Improved max-flow min-cut algorithms in a Circular Disk Failure Model with application to a road network

Kensuke Otsuki, Yusuke Kobayashi, Kazuo Murota

A R T I C L E   I N F O
Article history:
Received 25 March 2015
Accepted 13 July 2015
Available online 20 July 2015
Keywords:
Combinatorial optimization
Networks
Reliability
Graph theory

A B S T R A C T
In the evaluation of network reliability, the objectives are to model reliability of networks appropriately and to compute it in a realistic time. We can consider various models of reliability of networks, and Bienstock (1991) first introduced a geographical failure model where each failure is represented by a 2-dimensional region. In this model, we consider the situation that geographical networks such as road networks may be damaged by externally caused disasters, and such disasters may destroy several links of the networks simultaneously, rather than each link independently. Recently, Neumayer–Efrat–Modiano (2012) investigated the max-flow problem and the min-cut problem under the Circular Disk Failure Model, in which the shape of each failure is restricted to be a disk. Under this model, Kobayashi–Otsuki (2014) gave polynomial time algorithms to find optimal solutions of these two problems. In this paper, we improve the algorithms and evaluate their performance by computational experiments. Although our improvements work only when the max-flow value is equal to the min-cut value, this condition holds in almost all practical cases. Owing to the improvements, we can find in a realistic time optimal solutions of the max-flow problem and the min-cut problem in large networks under the Circular Disk Failure Model. As a realistic instance, we analyze reliability of a road network in New York consisting of 264,346 nodes.

1. Introduction

Recently, with the development of various networks such as road networks and the Internet, there is urgent need to evaluate reliability of large networks. In the evaluation of reliability, the crucial points are how to model reliability of networks appropriately and how to compute it in a realistic time. We can conceive various ways to formulate reliability of networks (see e.g. Habib, Tornatore, Dikbijl, & Mukherjee, 2013), but a complicated formulation of reliability would make it hard to design an efficient algorithm for computing reliability.

In most previous studies, connectivity of networks is regarded as one of the most significant factors of network reliability, where connectivity is defined as the minimum number of link-failures (or node-failures) that disconnect the network. Connectivity is a simple and natural model for reliability of networks, and an advantage of using connectivity is that we have efficient algorithms for computing it (see Schrijver, 2003). In particular, there are many studies on the problem of finding a minimum number of link-failures that disconnect two specified nodes s and t (which we call the min-cut problem) and its dual problem called the max-flow problem. It is known as the max-flow min-cut theorem (Ford & Fulkerson, 1956; Menger, 1927) that the optimal values of the min-cut problem and the max-flow problem are equal, and the proof of this theorem leads to algorithms for these two problems. Since then, many faster algorithms for the max-flow problem and the min-cut problem have been proposed (see Schrijver, 2003), and currently we can solve these problems for huge networks in a realistic time.

In this paper, we consider variants of the max-flow problem and the min-cut problem, motivated by modeling geographical failures on geographical networks such as road networks. Geographical networks may be damaged by externally caused disasters like earthquakes or floods. Such disasters, occurring in specific geographical regions, may often destroy several links of the networks simultaneously, rather than each link independently. In 1991, Bienstock (1991) first introduced a geographical failure model where each failure is represented by a 2-dimensional region. Recently, various models which takes such region failures into consideration have been drawing a great deal of attention.
Let a simpler min-cut algorithm was also given in Kobayashi and Otsuki (2014) gave a polynomial time min-cut algorithm and Kobayashi and Otsuki failure intersects a pair of these paths. whose centers are not contained in the protective disks of radius regarded as a special case of Bienstock's model, in which the shape of the max-flow problem and the min-cut problem, where each failure Neumayer & Modiano, 2010; Neumayer, Zussman, Cohen, & Modiano, 2011; Saito, 2015; Sen, Murthy, & Banerjee, 2009; Trajanovski, Kuipers, & Mieghem, 2013). In particular, Neumayer, Efrat, and Modiano (2012) adopted the Circular Disk Failure Model for the max-flow problem and the min-cut problem, where each failure is represented by a disk with a predetermined size. This model can be regarded as a special case of Bienstock's model, in which the shape of each failure is restricted to be a disk. Under that model, the min-cut problem is to find a minimum number of disk failures that disconnect two specified nodes $s$ and $t$ of the given planar graph $G$, and the max-flow problem is to find a maximum number of $s$-$t$ paths such that no disk failure intersects a pair of these paths.

Under the Circular Disk Failure Model, Neumayer et al. (2012) gave a polynomial time min-cut algorithm and Kobayashi and Otsuki (2014) gave a polynomial time max-flow algorithm. A faster and simpler min-cut algorithm was also given in Kobayashi and Otsuki (2014). However, as we can see in 9th DIMACS Implementation Challenge — Shortest Paths, many of real road networks consist of more than 100,000 nodes, which are too huge for the aforementioned algorithms to solve in a realistic time.

In this paper, we improve the max-flow algorithm and the min-cut algorithm given in Kobayashi and Otsuki (2014), and evaluate the performance of these algorithms by computational experiments. Although our improvements work only when the max-flow value is equal to the min-cut value, this condition holds in almost all practical cases (see Section 2 for details). Owing to the improvements, now we can find in a realistic time optimal solutions of the max-flow problem and the min-cut problem under the Circular Disk Failure Model in huge networks consisting of more than 100,000 nodes. As a practical instance, we analyze reliability of a road network in NewYork (9th DIMACS Implementation Challenge — Shortest Paths) with 264,346 nodes. We can solve the problems in the network in about 30 seconds on a computer with Intel Core i7, 2.8 GigaHertz and 8 Gigabyte of memory, while it takes about 3 hours by the existing algorithms.

The remainder of this paper is organized as follows. In Section 2, we give formal descriptions of the problems with some mathematical definitions. In Sections 3 and 4, we describe our improvements of the max-flow algorithm and the min-cut algorithm, respectively. In Section 5, we evaluate the performance of our algorithms and apply our algorithms to the road network in NewYork. Finally, in Section 6, we conclude this paper.

2. Problem settings

Let $G = (V, E)$ be a graph drawn in the plane with a node set $V$, a link set $E$, and two distinct nodes $s, t \in V$. Throughout this paper we assume that each link is drawn as a line segment, and a link is sometimes called an edge. Each region surrounded by links is called a face of $G$, where we note that the outer infinite region is also a face. Let $r_B$ be a hole radius, where each hole represents a failure of the network. We consider two disks of another radius $r_P$ whose centers are $s$ and $t$, respectively. We call them protective disks and their radius $r_P$ protective radius. Define $H(r_B, r_P)$ as the set of all disks of radius $r_B$ whose centers are not contained in the protective disks of radius $r_P$. We call each element of $H(r_B, r_P)$ a hole in this paper. We shall think of each hole as a closed set, including its boundary.

The geometrical min-cut problem and the geometrical max-flow problem introduced by Neumayer et al. (2012) are defined as below. Solutions of the problems represent connectivity of the graph between two nodes $s$ and $t$ under the Circular Disk Failure Model.

Geographical Min-Cut by Circular Disasters (GMCCD)

**Input:** a graph $G = (V, E)$ drawn in the plane, two distinct nodes $s$ and $t$, a hole radius $r_B$, and a protection radius $r_P$.

Find: a minimum cardinality set of holes in $H(r_B, r_P)$ that disconnect $s$ from $t$.

Geographical Max-Flow by Circular Disasters (GMFCD)

**Input:** a graph $G = (V, E)$ drawn in the plane, two distinct nodes $s$ and $t$, a hole radius $r_B$, and a protection radius $r_P$.

Find: a maximum cardinality set of $s$-$t$ paths such that no hole in $H(r_B, r_P)$ intersects a pair of these paths.

It is pointed out in Kobayashi and Otsuki (2014) that most results for these problems can be extended to the case where $H(r_B, r_P)$ is replaced with a set $H$ of convex shapes (holes) satisfying a certain property (see Concluding Remarks of Kobayashi and Otsuki (2014)). For example, $H$ can be a set of disks of different sizes or a set of axis parallel squares. All the results in this paper can be extended to such cases, but, for simplicity, we focus on the case when each failure is in $H(r_B, r_P)$.

In what follows, we say that two $s$-$t$ paths are hole-disjoint if there is no hole in $H(r_B, r_P)$ that intersects both of them. Let MIN-CUT and MAX-FLOW denote the optimal values of the GMCCD and the GMFCD, respectively, and we call feasible solutions of the GMCCD and the GMFCD a hole cut and a hole-disjoint flow, respectively. The reason we introduce protective disks is that the GMCCD and the GMFCD will have trivial optimal solutions if a hole contains $s$ or $t$. In terms of application, we can assume that $s$ and $t$ are well-protected so that no failure can occur nearby nodes $s$ and $t$. Although it is impossible to protect a certain region completely in practical situations, we believe that the above problems are useful to evaluate the network reliability.

**Example 1.** Consider the graph as in Fig. 1. Small circles represent holes of radius $r_B$ and two large circles represent protective disks of radius $r_P$. In the graph, we can easily see that MAX-FLOW $= 1$ and MIN-CUT $= 2$. The thick $s$-$t$ path represents an optimal solution of the GMFCD, and two small circles represent an optimal solution of the GMCCD.

This example is given in Neumayer et al. (2012) as an example to show that MIN-CUT $= MAX-FLOW$, which is an analogy to Menger's theorem, does not necessarily hold. Practically, however, MIN-CUT $= MAX-FLOW$ holds in most cases. Indeed, it is claimed in Neumayer et al. (2012) that among 1000 randomly generated instances with 13 nodes, MIN-CUT $= MAX-FLOW$ holds for 99 percent of the instances and MIN-CUT $= MAX-FLOW + 1$ holds for the remaining 1 percent. Later, it is proved in Kobayashi and Otsuki (2014) that MAX-FLOW $\leq$ MIN-CUT $\leq$ MAX-FLOW $+ 1$ holds for any instances.

In this paper, we improve the existing polynomial time algorithms (Kobayashi & Otsuki, 2014) for the GMCCD and the GMFCD under the assumption that MIN-CUT $= MAX-FLOW$. Although we need the extra assumption that MIN-CUT $= MAX-FLOW$ for our improvements, this equality holds in almost all practical cases. Indeed, MIN-CUT $= MAX-FLOW$ holds in a large practical network considered in Section 5, which motivates us to focus on the case meeting this assumption. Note that if this equality does not hold, then the proposed algorithm is the same as the one in Kobayashi and Otsuki (2014), which means that our algorithm can find an optimal solution correctly and conclude that MIN-CUT $= MAX-FLOW + 1$, although it requires much more running time.
Here we introduce some mathematical definitions concerning the GMFCD and the GMFCD, following (Kobayashi & Otsuki, 2014). For any s-t path $P$, we think of $P$ as being oriented from $s$ to $t$, and denote by $P^-$ the path oriented from $t$ to $s$ along s-t path $P$. For two s-t paths $P'$ and $P''$ without crossings, let $R(P', P'')$ denote the closed region encircled by $P' - (P'')^{-1}$ in clockwise orientation, where $P' - (P'')^{-1}$ is the closed curve obtained by concatenating $P'$ and $P''$ with $P''$ in the same orientation. Let $|R(P', P'')|$ be the number of faces of $G$ contained in $R(P', P'')$. For two s-t paths $P'$ and $P''$ without crossings, we call a pair $(P', P'')$ clockwise hole-disjoint if for any hole $H$ in $H(r_b, r_p)$, $R(P', P'') + H$ is connected. Note that if two s-t paths $P'$ and $P''$ are hole-disjoint, $(P', P'')$ is clockwise hole-disjoint, while the opposite is not always true. $P'$ and $P''$ are hole-disjoint if and only if both pairs $(P', P'')$ and $(P'', P')$ are clockwise hole-disjoint.

For a pair of edges $e, e' \in E$, if there exists a hole $H \in H(r_b, r_p)$ that intersects both edges, we take two points $w_{e, e'}$ on $e$ and $w_{e', e}$ on $e'$ arbitrarily that are contained in hole $H$. Then, define $W := \{w_{e, e'} | e, e' \in E \text{ contained in a common hole}\}$. Theoretically, $|W|$ is obviously bounded by $|E|^2$. In practical networks, $|W| = O(|E|)$, that is, $|W| = O(|V|)$. Since there are only a small number of pairs that are close enough to be contained in a common hole in $H(r_b, r_p)$.

3. Max-flow algorithms

3.1. Existing max-flow algorithm

In this subsection, we review the algorithm for the GMFCD given by Kobayashi and Otsuki (2014). Algorithm 1 either finds pairwise hole-disjoint $k + 1$ s-t paths or reports the non-existence of such s-t paths, under the assumption that we already have pairwise hole-disjoint $k$ s-t paths.

Algorithm 1 Induction step.

**Input:** pairwise hole-disjoint $k$ s-t paths $P_1, \ldots, P_k$ and an s-t path $P_{k+1}$ in $R(P_1, P_k)$ such that $(P_k, P_{k+1})$ is clockwise hole-disjoint

**Output:** pairwise hole-disjoint $k + 1$ s-t paths

1: for $l = k + 1, k + 2, \ldots, k + |W| + 2$
2: if $(P_l, P_{l-1})$ is clockwise hole-disjoint then
3: return $P_1, \ldots, P_{k+1}$, which are pairwise hole-disjoint paths
4: else
5: let $P_{l-1}$ be the s-t path in $R(P_{l-1}, P_{l-1})$ such that $(P_{l-1}, P_{l})$ is clockwise hole-disjoint and $R(P_{l-1}, P_{l-1})$ is maximized under this condition
6: end if
7: end for
8: reports the non-existence of pairwise hole-disjoint $k + 1$ s-t paths

Lemma 1 (Kobayashi & Otsuki, 2014). Suppose that we are given a graph $G = (V, E)$ drawn in the plane, two distinct nodes $s$ and $t$, a hole radius $r_h$, and a protection radius $r_p > r_h$. Moreover suppose that we have pairwise hole-disjoint $k$ s-t paths $P_1, \ldots, P_k$, and an s-t path $P_{k+1}$ in $R(P_1, P_k)$ such that $(P_k, P_{k+1})$ is clockwise hole-disjoint. If Algorithm 1 reports the non-existence of pairwise hole-disjoint $k$ s-t paths, then MAX-FLOW = $k$.

An example of iterations in Algorithm 1 is shown in Fig. 2. Now, by using Algorithm 1 with $k = 1, 2, \ldots$ in order, an optimal solution of the GMFCD can be computed in polynomial time (see Algorithm 2).

Theorem 2 (Kobayashi & Otsuki, 2014). An optimal solution of the GMFCD can be computed in polynomial time.

**Proof.** Lemma 1 implies that an optimal solution of the GMFCD can be computed by using Algorithm 1 with $k = 1, 2, \ldots$ in order at most $|V|$ times (see Algorithm 2). For each $k$, the number of iterations of Algorithm 1 is at most $|W|$. Since each iteration runs in $O(|W|^2)$ time, the total running time is $O(|W|^2)$ ($k := \text{MAX-FLOW}$), which is polynomial.

As pointed out in Kobayashi and Otsuki (2014), in most practical cases, the running time is $O(|V|^2)$ since $k$ is small and $|W| = O(|V|)$.

3.2. Improvement of max-flow algorithm

We improve the max-flow algorithm (Algorithm 2) of Kobayashi and Otsuki (2014) in this subsection. As we reviewed in the previous subsection, the running time of the algorithm is $O(|V|^2)$ practically. The improvement does not change the theoretical computational time complexity; in other words, it still takes $O(|V|^2)$ time to solve the GMFCD in the worst case. Practically, however, the running time of the algorithm is decreased drastically in most cases, as we will see in Section 5.2.

Recall that Algorithm 2 uses Algorithm 1 with $k = 1, 2, \ldots$ in order, where Algorithm 1 either finds pairwise hole-disjoint $k + 1$ s-t paths or reports the non-existence of such s-t paths, under the assumption that we already have pairwise hole-disjoint $k$ s-t paths $P_1, \ldots, P_k$.

Let $k' := \text{MAX-FLOW}$. The most time-consuming part of Algorithm 2 is Algorithm 1 with $k = k'$, which needs $|W|$ iterations, whereas the total running time of Algorithm 1 with $k = 1, 2, \ldots, k' - 1$ is much smaller since only a small number of iterations are necessary. Therefore, to improve the running time of Algorithm 2, we have to aim at improving Algorithm 1 with $k = k'$ so that it needs a smaller number of iterations.

Suppose that we have pairwise hole-disjoint $k'$ s-t paths $P_1, \ldots, P_{k'}$. We note that it is sufficient to retain only $k$ s-t paths $P_1, \ldots, P_k$ in the $l$th iteration in Algorithm 1 with $k = k'$. Then, we show that if $P_{l+1} = P_{l-k+1}$ holds in some $k$ consecutive iterations indexed by $l$, it will hold in the following iterations (Theorem 3 below). For example, in Fig. 2, $P_{l+3} = P_l$ holds for $l \geq 4$.

This fact shows that if $P_{l+1} = P_{l-k+1}$ holds in some $k$ consecutive iterations in Algorithm 1, we can break iterations and terminate Algorithm 1 (see Algorithm 3).

Theorem 3. Algorithm 3, when used as a subroutine in Algorithm 2 in place of Algorithm 1, can find an optimal solution of the GMFCD.

**Proof.** It is sufficient to show that if there exists an integer $l$ such that

- $(P_l, P_{l+1})$ is clockwise hole-disjoint, and
- $P_{l+1} = P_{l-k+1}$
shown the first five iterations. Note that clockwise hole-disjoint. In this example, MAX-FLOW = 3, and so Algorithm 1 reports the non-existence of pairwise hole-disjoint four s-t paths after finite iterations. This figure shows the first five iterations. Note that \( P_3 \) and \( P_4 \) coincide with each other, and similarly \( P_2 = P_3 \) and \( P_1 = P_3 \) hold. Moreover we can easily see that \( P_{i, s} = P_i \) holds for \( i \geq 4 \).

**Algorithm 3** Improvement of Algorithm 1.

**Input:** same as Algorithm 1

**Output:** same as Algorithm 1

```
1: for \( l = k + 1, k + 2, \ldots, k + |W| + 2 \) do
2:   if \((R_l, R_{l-k})\) is clockwise hole-disjoint then
3:     return \((R_{l-k}, \ldots, R_{l-1}, P_l)\), which are pairwise hole-disjoint paths
4:   else
5:     let \( P_{l+1} \) be the s-t path in \( R(P_{l-k}, P_{l-k+1}) \) such that \((R_l, P_{l+1})\) is clockwise hole-disjoint and \( R(R_{l+1}, P_{l-k+1})\) is maximized under this condition
6:     if \( P_{l+1} = P_{l-k+1} \) holds for \( l' = l - k + 1, \ldots, l \) then
7:       break iterations and goto Line 11
8: end if
9: end if
10: end for
11: reports the non-existence of pairwise hole-disjoint \( k + 1 \) s-t paths
```

for \( l' = l - k + 1, l - k + 2, \ldots, l \) when Algorithm 1 is applied with \( k = \text{MAX-FLOW} \). The above conditions also hold for \( l' = l + 1, l + 2, \ldots, k + |W| + 2 \).

Consider the case of \( l' = l + 1 \). Since \((R_{l-k+1}, R_{l-2k+1})\) is clockwise hole-disjoint, \( P_{l+1} = R_{l-k+1} \), and \( P_{l-k+1} = R_{l-2k+1} \). \((R_{l+1}, R_{l-k+1})\) is also clockwise hole-disjoint. Furthermore \( P_{l+2} = R_{l-k+2} \) also holds since \( R_{l-k+2} \) is clockwise hole-disjoint and \( P_{l+1} = P_{l-k+1} \) for \( l' = l + 1, l + 2, \ldots, k + |W| + 2 \). □

### 3.3. Convergence property of the improved algorithm

The improvement that we gave in the previous subsection can reduce the running time of computing an optimal solution of the GM-FCD only if \( R_{l+1} = R_{l-k+1} \) holds in some \( k \) consecutive iterations in Algorithm 3, where \( k = \text{MAX-FLOW} \). This naturally raises the question whether there always exists an integer \( l \) such that \( P_{l+1} = P_{l-k+1} \) for \( l' = l - k + 1, \ldots, l \) in Algorithm 3 with \( k = \text{MAX-FLOW} \).

The objective of this subsection is to show that \( R_{l+1} = R_{l-k+2} \) holds for sufficiently large \( l \) under the assumption that MIN-CUT = MAX-FLOW. See Section 2 for the validity of this assumption. We mention that this question is for pure theoretical interest and that we can apply Algorithm 3 regardless of this issue.

**Theorem 4.** Suppose that we are given a graph \( G = (V, E) \) drawn in the plane, two distinct nodes \( s \) and \( t \), a hole radius \( r_h \), and a protection radius \( r_p \), and a protection radius \( r_p( > r_h) \). Moreover suppose that MIN-CUT = MAX-FLOW. Then, even if Algorithm 3 is modified so as not to stop after the iteration where \( l = k + |W| + 2 \) and when the modified algorithm is applied with
for \( n \geq k + 1 \). Then, since
\[
f(n) = \sum_{i=1}^{k} g_i(n') \leq k|V(G)||F(G)|
\]
for \( n = n'k \), \( f(n) \leq k|V(G)||F(G)| \) also holds for any \( n \geq k \).

Since \( f \) is an integer-valued non-decreasing function, there exists an integer \( l \) such that \( f(l' + 1) = f(l') \) for \( i' = l - k + 1, \ldots, l \). That is, \( P_{l'+1} = P_{l'-k+1} \) holds for \( i' = l - k + 1, \ldots, l \). \( \square \)

The above proof shows that Algorithm 3 terminates after at most \( k^2|V(G)||F(G)| + 1 \) iterations. Since the bound is finite, a sort of “convergence property” of the hole-disjoint max-flow algorithm is ensured theoretically under the assumption that \( \text{MIN-CUT} = \text{MAX-FLOW} \) holds. Practically, Algorithm 3 completes iterations after about 10–50 iterations in most cases, and hence the running time of the algorithm is decreased drastically in most cases (see Section 5.2).

4. Min-cut algorithms

4.1. Existing min-cut algorithm

In this subsection we review the polynomial min-cut algorithm for the GMCCD given in Kobayashi and Otsuki (2014). The following theorem ensures that either \( \text{MIN-CUT} = \text{MAX-FLOW} \) or \( \text{MIN-CUT} = \text{MAX-FLOW} + 1 \) holds, which plays a significant role in the min-cut algorithm.

Theorem 5 (Kobayashi & Otsuki, 2014). Suppose that we are given a graph \( G = (V, E) \) drawn in the plane, two distinct nodes \( s \) and \( t \), a hole radius \( r_h \), and a protection radius \( r_p \). Then,
\[
\text{MAX-FLOW} \leq \text{MIN-CUT} \leq \text{MAX-FLOW} + 1.
\]
Moreover, we can compute a hole cut whose size is at most \( \text{MAX-FLOW} + 1 \) in polynomial time.

This theorem implies that in order to find an optimal solution of the GMCCD, it is sufficient to check whether there exists a hole cut of size \( \text{MAX-FLOW} \) or not. In what follows, for \( i = 1, \ldots, k \), let \( F_i \) be the set of all faces of \( G \) contained in \( R(P_{i-1}, P_i) \), where \( P_0 := P_k \).

Theorem 6 (Kobayashi & Otsuki, 2014). An optimal solution of the GMCCD can be computed in polynomial time (by using Algorithm 4).

Algorithm 4 Hole-disjoint min-cut algorithm.

Input: pairwise hole-disjoint \( k \) s-t paths \( P_1, \ldots, P_k \) and a hole cut of size \( k + 1 \)

Output: a hole cut of minimum size

1: construct the directed graph \( D = (\mathcal{F}, A) \) defined as follows:
\[
\mathcal{F} := \bigcup_i F_i,
\]
\[
A := \{ (f_i, f_{i+1}) \mid i \in \{1, \ldots, k\}, f_i \in F_i, f_{i+1} \in F_{i+1}, \exists H \in \mathcal{H}(r_h, r_p) \text{ intersecting } f_i, P_i, \text{ and } f_{i+1} \}.
\]

2: for all \( f \in \mathcal{F} \) do
3: try to find a minimum-length directed cycle \( Q_f \) in \( D \) containing \( f \) by a breadth-first-search from \( f \)
4: if the length of \( Q_f \) is \( k \) then
5: return a hole cut of size \( k \) corresponding to directed cycle \( Q_f \)
6: end if
7: end for
8: return a hole cut of size \( k + 1 \)

Note that if we have a directed cycle in \( D = (\mathcal{F}, A) \), which is defined in Algorithm 4, we can obtain a hole cut whose size is the same.
as the length of the directed cycle. The running time of Algorithm 4 is bounded by \( O(|V||W|) \), since the number of nodes and edges of the directed graph \( D = (F, A) \) is \( |F| = O(V) \) and \( |A| = O(|W|) \), respectively. Thus, in Line 3 of Algorithm 4, finding a minimum-length directed cycle takes \( O(|F_r| (|F| + |A|)) = O(V |W|) \) time. Practically, since \( |W| = O(V) \), the running time of Algorithm 4 is \( O(V^2) \).

4.2. Improvement of min-cut algorithm

We improve the min-cut algorithm (Algorithm 4) of Kobayashi and Otsuki (2014) for the GMCCD in this section. As we reviewed in the previous subsection, the running time of this min-cut algorithm is \( O(|V|^2) \) practically. The improvement does not change the theoretical computational time complexity, in the same way as our improvement for the GMFCD. Practically, however, the running time of the algorithm is decreased drastically in most cases, as we will see in Section 5.2.

Let \( k := \text{MAX-FLOW} \). First, we note that we can obtain pairwise hole-disjoint \( k \) \( s-t \) paths \( P_1, \ldots, P_k \) satisfying the following property, by using the improved max-flow algorithm (see Section 3.3).

Property A: The first edges of \( P_1, \ldots, P_k \), which are incident to \( s \), occur in this order clockwise at \( s \), and for \( i = 1, \ldots, k \), \( P_i \) is the \( s-t \) path such that \( (P_{i-1}, P_i) \) is clockwise hole-disjoint and \( P(P_{i-1}, P_i) \) is minimized under this condition (\( P_0 := P_k \)).

Here we use the following lemma, which we can show by applying the argument in the proof of Theorem 3 in Kobayashi and Otsuki (2014) to the case with Property A.

Lemma 7 (Kobayashi & Otsuki, 2014). Suppose that we are given a graph \( G = (V, E) \) drawn in the plane, two distinct nodes \( s \) and \( t \), a hole radius \( r_h \), and a protection radius \( r_p(> r_h) \). Moreover suppose that there exist pairwise hole-disjoint \( k \) \( s-t \) paths \( P_1, \ldots, P_k \) that satisfy Property A. Then, there exists a closed curve \( C \) which does not intersect with itself, separates \( s \) and \( t \), and is a concatenation of \( J_1, J_2, J_3, \ldots, J_k \) in this order, where

- for \( j = 1, 2, \ldots, k \), \( J_j \) is a curve contained in a face \( f_j \in F_j \), and face \( f_j \) and path \( P_j \) share an edge of \( G \) in common, and
- for \( j = 1, 2, \ldots, k \), \( J_j \) is a line segment in \( \mathbb{R}^2 \) that can be covered by a hole \( H \in \mathcal{H}(r_h, r_p) \).

By using the lemma, we obtain Algorithm 5.

Algorithm 5 Improved hole-disjoint min-cut algorithm.

**Input:** pairwise hole-disjoint \( k \) \( s-t \) paths \( P_1, \ldots, P_k \) which satisfy Property A

**Output:** a hole cut of minimum size

1: construct the directed graph \( D' = (F', A') \) defined as follows:

- \( F'_1 := \{ f_i : \text{all faces in } F_i \} \)
- \( F' := \bigcup_i F'_i \)
- \( A' := \{(f_i, f_{i+1}) : \text{all faces in } F'_i, f_{i+1} \in F_{i+1}' \}, \exists H \in \mathcal{H}(r_h, r_p) \) intersecting \( f_i, P_i, f_{i+1} \).

2: for all \( f \in F' \)
3: try to find a minimum-length directed cycle \( Q_f \) in \( D' \) containing \( f \) by a breadth-first-search from \( f \)
4: if the length of \( Q_f \) is \( k \) then
5: return a hole cut of size \( k \) corresponding to directed cycle \( Q_f \)
6: end if
7: end for

Theorem 8. Suppose that we are given a graph \( G = (V, E) \) drawn in the plane, two distinct nodes \( s \) and \( t \), a hole radius \( r_h \), and a protection radius \( r_p(> r_h) \). Moreover suppose that we have pairwise hole-disjoint \( k \) \( s-t \) paths \( P_1, \ldots, P_k \) that satisfy Property A. Then, Algorithm 5 can find an optimal solution of the GMCCD.

Proof. By Lemma 7, there exists a closed curve \( C \) which does not intersect with itself, separates \( s \) and \( t \), and is a concatenation of \( J_1, J_2, J_3, \ldots, J_k \).

5. Experimental results

In this section, we describe our experimental results. We implemented the algorithms for the GMFCD and the GMCCD, and evaluated their performance by computational experiments. Our experiments were conducted on a computer with Intel Core i7, 2.8 GigaHertz and 8 Gigabyte of memory. All programs are implemented in Java.

5.1. Experimental evaluation of the algorithms

We evaluate the performance of the improved max-flow algorithm and the improved min-cut algorithm by comparing them with the existing algorithms.

We generated 10 input plane graphs with 100, 300, 1000, 3000, 10000, 30000, 100000 nodes, respectively, randomly in a 300 x 400 rectangular. We randomly chose the positions of \( s \) and \( t \) in the rectangular and let \( r_h = 7 \) and \( r_p = 30 \).

Fig. 5 shows the result comparing the average running time of the improved max-flow algorithm with that of the existing max-flow algorithm to the 10 input graphs consisting of the respective number of nodes. Fig. 6 shows the result comparing the average running time of the improved min-cut algorithm with that of the existing min-cut algorithm.
algorithm, in the same way. We can see that the running time of the algorithms were improved from $O(|V|^{1.811})$ to $O(|V|^{1.162})$ for the GMFCD, and from $O(|V|^{1.811})$ to $O(|V|^{1.162})$ for the GMCCD, where we estimated the time complexity by the least-squares method (in double-logarithmic scale).

5.2. Application to NewYork road network

As we can see in the previous subsection, our improvements of the algorithms for the GMFCD and the GMCCD enable us to find in a realistic time optimal solutions in huge networks with more than 100,000 nodes. As a practical instance, we analyze reliability between John F. Kennedy International Airport (JFK) and Newark Liberty International Airport (EWR) in NewYork road network (9th DIMACS Implementation Challenge — Shortest Paths) consisting of 264,346 nodes and 366,923 edges.

We set two nodes $s$ and $t$ to be JFK and EWR, respectively, and let the hole radius $r_h$ be 2 kilometers (approximately), and the protection radius $r_p$ be about 7 kilometers (approximately). The solutions of the GMFCD and the GMCCD are shown in Fig. 7. We can see MAX-FLOW = MIN-CUT = 4. It took 32 seconds to solve the GMFCD by Algorithm 3, and 14 seconds to solve the GMCCD by Algorithm 5 (after solving the GMFCD).

The solutions show that even if three failures of radius 2 kilometers happen simultaneously, at least one transit routes remain intact. We can also see that all the disks of the optimal hole cut are placed on the bridges, which means that the bridges are vulnerable when we consider reliability of NewYork road network between JFK and EWR.

Here we focus on vulnerability of bridges. We can expect that if we protect one of the bridges, that is, if we introduce a new protective disk around the bridge and prohibit placing a hole whose center is contained in the protective disk, connectivity of the network will increase. Indeed, Fig. 8 shows that connectivity increases from 4 to 5 by protecting the bridge.

In this way, by using our algorithms, we cannot only find optimal solutions of the GMFCD and the GMCCD, but also identify vulnerable regions we should protect particularly.

We also note that our algorithms can be applied to the case with new protective disks, or with holes of different shapes under some conditions. More specifically, we can replace $\mathcal{H}(r_h, r_p)$ with any set $\mathcal{H}$ of convex shapes satisfying the property below (see the concluding remarks in Kobayashi & Otsuki (2014)).

Property B: If two line segments $L_1$ in $H_1 \in \mathcal{H}$ and $L_2$ in $H_2 \in \mathcal{H}$ are crossing, then, $H_1$ contains an endpoint of $L_2$ or $H_2$ contains an endpoint of $L_1$.

5.3. Shortening hole-disjoint paths

In the previous subsection, we successfully found optimal solutions of the GMFCD and the GMCCD in NewYork road network. However, each path of the hole-disjoint flow looks quite unnatural. In particular, as we can see in Figs. 7 and 8, the path starting from the lower-left part of the protective disk around JFK goes down outside the figure once, travels half round NewYork, comes in from the top of the figure, and finally reaches EWR, which is extremely long. Since a solution of the GMFCD represents a set of transit routes, it is not desirable that each transit route is unnecessarily long.

In this subsection, we shorten the paths by the following local search method. Let $k := \text{MAX-FLOW}$. We choose a path $P$ from among pairwise hole-disjoint $k$-s-t paths $P_1, P_2, \ldots, P_k$, fix the remaining $k - 1$ paths, and then find a shortest path $P'$ which is hole-disjoint from any of the remaining $k - 1$ paths by a depth-first-search, and replace path $P$ with another path $P'$, repeatedly, until the set of the paths is not updated.

We note that this algorithm stops in finite iterations since the total number of edges contained in $P_1, P_2, \ldots, P_k$ is non-increasing. Fig. 9
shows the result of this algorithm applied to Fig. 7. We can see that each path turns shorter. The number of iterations was 12 in this case.

6. Conclusion

In this paper, we discussed the geographical max-flow and the geographical min-cut in the Circular Disk Failure Model introduced by Neumayer et al. (2012). We improved the algorithms to find optimal solutions of the GMFCD and the GMCCD given by Kobayashi and Otsuki (2014). We also evaluated the performance of these algorithms by computational experiments. The running time of the algorithms was improved from $O(|V|^{2.172})$ to $O(|V|^{1.266})$ for the GMFCD, and from $O(|V|^{1.811})$ to $O(|V|^{1.162})$ for the GMCCD in random instances.

The improvements enable us to compute in a realistic time connectivity of huge networks consisting of more than 100,000 nodes. We analyzed connectivity of New York road network with 264,346 nodes. We showed that by using our algorithms, we cannot only find optimal solutions of the GMFCD and the GMCCD, but also identify vulnerable regions we should protect particularly. Moreover, we proposed a heuristic based on a local search method to shorten each path of a hole-disjoint flow.

Acknowledgments

This work is supported by JST, ERATO, Kawarabayashi Large Graph Project, Japan, and by KAKENHI Grant number 24106002, 24700004, 25242029, 26280004.

References