IEICE TRANS. INF. & SYST., VOL.E97–D, NO.10 OCTOBER 2014

PAPER Special Section on Frontiers of Internet of Things

# On Finding Maximum Disjoint Paths for Many-to-One Routing in Wireless Multi-Hop Network

**Bo LIU**[†a)], ***Member***, **Junzhou LUO**[†], **Feng SHAN**[†], **Wei LI**[†], **Jiahui JIN**[†], ***and*** **Xiaojun SHEN**[††], ***Nonmembers***

**SUMMARY** Provisioning multiple paths can improve fault tolerance and transport capability of multi-routing in wireless networks. Disjoint paths can improve the diversity of paths and further reduce the risk of simultaneous link failure and network congestion. In this paper we first address a many-to-one disjoint-path problem (MOND) for multi-path routing in a multi-hop wireless network. The objective of this problem is to maximize the minimum number of disjoint paths of every source to the destination. We prove that it is NP-hard to obtain k disjoint paths for every source when $k \geq 3$. To solve this problem efficiently, we propose a heuristic algorithm called TOMAN based on network flow theory. Experimental results demonstrate that it outperforms three related algorithms.

***key words:*** *disjoint path, multi-routing, multi-hop, heuristic algorithm*

## 1. Introduction

In wireless ad hoc and mesh networks, the reliability of data transmission has received more and more attentions. Multi-path routing is an important method affecting the system performance. Multi-path routing establishes multiple paths between a single source node and a single destination node [1], [2]. Multi-path routing has been widely studied and many protocols and theories have been proposed in the past years [1]–[5].

Moreover, some research has been done to maximize the advantages of multipath routing. Disjoint paths are of great significance in multi-path routing since they can improve the diversity of paths and further reduce the risk of simultaneous failure and the congestion of paths [6]–[12]. Disjoint paths can avoid the risk of Shared Risk Link Group [13] and provide high reliability. They can also balance the traffic among different links and avoid network congestion. Node disjoint and edge disjoint are two commonly used kinds of disjoint paths. Node disjoint is more preferred than edge disjoint for the multi-hop wireless network because node disjoint implies edge disjoint. In this paper, we mainly discuss edge disjoint path in order to simplify the whole process. Node disjoint problem can also be solved by our proposed algorithm through some transformations which will be demonstrated in Sect. 4.2.

Recently, Yasuhiro Ohara proposed an all-to-one routing algorithm called MARA [6] which takes MA ordering to build maximum alternative routes for all sources in a network where there is one destination and other nodes except the destination are all sources. However, their algorithm cannot work well if only some of the nodes are specified to be sources in the network. It is very common in some wireless sensor networks, some sensors not only collect data but also store-and-forward messages to the sink node as routing nodes. This is a typical many-to-one problem. In this paper, we discuss on building reliable and efficient multi-path routing in a wireless multi-hop network with one destination and multiple sources. We study how we can build maximum disjoint paths for every source-destination pair. The problem is called MOND (Many-to-ONe maximum Disjoint paths problem). The main contributions of this paper are: (1) the introduction of a new routing problem; (2) the proof of finding K disjoint multi-to-one routing problem is NP-hard for $k \geq 3$; (3) a heuristic algorithm to produce maximum disjoint paths for every source; (4) the evaluation of the proposed algorithm on several types of network topologies. Simulations show that our algorithm consistently outperforms 3 existing algorithms.

This remainder of this paper is organized as follows. Section 2 introduces necessary notations and background for the MOND problem. Section 3 presents that finding optimal solution for MOND is NP-hard. Section 4 presents a heuristic algorithm for this problem. In Sect. 5 the algorithm is evaluated in several topologies. Section 6 concludes this paper.

## 2. Related Works

Multi-path routing can provide high reliability and improve transport capability. As we have discussed in Introduction, many multi-path routing protocols [2]–[4] have been proposed in the previous research. A number of multi-path routing protocols are based on distance vector routing algorithms. For example, DASM [14] builds loop-free multipath routing by means of a generalization of Dijkstra and Scholten's diffusing computations. MDVA [15] also takes diffusing computation to build paths. MPATH [16] proposes that routers exchange second-to-last hop on the shortest path to destinations in addition to shortest distances to build loop-free multipath. These protocols are the extension of the shortest path and they usually need extra computational cost.

In addition, some universal routing mechanisms are proposed to build multiple paths. FIR [17] proposes to generate a routing table for every network interface by execut-

Manuscript received January 17, 2014.
Manuscript revised July 19, 2014.

[†]The authors are with the School of Computer Science and Engineering, Southeast University, Nanjing 211189, China.
[††]The author is with School of Computing and Engineering, University of Missouri, Kansas City, MO 64110 USA.
a) E-mail: bliu@seu.edu.cn

DOI: 10.1587/transinf.2013THP0015

ing the Shortest Path First (SPF) algorithm to improve the fault tolerance ability of network. Yang and Wetherall propose a mechanism call Deflection [18] which takes the previous hop to increase the number of next-hops.

The methods of finding optimized disjoint paths can be classified into four types: All-to-All, All-to-One, Many-to-Many, Many-to-One. Many-to-Many maximum disjoint paths problem in single-path routing is addressed in our another paper [19], where a requirement is defined that the route of one source node should not cross another source node. This requirement is reasonable for Internet but not for wireless networks. Many-to-One is a most popular routing problem in wireless multi-hop networks which is the focus of this paper. A special case of many-to-one routing is all-to-One routing problem where all nodes are sources except the destination node.

The most related work to our paper is done by Ohara, Imahori, and Meter [6]. They propose an algorithm called MARA. However, MARA sometimes may not truly improve the reliability because some available routes for some node may share a common node or link with other available routes for the same node. Obviously, if the common node or link fails, both nodes will be blocked simultaneously. Therefore, constructing multiple disjoint paths is more important than constructing more paths but not disjoint.

Now, a challenging problem is to find an optimal configuration of a given network so that every source has at least $k$ disjoint (node disjoint) paths to the destination, where $k$ is a given required redundancy (also called survivability). Due to the difficult nature there have been only sketchy discussions on many-to-one routing problem in the prior research. Deying Li [20] discuss how to find a minimum energy subgraph for a given subset of nodes and a destination node such that there are k node-disjoint paths from each node in the subset to the destination node in the sub-graph. However, their problem is completely different from ours since energy is not considered here. As far as we know, the many-to-one maximum disjoint paths routing problem without considering energy is first addressed in this paper.

## 3. NP-Hard Proof for Many-to-One Node-Disjoint Routing Problem

Given a network topology with one node $t$ specified to be the destination and a set $S$ of some other nodes specified to be the sources, we wish to configure its routing network such that the number of disjoint paths from any source is maximized.

In this section, we first give a formal definition of this problem and then prove that it is NP-hard to find the optimal solution for this problem.

### 3.1 Problem Formation

**Definition 1.** A network topology is represented by an undirected graph $G(V, E, t, S)$ where $V = \{v_1, v_2, \ldots, v_n\}$ represents the set of network nodes and edge set $E = \{e_1, e_2, \ldots, e_m\}$ represents the set of links. Edge $(u, v) \in E, u, v \in V$, if nodes $u$ and $v$ can send and receive messages each other. In addition, a special node $t \in V$ is specified as the sink node and a set of nodes $S \subseteq V - \{t\}$ are specified to be source nodes.

**Definition 2.** Given a network topology $G(V, E, t, S)$, an orientation is a function $F : E \to E'$ that gives every edge a direction such that $G(V, E', t, S)$ becomes a DAG (directed acyclic graph).

**Definition 3.** Given a network topology $G(V, E, t, S)$, a many-to-one routing is an orientation $F : E \to E'$ such that in the DAG $G(V, E', t, S)$, every node $u \in S$ has at least one path to $t$. Moreover, if every node $u \in S$ has at least $k$ node-disjoint paths to $t$. Then the orientation $F$ is called a (many-to-one) routing with redundancy $k$. Multiple paths $P_1, P_2, \ldots, P_k$ from source node $u$ to sink $t$ are said to be edge (node) disjoint if they do not has any common edge (node) except $u$ and $t$.

Now we define the routing optimization problem as follows.

**Definition 4.** (MOND problem) Given a network topology $G(V, E, t, S)$, a many-to-one routing is called optimal if its redundancy is maximized among all possible routings. The MOND (Many=to-ONe Disjoint routing) problem is to find an optimal routing for a given network topology.

### 3.2 NP-hardness Proof

**Definition 5.** Given a network topology $G(V, E, t, S)$ and a positive integer $k$, the $k$-MOND problem is to decide whether a many-to-one routing exists with redundancy $k$.

In the following, we show that the k-MOND is NP-hard if $k \geq 3$.

**Theorem 3.1** The $k$-MOND problem, $k \geq 3$, is NP-hard.

Proof. We prove this theorem by constructing a polynomial-time reduction from 3-SAT to 3-MOND. The proof for k-MOND ($k > 3$) can be obtained from 3-MOND easily.

Let $\Phi$ be a 3CNF formula that consists of s variables, $x_1, x_2, \ldots, x_s$ and $m$ clauses. We label the $t$ clauses with $C_1, C_2, \ldots, C_t$. For example, $\Phi = (x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee x_4) \wedge (x_1 \vee x_3 \vee \bar{x}_4)$ consists of 4 variables and 3 clauses, where $C_1 = x_1 \vee x_2 \vee x_3$, $C_2 = \bar{x}_1 \vee \bar{x}_2 \vee x_4$, $C_3 = x_1 \vee x_3 \vee \bar{x}_4$. We transform the 3-CNF to a network topology $G_\Phi$ (an undirected graph) as follows.

1) For each variable $x_i$ ($1 \leq i \leq s$), construct a small graph $G_i$ as shown in Fig. 1, where $a_i$, $\bar{a}_i$ ($1 \leq i \leq s$) are source nodes, $x_i$, $\bar{x}_i$, $A$, $B$, $C$ are not source nodes, $t$ is the destination. Note that the 4 nodes, $A$, $B$, $C$, $t$, are shared by all graphs $G_i$, $1 \leq i \leq s$. They occur only once in entire $G_\Phi$. We draw node $t$ 3 times in $G_i$ just for clarity and ease of understanding.

2) For each clause $C_j$, $1 \leq j \leq m$, construct a source node $C_j$. In additional, if a literal $y$ occurs in clause $C_j$, then construct an edge between node $y$ and node $C_j$. For example, in above example, clause $C_2$ contains $\bar{x}_1$, $\bar{x}_2$, and

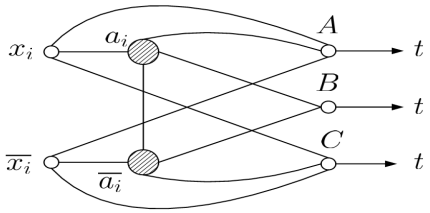
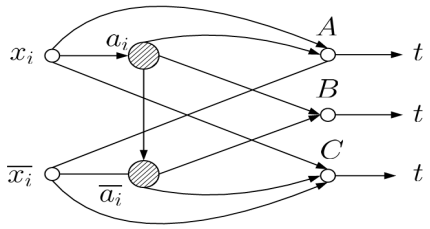
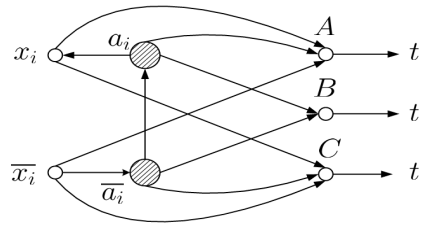



**Fig. 1** The construction of $G_i$.




(a) The orientation for $G_i$ when $x_i = 1$.




(b) The orientation for $G_i$ when $x_i = 0$.

**Fig. 2** The orientation of $G_i$.

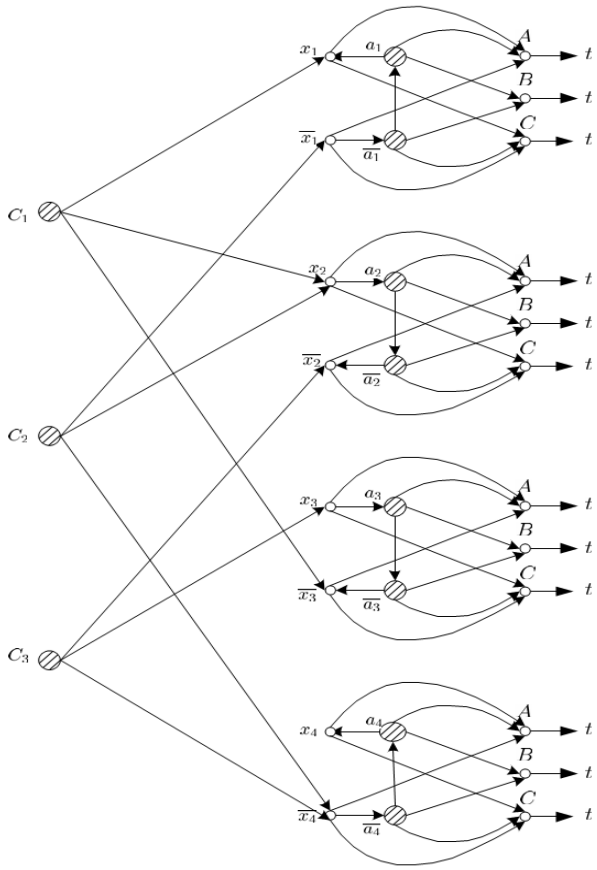



**Fig. 3** The graph $G_\Phi$ constructed for the example of $\Phi = (x_1 \vee x_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_2 \vee \bar{x}_4) \wedge (\bar{x}_2 \vee x_3 \vee \bar{x}_4)$ and its orientation for the assignment $x_1 = x_4 = 0$ and $x_2 = x_3 = 0$.

$x_4$, we include edges $(C_2, \bar{x}_1)$, $(C_2, \bar{x}_2)$, $(C_2, x_4)$ in the graph.

As an example, the graph transformed from $\Phi = (x_1 \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_4}) \wedge (\overline{x_2} \vee x_3 \vee \overline{x_4})$ is shown by Fig. 3. Note again that the nodes $A$, $B$, $C$, $t$ are single nodes. They should occur only once in the entire graph $G_\Phi$. Obviously, the construction of $G_\Phi$ can be done in polynomial time.

Now, we show that the 3-CNF $\Phi$ is satisfiable if and only the graph $G_\Phi$ can be oriented such that every source node has at least 3 node disjoint paths to node $t$.

(1) Assume the 3-CNF $\Phi$ is satisfiable. Let $\pi$ be a satisfying truth assignment for $\Phi$, we construct a feasible orientation for $G_\Phi$ as follows. It is clear, any path to $t$ must go through node $A$ or $B$ or $C$.

(1.1) If the value of $x_i$ is 1, then the edge $(a_i, \bar{a}_i)$ in $G_i$ $(1 \le i \le n)$ is given the direction from $a_i$ to $\bar{a}_i$ and other edges in $G_i$ are oriented as shown in Fig. 2 (a). Obviously, this orientation provides $a_i$ three disjoint paths to $t$ vis $A$, $B$, $C$ respectively. It also provides three disjoint paths from $\bar{a}_i$ to $t$ (1st path: $\bar{a}_i \to B \to t$; 2nd path: $\bar{a}_i \to \bar{x}_i \to A \to t$; 3rd path: $\bar{a}_i \to C \to t$).

(1.2) If the value of $x_i$ is 0, then the edge $(a_i, \bar{a}_i)$ is given the direction from $\bar{a}_i$ to $a_i$ and other edges are oriented as shown in Fig. 2 (b). This is a symmetrical case to (1.1). Again, both sources $a_i$ and $\bar{a}_i$ have three disjoint paths to t.

(1.3) Orient each edge $(C_i, y)$ from node $C_i$ to node $y$ where node $C_i$ represents clause $C_i$ and $y \in C_i$. $1 \le i \le m$. It is clear that any literal $y \in C_i$ can provide a path from $C_i$ to $t$ either via node $A$ or $B$ or $C$ if $y = 1$. Here is a path from $C_i$ to $t$ via $y$ if $y = 1$. However, if $y = 0$, then $y$ can provide a path from $C_i$ to $t$ either through node $A$ or $C$, but not $B$. Since there is at least one literal $y \in C_i$ has value $y = 1$, node $C_i$ has a path to $t$ via this node $y$ and $B$. Plus the other two paths via the other two literals which can go through nodes $A$ and $C$, each node $C_i$ has 3 node-disjoint paths.

Therefore, if the 3-CNF $\Phi$ is satisfiable then graph $G_\Phi$ can be oriented such that every source node has at least 3 node disjoint paths to node $t$.

Figure 3 illustrates an example of $G_\Phi$ and its orientation for $x_1 = x_4 = 0$ and $x_2 = x_3 = 0$. This truth assignment satisfies $\Phi$.

(2) Assume graph $G_\Phi$ can be oriented such that every source node has at least 3 node disjoint paths to node $t$. We will show how to satisfy $\Phi$. Specifically, we check the orientation of edge $(a_i, \bar{a}_i)$ in each $G_i$, $1 \le i \le n$. If it is oriented from $a_i$ to $\bar{a}_i$, then assign $x_i = 1$, otherwise $x_i = 0$. We prove that this assignment satisfy $\Phi$.

Because there are 3 disjoint paths from $C_i$ to $t$, one of the 3 paths must go through node $B$ through a literal node, say $x_k$, $x_k \in C_i$, $1 \le i \le m$. Then, edge $(x_k, a_k)$ in $G_i$ must be oriented from $x_k$ to $a_k$. Consequently, edge $(a_k, \bar{a}_k)$ must be

oriented from $a_k$ to $\bar{a}_k$, because otherwise, $a_k$ has only two outgoing edges, and cannot have 3 disjoint paths. So, $C_i$ has at least one literal whose value is 1, which implies that $\Phi$ is satisfied. We have successfully reduced the 3-SAT problem to the $k$-MOND problem in polynomial time. Theorem 3.1 holds. □

## 4. A Heuristic Algorithm for MOND

In this section, we propose a heuristic algorithm called TOMAN (TwO-phase MAx-flow based execution) for MOND problem.

### 4.1 Basic Idea of TOMAN

Algorithm TOMAN is consists of two phases: the global phase and the individual phase. In the global phase, we find a set of disjoint paths from sources to destination by maximum flow technique. In the individual phase, we attempt to improve the redundancy for those source nodes which have the least number of node-disjoint paths to destination $t$. The process repeats until no improvement can be made.

### 4.2 Algorithm Description

We focus on an edge-disjoint multiple path problem with the use of network flow technology.

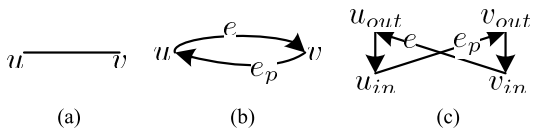
**Algorithm 1** *the TOMAN algorithm*
**input**:
A network topology $G=(V,E,t,S)$
**output**: An oriented graph $G'(V',E',t,S)$
**begin**
Initialize $G(V,E,t,S)$, result in a directed graph $G_d(V_d,E_d,t,S)$
$P$ = global-phase-search( $G_d(V_d,E_d,t,S)$ ,$t$) //P is path set
$P$ = individual-phase-adjust( $G_d(V_d,E_d,t,S)$ , $P$)
Assign direction to edges along paths $P$, obtain $G'(V',E',t,S)$
**return** $G'(V',E',t,S)$
**end**

*1) Initialization*: We convert the given un-directed graph into a directed graph.

In the conversion, the set of nodes remains the same. Each undirected edge $(u, v)$ is transformed to two opposite directed edges $\langle u, v\rangle$[†] and $\langle v, u\rangle$. Then every edge is assigned with a capacity 1. For convenience, we use $e_p = oppsite(e)$ to denote that edge $e$ and $e_p$ are opposite each other.

The proposed TOMAN algorithm is also capable of solving the node disjoint multiple path problem. In the problem of finding node-disjoint paths, each node is split into two halves with a directed edge in between, and each undirected edge is transformed to two directed links (Fig. 4 (c)). While in edge disjoint paths, nodes are preserved unchanged and every undirected edge is transformed to two directed edges with opposite directions (Fig. 4 (b)). Thus, TOMAN only differs in the initialization phase to solve the node disjoint path problem.

[†]We denote directed edge from a to b as $\langle a, b\rangle$, and undirected edge as (a, b).




(a) The original undirected edge.
(b) The conversion for edge-disjoint
(c)The conversion for node-disjoint

**Fig. 4** Conversion from undirected edge to directed edges.

*2) Global phase*: Firstly, we convert directed graph $G_d$ into a network graph called $H$, a super source node $R$ is added into $H$. $R$ connects to each source node in S by $\langle R, s_i\rangle$ with edge capacity of $(+\infty)$. We then maximize flow from $R$ to t in $H$.

When finding the maximum flow, we modify a classic Ford-Fulkerson algorithm [23] to support fairness among sources. More specifically, we revise the search order of finding the next augmenting path.

Ford-Fulkerson algorithm works in iteration. In iteration, the breadth-first search is applied on the residual graph to find an augmenting path. In TOMAN, we specify the search in order to find one augmenting path for every source node in a cyclic way such that each source node is given equal chance to get an augmenting path. This is presented in function of global-phase-search, which is based on the classic Ford-Fulkerson algorithm.

Function 1 global-phase-search
input: a directed graph $G_d$
output: a set of paths
define: $Q$ as a node set; $i$=1 as an index variable;
$P$ as a path set;
$H$ as a network graph
begin
$Q=\{s_i | i=1,2,...n\}$
construct $H$ from $G_d$
while $Q$ is not empty
if $s_i \in Q$
find an augmenting path $f$ from $R$ to $t$ in $H$ which contains edge $< R, s_i >$
if $f \neq null$
$P$=augment($P$, $f$)
$H$=updateGraph($H$, $P$)
else
$Q = Q - \{s_i\}$
end if
end if
i=(i mod n)+1
end while
return $P$
end

*3) Individual phase*: This is the core part of our algorithm. Since different sources can share common paths, we

always try to improve the value of $\min\{K_i\}$ in this phase. All sources are sorted according to the number of their multiple paths to $t$ in ascending order. Source $s_i$ with fewest paths is selected to find augmenting path. In this way, $\min\{K_i\}$ is improved.

In order to describe the *individual phase* clearly, a new graph named *p-residual graph for* $S_k$ called $G_P^{S_k}$ is proposed. The main difference between p-residual graph $G_P^{S_k}$ and residual graph is: Every edge in residual graph traversed by any path has an backward edge with non-zero capacity; however, in the p-residual graph, only the edges on the path of $s_i$ have non-zero backward edge, the edges on any path of other source nodes have zero backward edge and non-zero forward edge.

Algorithm 2: construct-p-residual-graph( $G_d(V_d, E_d)$ , $S$, $t$, $P$)

1: let p-residual graph $G_P^{S_k}$ be with the same node set as that of $G_d$
2: **for each** edge $e$=($u$,$v$) of $G_d$ **do**
3: accordingly adding a forward edge $e$=<$u$,$v$> and a backward edge $e'$ =< $v$,$u$ > in $G_P^{S_k}$
4: setting the capacity capacity($e$)=1 and $capacity(e') = 0$
5: **if** $e$=<$u$,$v$> is an edge belonging to any path in $P$ in $G_d$ **then**
6: set $e_p$=opposite(e), capacity($e_p$)=0 in $G_P^{S_k}$
7: set capacity( $e'_p$ )=0, where $e'_p$ is the backward edge of $e_p$.
8: **end if**
9: **if** $e$=<$u$,$v$> is traversed by $P_i (i \neq k)$ in $G_d$ **then**
10: set capacity($e$)=1, capacity($e'$)=0 in $G_P^{S_k}$
11: where $e'$ is the backward edge of $e$
12: **end if**
13: **if** $e$=<$u$,$v$> is a path edge of $P_k$ **then**
14: set capacity($e$)=0 and its backward edge capacity($e'$)=1
15: **end if**
16: **end for**

Notes:
(1) Note for Line 3–4, when the forward edge is traversed, it represents a flow; when the backward edge is traversed, it means previous flow on this edge is "canceled". The capacities of forward edges and backward edges are initially set to be 1 and 0 respectively.
(2) For the "**if**" part in Line 5–8, according to the initialization phase, every edge $e$ has an opposite direction (backward) edge $e'$ in $G_d$. Every edge on the path $P$ has an opposite direction edge (backward) $e'_p$. There are two pares of edges: e and $e'$, $e_p$ and $e'_p$. If an edge of one pair is traversed by a previous path, the other pair of edges should not appear in any augmenting path. As the example shown in Fig. 5, if there is a flow through $e$, then $e_p$ and $e'_p$ should never appear in any augmenting path. So their capacities are both set to be 0.
(3) Note for the "**if**" part in Line 9–12, since the precondition of the adjustment is that paths of other sources can be influenced. When we determine the augmenting path for $s_k$, the flow of other sources cannot be push backward (canceled). So the direction of the edges which are traversed by any path of $s_i$ ($i \neq k$) should be kept unchanged in $G_P^{S_k}$ and the capacity of their backward are set 0. However, these edges can be share between different flows, so the capacity of their forward edges is set to 1.
(4) For Line 13–15, since we need to find augmenting path for $S_k$, previous flows of $s_k$ on any edge can be "undo" by pushing them on the backward edge. This is illustrated in Fig. 6. So, in this step, we set the capacity of the backward edge to be 1, and the capacity of the forward edge to be 0. Note, that, some edges may have multiple flows, both for $s_k$ and $s_i$ ($i \neq k$). In that case, the capacity of the backward edge and forward edge are set in Line 9–12 and then reset in Line 13–15.

The main method of the individual phase includes the following steps. Let $K_{l1}$ be the smallest number in $\{K_i\}$, and the $K_{l2}$ be the second smallest number, where $K_i = |P_i|$, $i = 1, 2, \ldots, n$. Suppose $P$ is the previously found paths set. The algorithm *construct-p-residual-graph* constructs a p-residual graph $G_P^{S_{l1}}$ for source node $s_{l1}$. Suppose there are $d$ augmenting paths from $s_{l1}$ to $t$. If $d < K_{l2} - K_{l1}$, we commit all the $d$ augmenting paths and update the path set $P$ and the individual phase stops. Otherwise, we commit only $K_{l2} - K_{l1} + 1$ augmenting paths for $s_{l1}$ and update the path set $P$ and re-calculate $\{K_i\}$. This is a balanced strategy.

### 4.3 An Example

An example of finding edge disjoint paths is presented in Fig. 7. In this example, Fig. 7 (a) shows the original graph G, Fig. 7 (b) shows the result of global phase and Fig. 7 (c)–(f) explain the process of individual phase.

The undirected graph G with two source nodes $s_1$ and $s_2$ and one destination node $t$ is shown in Fig. 7 (a). Graph $G = (V, E, t, S)$ is converted to directed graph $G_d = (V_d, E_d, t, S)$ in the initialize phase. A super source $R$ is added into $G_d$ and the result of global-phase-search is given

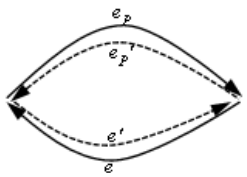
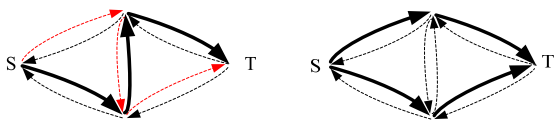
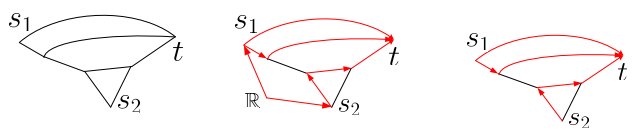



**Fig. 5** Relationships between $e$, $e'$, $e_p$, $e'_p$.

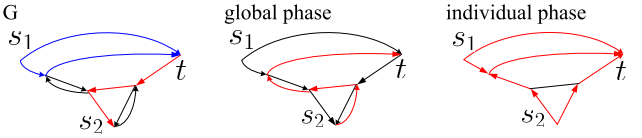



(a) (b)
(a) a path(in bold) and an augmenting path(in red)
(b) two paths after adjusting the augmenting path

**Fig. 6** Undo previous flows.




(a)The original graph G
(b)The result of the global phase
(c)The state before the individual phase




(d)P-residual graph for s2. Red arcs are the backward edge of paths belonging to $S_2$
(e)An augmenting path for s2. The path is marked by the red color.
(f)The final state

**Fig. 7** An example of find edge disjoint paths.

in Fig. 7 (b).

To adjust existing flows for $s_2$, we construct the p-residual graph for $s_2$ as shown in Fig. 7 (d) in which the edges on the paths of $s_2$ are reversed and the undirected edges are extended to two opposite direction edges. Figure 7 (e) shows the augmenting path found for $s_2$ based on the p-residual graph. According to the augmenting path in Fig. 7 (e), we adjust the flows in Fig. 7 (c), and the result is shown in Fig. 7 (f).

After this step, $s_2$ and $s_2$ all have two disjoint paths to $t$ and it is impossible to add a path for each node, the algorithm finishes.

### 4.4 Complexity Analysis

The time complexity of the proposed algorithm is O($E * l$), where $E$ is the number of all edges, $l$ is the total disjoint paths from all sources in $S$ to the sink. In the TOMAN, finding an augmenting path (in both global phase and individual phase) means one more disjoint path is introduced for one source. And there are totally $l$ disjoint paths. According to the analysis of Ford-Fulkerson algorithm [23], finding an augmenting path need $O(E)$ steps. Therefore, the proposed algorithm finishes in $O(E * l)$ steps.

## 5. Evaluation

We evaluated TOMAN on different network topologies to verify its performance in different node scale and link degrees.

### 5.1 Reference Multipath Routing Algorithms

To evaluate the performance of TOMAN, we selected several reference multipath routing algorithms for comparison. MARA is an optimal algorithm to solve the All-to-One Maximum Connectivity Routing Problem. MDVA [15] is the first distance vector routing algorithm that uses a set of loop-free invariants to prevent the count-to-infinity problem. Dijkstra calculates equal cost multipath (ECMPs) for shortest paths.

Additionally, we calculate an upper bound for the optimal solution in every experiment environment. We elaborate a strategy called MNDP (Maximize Number of Disjoint Paths algorithm) to obtain the upper bound. In MNDP, we calculate the number of disjoint paths for every source respectively. Since MNDP calculates one-to-one disjoint paths and the conflict is not considered, the result of MOND is definitely not larger than obtained by MNDP.

### 5.2 Experimental Settings

We do experiments on three typical network topologies: denoted IT0722, AS1239 and Sim300. IT0722 is large-scale network topology and the other two are small-scale network topology. The topology of AS1239 consists of about 300 nodes and 1500 links. To evaluate the performance of TOMAN on high-connectivity topologies, such as sensor network or ad hoc network, we generated Sim300 by BRITE [21] based on Waxman's probability model [22] which contains 300 nodes and 1500 links. We quantized the nodes of every topology into six groups based on the node degree. And listed the node number and the proportion of each group in Table 1.

**Table 1** Distirbution of node degree.

| #Degree | IT0722 | | AS1239 | | Sim300 | |
|---|---|---|---|---|---|---|
| | *#Num* | *%Pct* | *#Num* | *%Pct* | *#Num* | *%Pct* |
| 1 | 11516 | 33% | 31 | 10% | 0 | 0% |
| 2 | 13980 | 40% | 59 | 19% | 0 | 0% |
| 3-4 | 5106 | 14% | 78 | 24% | 0 | 0% |
| 5-6 | 1388 | 4% | 75 | 24% | 100 | 33% |
| 7-9 | 952 | 3% | 28 | 9% | 76 | 26% |
| ≥ 10 | 2011 | 6% | 44 | 14% | 124 | 41% |
| **Total** | **34953** | **100%** | **315** | **100%** | **300** | **100%** |

If the node degree of the destination is too low, it is not valuable to compute disjoint paths for the sources because the number of disjoint paths is impossible to exceed the node degree of the destination. So, in every topology, we randomly select a node whose node degree is larger than 40 as the destination. To verify the performance of TOMAN under different number of source nodes, we select 30, 60, 90, 120 and 150 nodes respectively whose node degree is larger than a threshold $D_s$ as sources in every topology. To evaluate TOMAN on different source node degree, $D_s$ is set to be 2, 3, 5 in IT0722 and AS1239, and 5, 7, 10 in Sim300. Because there are only 132 nodes whose degrees are greater than 5 in AS1239, the experiment with 150 sources when $D_s = 5$ cannot be executed. Similar situation occurs in Sim300 as well. The experiment on the same source node scale in every topology is executed for fifty times.

### 5.3 Performance Comparison

We compare the number of disjoint path of these algorithms under two different situations: average situation and minimum situation.

*A. Average number Comparison*

The average number of disjoint path is adopted to assess the performance of TOMAN and other algorithms.

The simulation results are shown in Fig. 8. The figure presents that TOMAN approaches the optimal solution in some cases, and it always outperforms MARA, MDVA and Dijkstra on finding maximum disjoint paths for many-to-one routing. It can be seen from Fig. 8 that the higher the connectivity of the topology is, the more disjoint paths are available for each source. And in every topology, the larger the proportion of source nodes is, the fewer average disjoint paths are built for each source, which attribute to the conflict.

On IT0722, TOMAN can always find disjoint paths as many as MNDP. That is because less than 0.05% nodes are

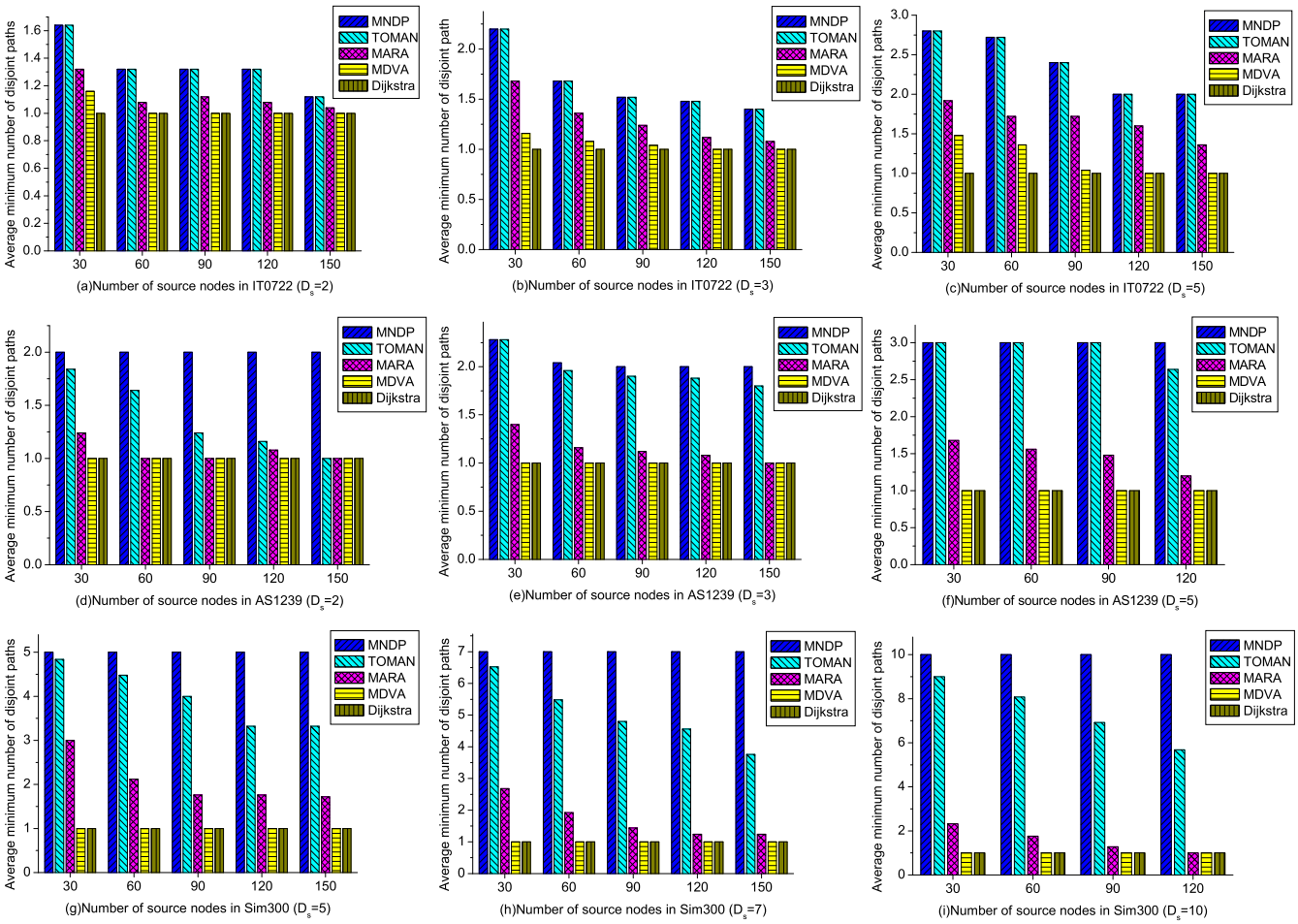



**Fig. 8** Comparison on average minimum number of disjoint paths.

selected as source nodes in IT0722 and the conflict rarely happens. And TOMAN almost calculates disjoint paths for every source independently. However, because the degree of most nodes in IT0722 is less than three, the performance gap between the result of TOMAN, MARA, MDVA and Dijkstra is not clear (See Fig. 8 (a)~(c)).

Experiments on Sim300 show that TOMAN is an efficient algorithm on the high-connectivity topology. On such topology, TOMAN can eliminate most conflicts by finding augmenting paths for the source nodes. The results of TOMAN are at least 3 times better than MARA. Figure 8 (i) shows that TOMAN finds 9 disjoint paths for 30 source nodes, while it is 2.23 for MARA, 1 for MDVA and 1 for Dijikstra. When source number is 120, the result of TOMAN is 5.68, while those of MARA, MDVA and Dijkstra are 1. This means TOMAN has considerable advantage over other three algorithms for high-connectivity networks.

*B. Minimum number Comparison*

To compare the probability that minimum number of disjoint paths found by TOMAN with other algorithms, another simulation is enforced. In this simulation, four algorithms are executed for fifty times and in every time a set of source nodes are randomly chose. For the same source nodes and in the same topology, the minimum of disjoint path of every algorithm is executed for fifty times. Figure 9 gives the results of comparison.

The results show that in fifty times tests, the minimum number of disjoint path found by TOMAN is always more than that of other algorithms. That is to say, the probability of finding more disjoint paths by TOMAN always exceed by others.

## 6. Conclusion

We have addressed many-to-one multipath routing for the first time in this paper and discussed the problem of building maximum disjoint paths. Finding the optimal solution for this problem ($k \geq 3$) is proved to be NP-hard and a heuristic algorithm called TOMAN is given. TOMAN is evaluated in several topologies of different scale and node degree. The results show that TOMAN can build more disjoint paths than available multipath algorithms both on average scenario and on minimum scenario. Especially for high density wireless network, TOMAN has far better performance than other algorithms. Disjoint multi-path routing is a good choice of multi-path routing in multi-hop wireless network including mesh network, ad hoc and sensor network. Our work not only promotes the multi-path theory,

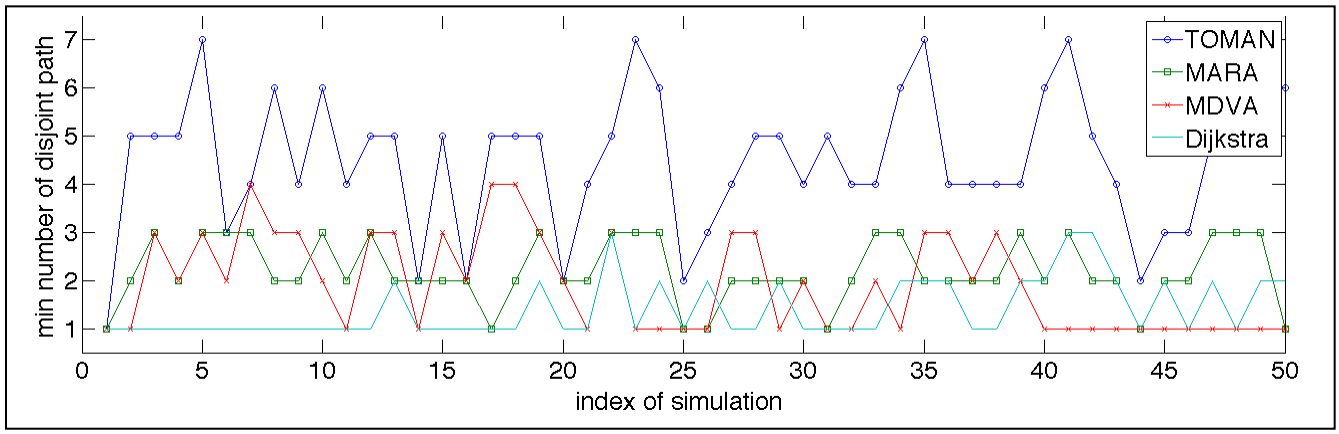



**Fig. 9** Comparison on minimum number of disjoint paths.

but also gives a practical performance reference in live network, especially in the high-connective wireless network. Our future work will include more experiments on different wireless network and consider energy parameters.

## Acknowledgments

This work is supported by National Key Basic Research Program of China under Grants No. 2010CB328104, National Natural Science Foundation of China under Grants, No. 61070158, No. 61003257, No. 61272531, No. 61370208, National High-tech R&D Program of China (863 Program) under Grants No. 2013AA013503, China National Key Technology R&D Program under Grants No. 2010BAI88B03 and No. 2011BAK21B02, China Specialized Research Fund for the Doctoral Program of Higher Education under Grants No. 20110092130002, Jiangsu Provincial Key Laboratory of Network and Information Security under Grants No. BM2003201, and Key Laboratory of Computer Network and Information Integration of Ministry of Education of China under Grants No. 93K-9.

### References

[1] Z. Ye, V. Srikanth, and K. Tripathi, "A framework for reliable routing in mobile ad hoc networks," IEEE INFOCOM 2003, pp.270–280, San Francisco, 2003.

[2] P. Yang and B. Huang, "Multi-path routing protocol for mobile ad hoc network," International Conference on Computer Science and Software Engineering, vol.4, pp.1024–1027, 2008.

[3] J. Wu and Y. Wang, "Social feature-based multi-path routing in delay tolerant networks," Proc. IEEE INFOCOM, 2012, pp.1368–1376, 2012.

[4] J.Y. Choi and Y.-B. Ko, "Multi-path routing with load-aware metric for tactical ad hoc networks," International Conference on Information and Communication Technology Convergence (ICTC), 2010, pp.370–375, 2010.

[5] Z. Qu, W. Ren, and Q. Wang, "A new node-disjoint multi-path routing algorithm of wireless Mesh network," International Conference on Computer, Mechatronics, Control and Electronic Engineering (CMCE), 2010, vol.4, pp.1–3, 2010.

[6] Y. Ohara, S. Imahori, and R.V. Meter, "MARA: Maximum alternative routing algorithm," Proc. IEEE INFOCOM, pp.298–306, April 2009.

[7] B.-l. Umverstty and R.-G. Israel, "Finding two disjoint paths between two pairs of vertices in a graph," J. AssoctaUon for Computing Machinery, vol.25, no.1, pp.1–9, Jan. 1978.

[8] B. Yang, S.Q. Zheng1, and E. Lu, "Finding two disjoint paths in a network with normalized $\alpha$+-MIN-SUM objective function," X. Deng and D. Du (Eds.): ISAAC 2005, LNCS 3827, pp.954–963, 2005.

[9] Y. Guo, F.A. Kuipers, and P. Van Mieghem, "A link-disjoint paths algorithm for reliable QoS routing," Int. J. Commun. Syst., vol.16, no.9, pp.779–798, 2003.

[10] A. Orda and A. Sprintson, "Efficient algorithms for computing disjoint QoS paths," IEEE Infocom'2004, pp.727–738, 2004.

[11] D. Sidhu, R. Nair, and S. Abdallah, "Finding disjoint paths in networks," Proc. ACM SIGCOMM'91, pp.43–51, Zurich, Switzerland, Sept. 1991.

[12] D. Xu, Y. Chen, Y. Xiong, et al., "On finding disjoint paths in single and dual link cost networks," Proc. INFOCOM, pp.715–725, March 2004.

[13] L. Shen, X. Yang, and B. Ramamurthy, "Shared risk link group SRLG)-diverse path provisioning under hybrid service level agreements in wavelength-routed optical mesh networks: Formulation and solution approaches," OptiComm, vol.5285, pp.126–138, Dallas, TX, Oct. 2003.

[14] W.T. Zaumen and J.J. Garcia-Luna-Aceves, "Loop-free multipath routing using generalized diffusing computations," Proc. IEEE INFOCOM, 1998, pp.1408–1417, San Francisco, CA, March 1998.

[15] S. Vutukury and J.J. Garcia-Luna-Aceves, "MDVA: A distance-vector multipath routing protocol," IEEE INFOCOM, 2001, pp.557–564, 2001.

[16] S. Vutukury and J.J. Garcia-Luna-Aceves, "MPATH: A loop-free multipath routing algorithm," J. Microprocess. Microsyst., vol.24, no.6, pp.319–327, 2000.

[17] S. Lee, Y. Yu, S. Nelakuditi, Z.-L. Zhang, and C.-N. Chuah, "Proactive vs reactive approaches to failure resilient routing," IEEE INFOCOM, 2004, pp.176–186, 2004.

[18] X. Yang and D. Wetherall, Source selectable path diversity via routing deflections, ACM SIGCOMM, 2006, pp.159–170, 2006.

[19] J. Tan, J. Luo, W. Li, and F. Shan, "Building reliable centralized intra-domain routing in Trustworthy and Controllable Network," IFIP/IEEE International Symposium on Integrated Network Management (IM), 2011, pp.462–469, 2011.

[20] D. Li, Y. Wang, Q. Zhu, and H. Yang, "Fault-tolerant routing: k-inconnected many-to-one routing in wireless networks," Theoretical Computer Science, 2009, doi:10.1016/j.tcs.2009.06.035

[21] A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: An approach to universal topology generation," Proc. Ninth International

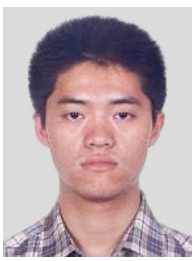
Symposium in Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'01), pp.346–353, Aug. 2001.

[22] B. Waxman, "Routing of multipoint connections," IEEE J. Sel. Areas Commun., vol.6, no.9, pp.1617–1622, 2002.

[23] L.R. Ford, Jr. and D.R. Fulkerson, Flows in Networks, Princeton Univ. Press, Princeton, NJ, 1962.

**Bo Liu** received her M.S. degree in computer science from Hohai University, China in 2002 and received her Ph.D. degree in Computer Science from Southeast University, China in 2007.She is currently an associate professor in the School of Computer Science and Engineering, Southeast University. Her current research interests include Social Networks and Pervasive Computing.

**Junzhou Luo** received his M.S. and Ph.D.degrees in Computer Science from the Southeast University, China in 1992 and 2000, respectively. He is currently a professor and the dean in School of Computer Science and Engineering, Southeast University. His research interests include Cloud Computing, Wireless Network and Network Security.



**Feng Shan** received the B.S. degrees in in computer scienceHoHai University, China in 2008. He is pursuing the Ph.D. degree now in Southeast university, Nanjing, China. His research interests include Wireless Sensor Network, algorithm, and wireless multi-hop networks.

**Wei Li** received his M.S. degree in computer science from Hohai University, China in 2002 and received his Ph.D. degree in Computer Science from Southeast University, China in 2007. He is currently an associate professor in the School of Computer Science and Engineering, Southeast University. His current research interests include Computer Network Control and Service Computing.

**Jiahui Jin** received the B.S. degrees in computer science in Southwest University, China in 2008. He is pursuing the Ph.D. degree now in Southeast university, Nanjing, China. His research interests include big data and cloud computing.

**Xiaojun Shen** received his M.S. and Ph.D. degrees in Computer Science from Nanjing University of Science and Technology, China in 1981 and 1989, respectively. He is currently a professor in School of Computer Science and Engineering, University of Missouri-Kansas City. His research interests include Computer Algorithm and Wireless Network.