead of computing a feasible flow for the whole \(N^c\) at one time. In this section, we fix the value of \(T\) to \(\Theta(N)\).

#### 3.6.1 Recovering a feasible flow in \(N^c\) from a feasible flow in \(N^c\)

We first construct a feasible flow \(f^{c}\) in \(N^c\) from a feasible flow \(f^c\) in \(N^c\). Recall that \(N^c\) is obtained from \(N^c\) by applying gadget compression and then source compression (see Section 3.5). Let \(N^{c^\prime}\) be the network obtained from \(N^c\) by perforing gadget compression. For each \(v \in N\), let \(G^c_{v}\) be the gadget in \(N^c\). We first compute a feasible flow \(f^{c^\prime}\) in \(N^{c^\prime}\) from a feasible flow \(f^c\) in \(N^c\). After this, we compute a feasible flow \(f^{c^\prime}\) in \(N^c\) from \(f^{c^\prime}\).

**From \(f^c\) to \(f^{c^\prime}\).** Let us consider a vertex \(a_{i,Q}\) in \(N^c\), and suppose that \(a_{i,Q}\) is connected to gadgets \(G^c_{v} (v \in N')\) with \(N' \subseteq N\). Recall that \(a_{i,Q}\) was obtained by combining vertices of \(V(i, Q)\) into a single one. Thus, for each \(v \in V(i, Q)\) we will determine the flow value of an arc from \(a_{v}\) to \(G^c_{w} (w \in N')\). This is simply done by arbitrarily splitting the values of \(f^c\) for arcs from \(a_{i,Q}\) to \(G^c_{v} (v \in N')\) so that the sum of the flow values of arcs from \(a_{v}\) to \(G^c_{v} (v \in N')\)
does not exceed $b(v)$ (see Figure 10). Notice that even if we split the values of $f_r$ for arcs from $a_tQ$, the amount of flow entering into each gadget does not change. Thus, flow conservation is satisfied at vertices in $G_r^e$ ($v \in N^e$). This step is done in $O(n^2)$ time.

**From $f_r$ to $f_e$.** Let us consider a gadget $G_r^e$ in $N_r$, and suppose that $N_r$ is obtained by combining gadgets $G^e$ ($e \in B$) with $B \subseteq gD(s)$. The values of $f_r$ for arcs in $G^e$ ($e \in B$) can be obtained by simply equally dividing the value of $f_r$ by $|B|$. Similarly, for each $v \in V$, the values of $f_r$ for arcs connected from $a_v$ to vertices in $G^e$ ($e \in V$) can be also simply computed by equally dividing the value of $f_r$ by $|B|$. This step is done in $O(nm)$ time. Thus, the time required to compute $f_r$ from $f_e$ does not affect total time of the algorithm.

**Figure 10:** (a) Feasible flow $f_e$ in $N_e$ for $N$ in Figure 6(a). The number attached to an arc indicates the value of flow. (b) Feasible flow $f_r$ in $N_r$.

### 3.6.2 Computing a feasible flow in $N_e$

In this section, we explain how to find a feasible dynamic flow in $N_e$. For this, we introduce necessary definitions. Recall that for each $v \in V$, only $v_{\text{lev}(v)}$ has the positive supply among $v_i$ ($i \in \{\text{lev}(v), \ldots, k\}$) in $N_e$. Let us consider a feasible flow $f_c$ in $N_c$, and suppose that we can distinguish the supplies originating from each $v_{\text{lev}(v)}$ in $f_c$. When the supplies originating from $v_{\text{lev}(v)}$ does not pass through holdover arcs other than $v_iv_i+1$ ($i \in \{\text{lev}(v), \ldots, k - 1\}$) for every $v \in V$, $f_c$ is called *simple*. The algorithm explained below finds a simple feasible flow in $N_e$.

For a feasible flow $f_c$ in $N_c$ and each $v \in V$, if a certain amount of the supply of $v_{\text{lev}(v)}$ flows into arcs of the $i$-th component, this amount is called the supply of $v$ consumed in the $i$-th component and is denoted by $\sigma_i(v)$. Here we show that if we know $\sigma_i(v)$ ($i \in \{1, \ldots, k\}$ and $v \in V$), we can restore a simple feasible flow $f_c$. Notice that $\sigma_i(v) = 0$ for each $v \in V$ and $i \in \{1, \ldots, \text{lev}(v) - 1\}$, and $b(v) = \sum_{i=1}^{k} \sigma_i(v)$ hold for each $v \in V$. Since $f_c$ is a simple feasible flow, $f_c(v_i,v_i+1)$ is $b(v) - \sum_{i=1}^{k} \sigma_i(v)$ for each $v \in V$ and $i \in \{1, \ldots, k - 1\}$. Then the flow value on arcs in the $i$-th component can be found by simply obtaining a feasible flow by setting the supply value at $v_i \in V$ to $\sigma_i(v)$ in the $i$-th component which has a single sink $s_i$. Namely, we can restore $f_c$ by computing maximum flow problems $k$ times for the network with $O(n)$ vertices and $O(m)$ arcs once we know $\sigma_i(v)$ ($i \in \{1, \ldots, k\}$ and $v \in V$).

If $\sigma_i: V \rightarrow \mathbb{R}_+ (i \in \{1, \ldots, k\})$ represent the supply amount consumed in the $i$-components for some simple feasible flow in $N_c$, they are called *eligible*. Thus, our aim is to obtain $\sigma_i$ ($i \in \{1, \ldots, k\}$) which are eligible. Recall that for each $e \in gD(s)$ and $v \in V^c$, $f_r(a_v g^e_{\text{lev}(v)})$ is equal to $b^e(v)$, i.e., the allocation of the supply of $v$ to $N^e$. For each $e \in gD(s)$, let $N^e_c$ be the condensed time-expanded network of $N^e_c$ with the supply of $v \in V^e$ set to $b^e(v)$. For each $e \in gD(s)$, we first obtain $\sigma^e_i: V \rightarrow \mathbb{R}_+$ ($i \in \{1, \ldots, k\}$) which is eligible in $N^e_c$ where $\sigma^e_i$ is defined in a manner similar to $\sigma_i$. Since $A^e (e \in gD(s))$ are disjoint, the union of simple feasible flows of $N^e_c$ ($e \in gD(s)$) becomes a simple feasible flow of $N_c$. Thus, we can obtain an eligible $\sigma_i$ ($i \in \{1, \ldots, k\}$) by setting $\sigma_i(v) = \sum\sigma^e_i(v): e \in gD(s)$ for each $v \in V$. Hence, we will focus
on how to compute \( \sigma^e_i \) \((i \in \{1, \ldots, k\})\) which is eligible in \( \mathcal{N}^e \) for each \( e \in \mathcal{G}(s) \). The algorithm is described in Procedure 1. We will prove its correctness. Notice that in the procedure we do not need to obtain \( \mathcal{N}^e \), but we only need to know \( V^e \).

**Procedure 1**

1. For each \( i \in \{1, \ldots, k\} \) and \( v \in V \), set \( \sigma^e_i(v) = 0 \).
2. For each \( e \in \mathcal{G}(s) \) and \( v \in V^e \), set \( \sigma^e_{\text{lev}(v)}(v) = f_r(a_v g^e_{\text{lev}(v)}). \)
3. for each \( i \in \{1, \ldots, k-1\} \) and \( e \in \mathcal{G}(s) \)
4. if \( \sum_{v \in V} \sigma^e_i(v) > f_r(g^e_i s^e_i) \) then
5. \( \hat{\sigma}^e_i : V \rightarrow \mathbb{R}_+ \) so that \( \hat{\sigma}^e_i(V) = f_r(g^e_i s^e_i) \) with \( 0 \leq \hat{\sigma}^e_i(v) \leq \sigma^e_i(v) \).
6. for each \( v \in V \) with \( \hat{\sigma}^e_i(v) < \sigma^e_i(v) \) do
7. \( \sigma^e_i(v) = \sigma^e_i(v) - \hat{\sigma}^e_i(v) \), and then set \( \sigma^e_i(v) = \hat{\sigma}^e_i(v) \).
8. end for
9. end if
10. end for

Let us fix \( e \in \mathcal{G}(s) \), and we prove that \( \sigma^e_i \) \((i \in \{1, \ldots, k\})\) by Procedure 1 are eligible in \( \mathcal{N}^e \). For this, it is sufficient to prove the following statements by the definition of eligibility.

1. For every \( v \in V \setminus V^e \) and \( i \in \{1, \ldots, k\} \), \( \sigma^e_i(v) = 0 \) holds.
2. For every \( v \in V^e \) and \( i \in \{1, \ldots, \text{lev}(v) - 1\} \), \( \sigma^e_i(v) = 0 \) holds.
3. For each \( i \in \{1, \ldots, k\} \), there exists a feasible flow in the \( i \)-th component with setting the supply of \( v_i \) to \( \sigma^e_i(v) \).

Conditions 1 and 2 clearly hold. Therefore, we prove that Statement 3 holds.

In order that there exists a feasible flow in \( \mathcal{N}^e \), it is sufficient to show \( \sigma^e_i(V^e) \leq |I_i| \) since the \( i \)-th component of \( \mathcal{N}^e \) is an in-tree rooted at \( s_i \) and every arc capacity is equal to \( |I_i| \). However, this obviously holds since the procedure at lines 7 carries over the excess of \( \sigma^e_i(V^e) - f_r(g^e_i s^e_i) \) to the \((i + 1)\)-st component which is equal to \( f_r(g^e_{i+1} s^e_{i+1}) \) and \( f_r(g^e_i s^e_i) \leq c_r(g^e_i s^e_i) = |I_i| \) holds by the capacity constraint in \( \mathcal{N} \). Since the time complexity of Procedure 1 is clearly \( O(kmn) \), we obtain the following theorem. Recall that once we obtain eligible \( \sigma_i \) \((i \in \{1, \ldots, k\})\), we can restore a feasible flow in \( O(kmn \log n) \) time.

**Theorem 3.10.** Given a fully connected dynamic network \( \mathcal{N} \) with uniform path-lengths, an optimal flow for the evacuation problem in \( \mathcal{N} \) can be computed in the time required to compute \( \Theta(\mathcal{N}) \) plus \( O(kmn \log n) \) time.

### 3.7 Integral capacity case

In the case where the capacities of arcs are integral, we can prove by splitting each arc with more than unit capacity into parallel ones whose capacity is one that our algorithm can be applied to this case. However, in this case, the number of arcs added to \( \mathcal{N} \) for allocation of supplies is not \( O(m) \). That is, in this case we can not say that the time complexity required to compute \( \Theta(\mathcal{N}) \) is \( O(k^2n^2 + kmn \log n) \) unlike the unit capacity case. Notice that since we assume that there exist no parallel arcs without loss of generality, the time required to compute an optimal flow is dominated by that required to compute the optimal value.

**Theorem 3.11.** Given a fully connected dynamic network \( \mathcal{N} \) with uniform path-lengths such that a capacity of every arc is integral, we can compute \( \Theta(\mathcal{N}) \) in the same complexity of Theorem 3.8 or \( O(m + kn^3 \log n) \). Moreover, an optimal flow can be computed in \( O(m + kn^3 \log n) \) time.
4 Evacuation Problem in Grid Networks with Unit Capacity

In the paper [11], we considered the evacuation problem in a $\sqrt{n} \times \sqrt{n}$ grid network with uniform arc capacity and uniform transit time such that the arc is directed so that the condition of uniform path-lengths holds. In the previous section, we generalized the class of networks which the ideas developed in [11] can be applied to. We again consider in this section the evacuation problem in grid networks with unit capacity.

First we define an $r$-dimensional grid graph $D = (V, A)$ (see Figure 11). We assume any $v \in V$ is on $(2M+1)^r$ grid points $\{-M,\ldots,0,\ldots,M\}^r$ in $\mathbb{R}^r$, and a vertex is identified with $(x_1,x_2,\ldots,x_r)$ with $-M \leq x_i \leq M$ for each $i \in \{1,2,\ldots,r\}$. Notice that $M = O(n^{1/r})$ holds. Moreover, we assume the sink is the center of the grid graph (the other case can be similarly treated), i.e., we define a vertex $(0,0,\ldots,0)$ as the sink $s$. The distance between two vertices $(x_1,x_2,\ldots,x_r)$ and $(y_1,y_2,\ldots,y_r)$ is defined as $\sum_{i=1}^{r} |x_i-y_i|$. Two vertices $v$ and $w$ are connected by an arc if and only if the distance between $v$ and $w$ is equal to one. The arc which connects $v$ and $w$ is directed from $v$ to $w$ if and only if the distance from $w$ to $s$ is smaller than that from $v$ to $s$ by one. A dynamic network defined on an $r$-dimensional grid graph such that transit time of any arc takes the same value is called an $r$-dimensional grid network.

![2-dimensional grid graph](image)

Figure 11: 2-dimensional grid graph.

Here we consider the evacuation problem in an $r$-dimensional grid network $\mathcal{N}$ in which the capacity of any arc is one. For every $v = (x_1,x_2,\ldots,x_r) \in V$, the length of any directed path from $v$ to $s$ is equal to $\sum_{i=1}^{r} |x_i|$ regardless of the choice of the path. Furthermore, an $r$-dimensional grid network is clearly fully connected. Thus, by Theorem 3.8, what we need to evaluate is $|\mathcal{N}|$, $k$ and $\eta$.

First we consider $|\mathcal{N}|$. By $s = (0,0,\ldots,0)$, the vertex whose distance from $s$ is equal to one satisfies the condition such that the only one element in the coordinates is equal to 1 or $-1$. Therefore, $|\mathcal{N}|$ is equal to $2r$, i.e., $O(r)$. Next we consider $k$. The furthest vertex from $s$ has the distance $rM$. Furthermore, there exists a vertex whose distance from $s$ is equal to $rM - i$ for every $i \in \{0,1,\ldots,rM-1\}$. Therefore, $k = O(rM)$ holds. Finally, we consider $\eta$. The trivial bound of $\eta$ is $O(n)$ since the number of vertices is $n$. Since the in-degree of $s$ is $2r$, the number of the distinct $R_v$ is $O(4^r)$ for all $v \neq s$. Thus, $\eta = O(\min\{n, r4^rM\})$ holds by $k = O(rM)$.

Furthermore, $m = O(rn)$ follows from the definition of an $r$-dimensional grid graph. Thus, by Theorem 3.8, the time required to compute $\Theta(\mathcal{N})$ is

$$O(r^2n + n \log n) + \min\{O(r^34^{2r}M^2 \log n), O((r^3Mn + rn^2) \log n)\}.$$

By $O(M) = n^{1/r}$, we have the following theorem.

**Theorem 4.1.** We can compute $\Theta(\mathcal{N})$ for an $r$-dimensional grid network $\mathcal{N}$ with unit capacity in the following time complexity.

$$O(r^2n + n \log n) + \min\{O(r^34^{2r}n^{2/r} \log n), O((r^3n^{1+(1/r)} + rn^2) \log n)\}.$$
If $r = 2$, the time complexity is $O(n \log n)$. This matches the result of [11] in the case of 2-dimensional grid network.

5 Discussion

Here we consider the algorithm for the evacuation problem in the case where the underlying graph is not acyclic. As mentioned before, the length of a directed cycle is zero in a dynamic network with uniform path-lengths. Thus, the existence of directed cycles with zero length is practically meaningless, but it may be of theoretical interest to consider the case where there exist directed cycles from the theoretical viewpoint. In fact, we use the fact that the underlying graph of the input dynamic network is acyclic only to prove Lemma 3.3, and it is clear that the other discussions hold without acyclicity assumption. Recently, Kamiyama et al [10] proved that Lemma 3.3 also holds in the case where the underlying graph is allowed to have cycles. Thus, we can apply the algorithm presented in this paper to dynamic networks with cycles.

As a related problem to the evacuation problem, there exists the universally quickest flow problem defined as follows. Given a dynamic network, the problem asks for finding a dynamic flow which attains the minimum evacuation time and also maximizes the amount of the supplies that has reached the sink at every time step. We have recently proved that this problem can be efficiently solved in a fully connected network with uniform path-lengths. However, the proof needs several non-trivial discussions. Thus, we will present this result elsewhere.

Acknowledgement. We appreciate the comments of anonymous reviewers who helped improve the presentation of this paper. The first author is supported by JSPS Research Fellowships for Young Scientists. The second and third authors are supported by the project New Horizons in Computing, Grant-in-Aid for Scientific Research on Priority Areas, MEXT Japan.

References


