Directed and Almost-Directed Flow Loops in Real Networks

M.Todinov
Department of Mechanical Engineering and Mathematical Sciences
Oxford Brookes University
Oxford, Wheatley, OX33 1HX, UK

***Abstract*—Directed flow loops are highly undesirable because they are associated with wastage of energy for maintaining them and entail big losses to the world economy. It is shown that directed flow loops may appear in networks even if the dispatched commodity does not physically travel along a closed contour. Consequently, a theorem giving the necessary and sufficient condition of a directed flow loop on randomly oriented straight-line flow paths has been formulated and a close-form expression has been derived for the probability of a directed flow loop. The results show that even for a relatively small number of intersecting flow paths, the probability of a directed flow loop is very large, which means that the existence of directed flow loops in real networks is practically inevitable. Consequently, a theorem and an efficient algorithm have been proposed related to discovering and removing directed flow loops in a network with feasible flows.**

The new concept 'almost-directed flow loop' has also been introduced for the first time. It is shown that the removal of an almost-directed flow loop also results in a significant decrease of the losses. It is also shown that if no directed flow loops exist in the network, the removal of an almost-directed flow loop cannot create a directed flow loop.

Keywords—directed flow loops; almost-directed flow loops; flow networks; optimization; classical algorithms; maximising the flow.

I. Directed Loops Of Flow In Networks

The existence of routing loops have already been reported in computer networks [1,2]. Due to inconsistencies in routing state among a set of routers, the packets physically travel along a closed loop and never reach their destination. Surprisingly, directed loops of commodity may exist even if none of the dispatched commodities physically travels along a closed loop. This point is illustrated by the examples in Fig.1 featuring supply networks (e.g. supply of petrol from a number of fuel terminals to a number of filling stations), where the same exchangeable commodity is transported along straight lines which are the shortest paths from sources to destinations. Selecting the shortest paths for a data transfer for example, is also a common strategy in communication networks [3].

Suppose that the throughput capacity of each source-destination straight-line path is 10 units. Despite that none of the dispatched commodities physically travels along a closed contour, a directed loop carrying 10 units of flow effectively appears between the intersection points (real or imaginary) x1,x2 and x3 in the network from Fig.1a and between nodes x1,x2,x3 and x4 in the network from Fig.1b.

Removing 10 units of flow from the segments (x1,x2), (x2,x3) and (x3,x4) in Fig.1a and from the segments x1,x2), (x2,x3), (x3,x4) and (x4,x1) in Fig.1b turns the flow circulating along the contours x1,x2,x3,x1 and x1,x2,x3,x4,x1 into zero, without diminishing the amount of total flow sent from the source nodes to destination nodes (Fig.1c,1d).

Figure 1 shows that directed loops of flow can even be found in networks where the intersecting source-destination paths are straight-line segments and no transported commodity physically travels along a closed contour.

A closed contour formed by a sequence of n nonempty sections ($n \geq 3$), in which the flows point along the direction of traversing the contour will be referred to as "directed flow loop".

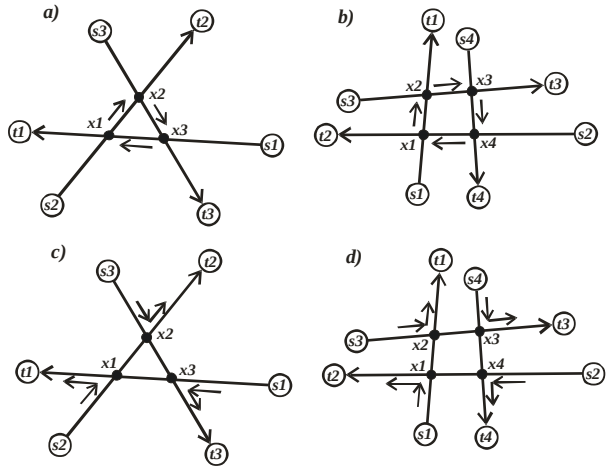

Fig. 1. a,b) Directed closed flow loops naturally appear in networks where the same type of commodity is transported between source-destination pairs; c,d) The directed flow loops can be removed without affecting the throughput flow from sources to destinations.

The directed loops of flow are highly undesirable because: (i) they increase unnecessarily the cost of transportation of the flow in the network, (ii) they consume residual capacity from the edges of the network and (iii) energy is unnecessarily wasted for maintaining the directed flow loops. The presence of directed loops of flow in networks causes big financial losses in the affected sectors of the economy. In computer networks, directed loops of flow consume bandwidth capacity

unnecessarily, increase data traffic and ultimately lead to congestion and delayed data transmission. This affects negatively the quality of service of the network.

In supply networks, the existence of directed loops of flow means high transportation costs because energy is wasted on circulating commodities unnecessarily.

The probability of existence of a directed flow loop between the intersection points of random source-destination paths has not yet been considered in the literature, despite its importance. Finding the strongly connected components of a graph, which implies the existence of cyclic paths has been has been considered before [4]. The question of identifying and removing directed flow loops in flow networks however, has been evading the attention of researchers for a very long time. This is evidenced by the fact that in spite of the years of intensive research on static flow networks, the algorithms for maximising the throughput flow published since 1956 leave highly undesirable directed loops of flow in the "optimised" networks. This surprising omission has already been demonstrated in [5] and [6].

There have been a number of published algorithms for optimising the flows in networks. Research related to optimizing network flows has been reviewed in [7-16]. Most of this research is related to determining the edge flows which maximise the throughput flow transmitted from a number of sources to a number of destinations (sinks).

There are two main categories of algorithms solving this problem. The augmentation algorithms preserve the feasibility of the network flow at all steps, until the maximum throughput flow is attained [17-20].

The second major category of algorithms for optimising the throughput flow are based on the preflow concept proposed in [21] and subsequently used as a basis for the algorithms proposed in [22] and [23]. For the preflow, the sum of all edge flows going into a node is allowed to exceed the sum of all flows going out of the node. As a consequence, the flow conservation law at the nodes may be violated and the nodes may contain excess flow. The central idea behind the preflow-push algorithms is converting the preflow into a feasible flow.

In a recent work [6], it was shown that optimising the network flow by using classical augmentation and preflow-push algorithms does not guarantee that there will be no directed flow loops in the optimised networks.

This point can be illustrated immediately with Fig.2, featuring a flow network with three sources s1, s2 and s3, each with capacity 100 units of flow per unit time and three destinations (sinks) t1,t2 and t3, each with capacity 100 units of flow per unit time. Suppose, for the sake of simplicity that the capacities of the separate connecting edges are also 100 units of flow per unit time. To maximise the throughput flow from the sources to the sinks, the classical Edmonds and Karp shortest-path algorithm [18] proceeds with saturating the shortest path (1,2,3,4) with 100 units of flow, followed by saturating the next shortest path (5,6,7,3,8,9,10) with 100 units of flow and finally, with saturating the remaining path (11,12,13,14,15,8,2,16,17,18) with 100 units of flow. As a result, a directed flow loop (2, 3, 8, 2) appears, carrying 100 units of flow. This flow loop is not only associated with wastage of energy. It also congests the network and makes it impossible to transfer additional flow, for example, from node 8 to node 2.

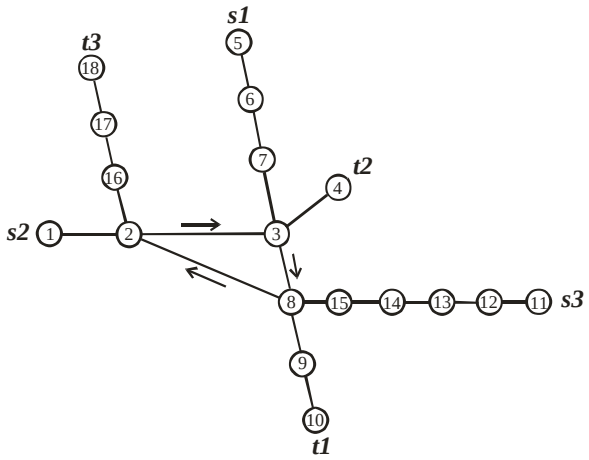

Fig. 2. Network, demonstrating that selecting sequentially the shortest paths between sources and destinations leaves a directed flow loop (2,3,8,2) in the optimised network. All edges have a flow capacity of 100 units.

Finally, to the best of our knowledge, no published analyses exist on almost-directed flow loops which are also associated with losses. The almost-directed flow loops are introduced and defined rigorously in the next section.

Consequently, the objectives of this paper are:

a) To show that directed flow loops can exist in networks even if all none of the dispatched commodity physically travels along a closed contour.

b) To estimate precisely the probability of a directed flow loop in a network defined by the intersections of straight-line randomly oriented source-destination paths.

c) To introduce the new concept almost-directed loop of flow in networks and formulate its basic properties.

d) To demonstrate that for flow networks (transportation networks, manufacturing networks, electrical networks and computer networks), directed and almost-directed flow loops are always associated with losses and their removal is highly beneficial.

e) To propose an efficient algorithm for identifying and removing directed loops of flow in networks with complex topology.

II. Removal Of Directed And Almost-Directed Loops Of Flow From Networks.

Denote the actual forward flows along the edges of a directed loop by Δ_{1f} , Δ_{2f} ,..., Δ_{nf} . These are all positive quantities and let the smallest among them be $\Delta_{\min} = \min\{\Delta_{1f},...,\Delta_{n-1f},\Delta_{nf}\}$. The amount $\Delta_{\min}$ will be referred to as 'bottleneck residual capacity'.

The directed flow loop can always be 'drained' by decreasing the flow along its edges by amount $\Delta \le \Delta_{\min}$. The result is a network which is characterised by smaller losses. A 'removal' of a directed flow loop involves determining its bottleneck residual capacity $\Delta_{\min}$ and draining the loop with the amount $\Delta_{\min}$. As a result, at least one of the edges will become empty and the directed flow loop will be 'broken'.

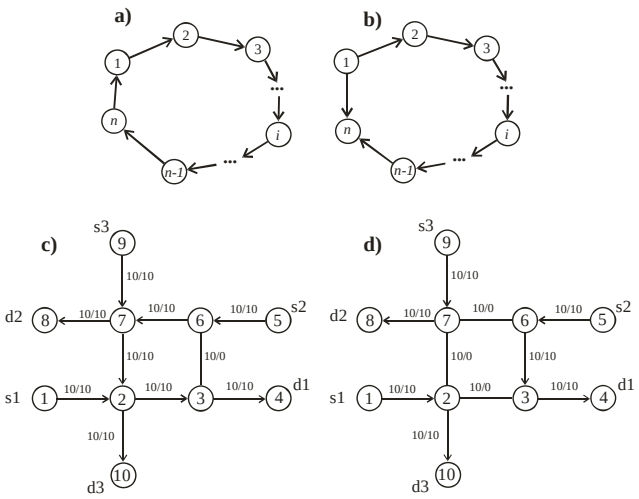

Fig. 3. a) A directed flow loop; b) An almost-directed flow loop; c,d) Removal of an almost-directed flow loop.

Suppose that there is at least one edge in the loop associated with non-zero transportation cost. The removal of flow along a directed flow loop leads to a new feasible flow and does not decrease the overall throughput flow to the destinations. In the process of flow loop removal, all edge flows have been decreased and no edge flow has been increased. Because there is at least one edge with nonzero transportation cost, the cost of transportation after the removal of the loop decreases. Thus, removing (draining) the directed flow loop (2, 3, 8, 2) carrying 100 unit of flow in the network from Fig.2, leads to a new feasible flow associated with reduced losses. The removal of the directed flow loop does not affect the throughput flow from sources to destinations.

In short, the removal of a directed flow loop always results in decreasing the losses in the network.

An almost-directed flow loop is a sequence of n sections ($n \ge 3$) in which the flow in $n-1$ edges points along the direction of traversing of the loop and the last (the closing n-th section) edge is augmentable in a direction opposite to the direction of traversing. This means that the flow in the closing edge can be increased in the direction opposite to the direction of the flow in the rest of the edges. As a result, the closing edge should not be fully saturated with flow in the direction opposite to the direction of the flow in the rest of the edges, because no flow augmentation will be possible for the closing edge.

Denote the actual flows in the forward edges by Δ_{1f}, Δ_{2f} ,..., Δ_{n-1f} and the residual space (not occupied with flow) in the closing edge by Δ_{nb}. These are all positive quantities and let the smallest among them be $\Delta_{\min} = \min\{\Delta_{1f},...,\Delta_{n-1f},\Delta_{nb}\}$. Again, the amount $\Delta_{\min}$ will be referred to as 'bottleneck residual capacity'.

The almost-directed flow loop can always be drained by decreasing the flow along the edges with forward flow by amount $\Delta \le \Delta_{\min}$ and increasing the flow with the same amount Δ along the closing edge. The draining operation does not violate the flow conservation at each node and the capacity constraints at the edges and leads to a new feasible flow.

Similar to the directed flow loops, the almost-directed flow loops are also associated with losses and their removal is highly beneficial. A 'removal' of an almost-directed flow loop means determining its bottleneck residual capacity $\Delta_{\min}$ and draining the loop with the amount $\Delta_{\min}$. As a result, either one or more of the edges with forward flow will become empty or the closing edge of the loop will become fully saturated with flow. As a result, the almost-directed flow loop will be broken.

If the cost of transportation per unit distance does not vary on the different edges, draining of an almost-closed loop always results in a reduction of the losses.

This point has been illustrated in Fig.3c with the almost closed flow loop (6,7,2,3) carrying 10 units of flow. The first label on the edges denotes the edge capacity and the second label – the actual flow through the edge. Flow of magnitude 10 units can be removed from the edges with forward flow and the flow along the closing edge (3,6) can be increased by 10 units. The result is the network in Fig.3d which is characterised by smaller losses.

Suppose that the cost of transportation per unit distance does not vary on different edges. The following theorem can then be stated.

Theorem 1. The removal of an almost-directed flow loop results in decreasing the losses in the network.

Proof. Denote the cost of transportation per unit length by c. Consider node 1 and node n. The edges (1,2), (2,3),...,(n-1,n) form a polygonal path between nodes (points) 1 and n. Denote the length of these sections (edges) by l_{12}, $l_{2,3}$,..., $l_{n-1,n}$. Denote the length of the closing section (edge) by $l_{n,1}$. The length of a polygonal path between two points however, is greater than the length of the distance between the two points. Therefore,

$$\sum_{i=1}^{n-1} l_{i,i+1} > l_{n,1} \qquad (1)$$

holds for a polygonal path which does not degenerate into a straight line. Because the cost of transportation per unit length is the same, removing the bottleneck flow $\Delta_{\min}$ from the almost-directed loop will result in a reduction of the transportation cost by $-c\Delta_{\min}\sum_{i=1}^{n-1} l_{i,i+1}$ along edges (1,2), (2,3),...,(*n*-1,*n*) and an increase of the transportation cost by $c\Delta_{\min} l_{n,1}$ along edge (*n*,1). Considering inequality (1), the inequality

$$-c\Delta_{\min}\sum_{i=1}^{n-1} l_{i,i+1} + c\Delta_{\min} l_{n,1} = c\Delta_{\min}(l_{n,1} - \sum_{i=1}^{n-1} l_{i,i+1}) < 0$$

is then valid, which means that removing the almost-directed loop will result in a net decrease of the cost of transportation and therefore in reduction of the losses.□

The next theorem permits the removal of directed flow loops and almost-directed flow loops to proceed in two stages. During the first stage only directed flow loops are removed while during the second stage, only almost-directed flow loops are removed.

Theorem 2. *If there are no directed flow loops in the network, the removal of an almost-directed flow loop cannot possibly create a directed flow loop.*

Proof. Suppose that the removal of the almost-directed loop (1,...,k,...,n) created a directed flow loop (Fig.4). Initially, by assumption, no directed flow loops exist in the network. Because the flow through the entire (1,k,n) section has been decreased by the removal of the almost-directed loop (1,k,n,1), a new directed flow loop can only appear if the closing edge (n,1) has been initially empty and after the increase of its flow from node 1 to node n, is now part of the new directed loop (n,p,1,n), (Fig.4).

Let (1,n,p,1) be the new directed flow loop. This is however impossible because before the removal of the almost-directed loop (1,k,n,1), a concatenation of sections (n,p,1) and (1,k,n) would have created a directed flow loop. This contradicts the assumption that no directed flow loops exist initially, therefore the theorem is true.

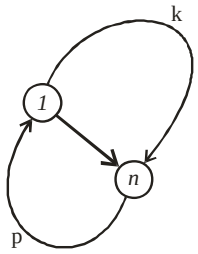

Fig. 4. In a network without directed flow loops, the removal of an almost-directed flow loop cannot possibly create a directed flow loop.

Finally, it can be shown that the process of removing almost-directed flow loops is finite for networks with integer capacities and must terminate.

Indeed, if the number of edges is m and the largest edge capacity is C, the maximum possible flow quantity contained in the network is mC. At each removal of an almost-directed flow loop, at least 1 unit of flow is removed from more than one edge and the same amount of flow is added to the single closing edge of the almost-directed loop. Consequently, the net change of the flow is negative and the amount of flow contained in the network decreases by at least 1 unit. As a result, after a finite number of steps, the process of removing the almost-directed loops will terminate.

III. Estimating The Likelihood Of A Directed Flow Loop In A Network Formed By The Intersections Of Randomly Oriented Straight-Line Source-Destination Paths

The unexpectedly high probability of existence of directed flow loops will be demonstrated by considering the general case where the source-destination paths are randomly oriented intersecting straight lines (Figure 5a).

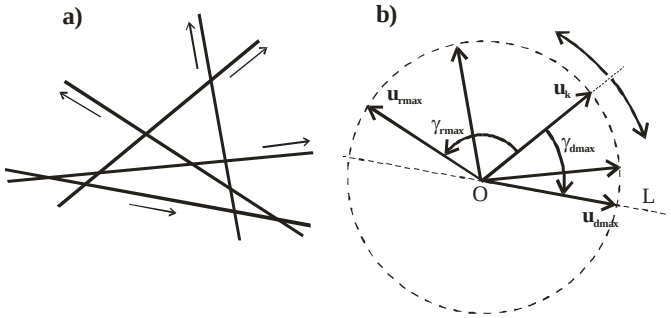

Fig. 5. a) Randomly oriented intersecting source-destination paths; b) All direction vectors of the source-destination paths can be translated to start from a common point *O*.

All source-destination paths transport the same type of interchangeable commodity (e.g. petrol) and for each source-destination path; there is a particular direction of the flow (Fig.5a).

It is assumed that there are at least three source-destination paths; there are no parallel paths and no three paths intersect into a single point. These conditions are natural and common. Indeed, for randomly oriented straight-line paths on a plane, it is very unlikely to find two parallel paths or three paths intersecting into a single point.

The likelihood that a directed flow loop will be present in the network, given that the orientation of the intersecting source-destination flows is random, will be termed 'probability of a directed flow loop for random source-destination paths'.

The existence of a directed flow loop anywhere between the points of intersection implies the existence of a triangular directed flow loop (Fig.6a). As a result, the existence of a triangular directed flow loop is a necessary condition for the existence of a flow loop. Conversely, the existence of a triangular directed flow loop is also a sufficient condition for the existence of a flow loop. Consequently, the probability of a directed flow loop for randomly oriented source-destination flows can be estimated by estimating the probability of a

directed triangular flow loop – the existence of three intersection points, between which the flow travels in the direction of traversing these points (Fig.6a).

A unit vector can be assigned to each source-destination path, whose direction is the same as the direction of the flow along the path (Fig.5b). The angle α the unit vector subtends with the horizontal axis (Fig.7a) gives the orientation of the source-destination path and the direction of the flow along the path. A random orientation of a source-destination path and the direction of its flow means that the angle α the direction vector subtends with the fixed horizontal x-axis is uniformly distributed in the interval $(0,2\pi)$. In other words, all possible orientations are characterised by the same probability.

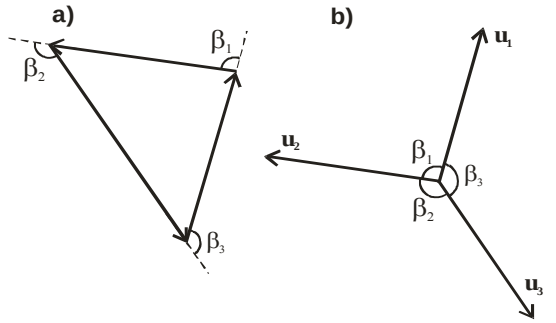

Fig. 6. a) A directed triangular flow loop; b) direction vectors of the source-destination paths forming the directed flow loop.

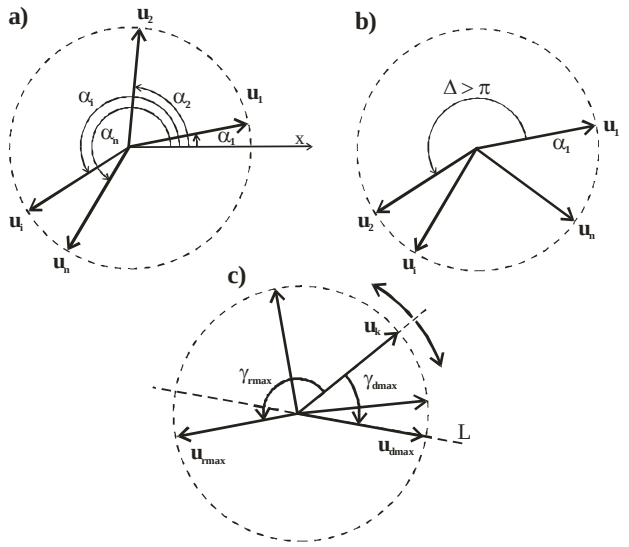

Fig. 7. a) Ordering the direction vectors, according to the angle they subtend with the horizontal axis; b) a gap of size at least π between two random direction vectors; c) If no half-plane can be selected where all direction vectors reside, there is always a possibility to select three direction vectors which do not reside in a single half-plane.

The unit vectors assigned to the source-destination paths will be referred to as 'direction vectors'. They can all be translated at the common origin *O*, as shown in Fig.5b. The following theorem then holds.

Theorem 3. The necessary and sufficient condition for a directed flow loop in a network defined by the intersections of randomly oriented straight-line source-destination paths, is the non-existence of a half-plane where all direction vectors reside.

Before proving this theorem two lemmas will be stated and proved.

Lemma 1. If any three selected direction vectors reside in a single half-plane, then all direction vectors reside in a single half-plane.

Proof. Let us select an arbitrary direction vector $\mathbf{u_k}$ and introduce counterclockwise and clockwise direction with respect to this vector to mark the angular positions of the rest of the direction vectors (Fig.5b). The angles γ_{di} mark the position of the unit vectors located in a clockwise direction from the vector $\mathbf{u_k}$ up to an angle equal to π. The angles γ_{ri} mark the positions of the unit vectors located in a counter-clockwise direction from the unit vector $\mathbf{u_k}$ up to an angle equal to π.

Let $\gamma_{d\max}$ be the angle corresponding to the most extreme direction vector $\mathbf{u_{dmax}}$ in a clockwise direction and $\gamma_{r\max}$ be the angle corresponding to the most extreme direction vector $\mathbf{u_{rmax}}$ in a counterclockwise direction. By assumption, any three direction vectors reside in a single half-plane, therefore the three direction vectors $\mathbf{u_k}$, $\mathbf{u_{dmax}}$ and $\mathbf{u_{rmax}}$ also reside in a single half-plane. Consequently, $\gamma_{d\max} + \gamma_{r\max} < \pi$ and the three vectors $\mathbf{u_k}$, $\mathbf{u_{dmax}}$ and $\mathbf{u_{rmax}}$ reside in the half-plane defined by the line *L* (oriented along the unit vector $\mathbf{u_{dmax}}$) and the unit vector $\mathbf{u_k}$ (Fig.5b). Because the rest of the unit vectors reside either within the angle $\gamma_{d\max}$ or within the angle $\gamma_{r\max}$ ($\gamma_{d\max}$ and $\gamma_{r\max}$ are the extreme angles corresponding to the direction vectors), all of the direction vectors must also reside in the half-plane defined by the line *L* and the unit vector $\mathbf{u_k}$. □

Lemma 2. *If no half-plane can be selected where all direction vectors reside, there is always a possibility to select three direction vectors which do not reside in a single half-plane.*

Proof. Similar to the previous proof, an arbitrary direction unit vector $\mathbf{u_k}$ is selected and counterclockwise and clockwise direction with respect to this vector is introduced to mark the angular positions of the rest of the direction vectors (Fig.7c). Again, $\gamma_{d\max}$ marks the most extreme direction unit vector $\mathbf{u_{dmax}}$ in clockwise direction up to an angle equal to π and $\gamma_{r\max}$ marks the most extreme unit vector $\mathbf{u_{rmax}}$ in counterclockwise direction up to an angle equal to π. The three vectors $\mathbf{u_k}$, $\mathbf{u_{dmax}}$ and $\mathbf{u_{rmax}}$ do not reside in a single half-plane (Fig.7c).

Indeed, suppose that the three vectors $\mathbf{u_k}$, $\mathbf{u_{dmax}}$ and $\mathbf{u_{rmax}}$ reside in a single half plane. As a result, $\gamma_{d\max} + \gamma_{r\max} < \pi$ and the three vectors $\mathbf{u_k}$, $\mathbf{u_{dmax}}$ and $\mathbf{u_{rmax}}$ should reside in the

half-plane defined by the line *L* (oriented along the direction vector $\mathbf{u_{dmax}}$) and vector $\mathbf{u_k}$. Because $\gamma_{d\max}$ and $\gamma_{r\max}$ are the extreme angles corresponding to the separate direction vectors, the rest of the unit vectors reside either within the angle $\gamma_{d\max}$ or within the angle $\gamma_{r\max}$. As a result, all of the direction vectors must also reside in the half-plane defined by the line *L* and the unit vector $\mathbf{u_k}$. This is however impossible because, according to our assumption, no half-plane can be selected where all direction vectors reside. This contradiction shows that the selected direction vectors $\mathbf{u_k}$, $\mathbf{u_{dmax}}$ and $\mathbf{u_{rmax}}$ do not reside in a single half plane.□

Now, Theorem 3 can be proved.

Proof of Theorem 3. First note, that the existence of a directed flow loop implies the existence of at least one triangular directed flow loop (Fig.6a) because no two source-destination paths are parallel. Suppose that the source-destination paths with direction unit vectors $\mathbf{u_1}$, $\mathbf{u_2}$ and $\mathbf{u_3}$, (Fig.6b) define a triangular directed flow loop and the angles between the source-destinations paths from Fig.6a are β_1, β_2 and β_3. Because the intersecting paths form a triangle, the sum of the angles β_1, β_2 and β_3 is always exactly equal to 2π (Fig.6b).

$$\beta_1 + \beta_2 + \beta_3 = 2\pi \tag{1}$$

In addition, for each angle β_i, the conditions

$$0 < \beta_i < \pi\ ,\ i = 1,2,3 \tag{2}$$

Are always fulfilled. Suppose that there is a single half plane where the direction vectors of all source-destination paths reside. In this case, the direction vectors $\mathbf{u_1}$, $\mathbf{u_2}$ and $\mathbf{u_3}$ of the paths forming the directed triangular loop will also reside in the same half-plane. However, this is impossible because in this case, the conditions (1)-(2) will be violated.

Now, suppose that there is no half-plane where all of the direction vectors $\mathbf{u_1}$, $\mathbf{u_2}$,..., $\mathbf{u_n}$ reside. According to Lemma2, in this case, we can always select three vectors u1, $\mathbf{u_2}$ and $\mathbf{u_3}$ which do not reside in the same half-plane. For these three vectors, conditions (1) and (2) will be fulfilled. Because, by assumption, no three source-destination pairs intersect in a single point, the source-direction paths which correspond to the selected direction vectors $\mathbf{u_1}$, $\mathbf{u_2}$ and $\mathbf{u_3}$ will form a triangular directed flow loop.□

Theorem 3 serves as a basis for calculating the probability of a directed flow loop. This probability can be determined by determining first the probability of the complementary event that no directed flow loop exists.

To calculate this probability, the random direction vectors are ordered in ascending order, according to the magnitude of the angle they subtend with the horizontal *x*-axis (Fig.7a).

The probability that there will be no directed flow loop is equal to the probability that all random direction vectors will lie in a single half-plane. All random direction vectors will lie in a single half-plane if and only if a gap of minimum length $\Delta = \pi$ exists between two random direction vectors (Fig.7b). There can be no more than a single gap of size $\Delta = \pi$, therefore, the random events corresponding to a gap between the first and the second direction vector, between the second and the third direction vector, etc., are mutually exclusive events. As a result, a single gap of minimum size $\Delta = \pi$ may be located in *n* distinct, mutually exclusive ways between the direction vectors.

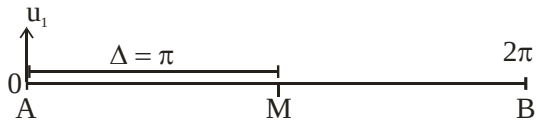

Fig. 8. A gap of length Δ can be located in *n* distinct ways between the direction vectors.

Let the circumference of the direction vectors circle be represented by the segment with length 2π (Fig.8). A gap of length $\Delta = \pi$ between direction vectors u_1 and u_2, can only occur if the rest of the *n*-1 random locations fall in the segment MB and none of them falls in the segment AM (Fig.8). Because the probability that a random direction vector location will 'fall' on the segment MB is $\pi/(2\pi) = 1/2$, the probability that *n*-1 random vector locations will fall in the segment MB is $1/2^{n-1}$.

Similarly, the probability of a gap between the second and the third direction vector is also $1/2^{n-1}$. As a result, the probability p of a gap of minimum size $\Delta = \pi$, between two direction vectors is a sum of the probabilities of these mutually exclusive events and becomes

$$p = \sum_{k=1}^{n} 1/2^{n-1} = n/2^{n-1} \tag{3}$$

Which is also the probability that no directed flow loop will exist. Because there can be either a directed flow loop or no directed flow loop, the probability of a directed flow loop is:

$$\Pr(directed\ \ flow\ loop) = 1 - n/2^{n-1} \tag{4}$$

The probability of existence of a directed flow loop from equation (4) has been plotted in Fig.9.

As can be verified, with increasing the number of intersecting random source-destination paths, the probability of a directed flow loop increases significantly. For five intersecting flow paths, the probability of a directed flow loop is already 69%.

Note from equation (4) that no matter how large the number n of intersecting source-destination paths is, the probability of a directed flow loop is always smaller than 1.

This means that for any possible number and for any possible orientation of straight-line flow paths on a plane, it is always possible to choose the directions of the flows along the

paths in such a way, that no parasitic flow loops appear between the points of intersection.

The results from equation (4) have been confirmed by a Monte Carlo simulation.

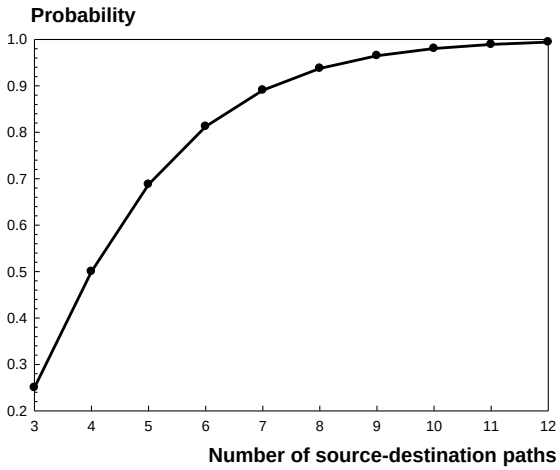

Fig. 9. Probability of a directed flow loop as a function of the number of intersecting source-destination paths.

IV. Discovering And Removing Directed And Almost –Directed Loops Of Flow In Networks

The algorithm for discovering and removing a directed flow loop is based on calling a depth-first-search (dfs) procedure from an initial node and subsequently calling the dfs-procedure from the descendents of this node, etc., until an already traversed node is discovered again. Initially, all nodes are marked as 'not visited' (coloured' white). As the nodes are visited by the dfs-procedure, they are marked as "visited" (coloured gray, Fig.10). We must point out that the dfs-procedure does not scan all successors of the current node. It scans all eligible successors only. A successor node *i* of the current node 'r_node' is eligible if: (i) there is an edge directed from node r_node to node *i* and the edge (r_node,i) carries nonzero flow. If edge (r_node, *i*) is empty, the successor *i is not considered by the dfs-procedure*. Suppose that the call of the dfs- procedure from node 'r_node1' does not discover any already traversed node (Fig.10a). In this case, after the return from the dfs-call, the node r_node1 is marked as 'completed' (coloured black) (the entry of the r_node1 in the array cmpl[] is set to '1', cmpl[r_node1]=1). Under these conditions, the following theorem holds.

Theorem 4. *In a network with feasible flow, a directed flow loop is present only if during a recursive call of the depth-first search procedure, an already traversed node has been discovered again and it has not been marked as 'completed'.*

Proof. Suppose that the node '*e*', which has been visited again, has been marked as 'completed' (Fig.10a). Because the node has been marked as 'completed' (cmpl[*e*]=1), an earlier depth-first search must have started from this node and no directed flow loop must have been discovered starting with node '*e*'. Therefore, node '*e*' discovered twice and marked as 'completed', cannot possibly belong to a directed flow loop.

Now suppose that the dfs-procedure has been called and during the subsequent calls of the dfs-procedure from subsequent descendent nodes, an already traversed node 'r_node' not marked as 'completed' (cmpl[r_node]=0), has been visited again (Fig.10b). Because node 'r_node' has been visited for a second time and it has not been marked as 'completed', no return from the earlier dfs-call initiated from this node has occurred (otherwise the node would have been marked as 'completed'). As a result, the 'r_node' has been visited again, after traversing a chain of descendent nodes starting from node 'r_node' and ending at r_node (Fig.10b). This essentially means that a directed flow loop has been discovered. □

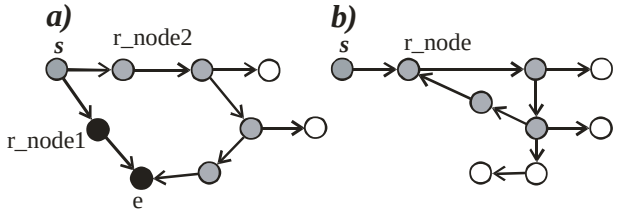

Fig. 10. Traversing the nodes of the network by recursive calls to the depth-first-search procedure.

Here is the algorithm in pseudo-code:

Direct all edges of the network to match the directions of the edge flows.
//As a result, the network is transformed from a network with undirected edges to a network with directed edges.

```
procedure retrieve_directed_flow_loop(cur_node)
{// retrieves and eliminates the identified directed
   loop of flow}

procedure dfs(r_node)
{
 marked[r_node] = 1;
 for i= 1 to all eligible successors of r_node do
{
 cur_node = current eligible successor;
 if (marked[cur_node] = 0) then {
                        pred[cur_node] = r_node;
                        dfs(cur_node);
                                    }
else { if (cmpl[cur_node] = 0) then
          {
            pred[cur_node] = r_node;
            retrieve_directed_flow_loop(cur_node);
            break;
          }
       }
}
 cmpl[r_node] = 1;
}

Statements before the call of the dfs-procedure:
for i=1 to n do {marked[i] = 0; cmpl[i] = 0; pred[i] = 0;}
dfs(1).
```

A directed flow loop can only be discovered if both conditions are fulfilled: (i) a node cur_node, already marked as 'visited' has been encountered during the search and (ii) the call dfs(cur_node) is still active, in other words, its activation record is still in the stack. After the end of the dfs (cur_node) call, node r_node is marked as 'completed' by the statement 'cmpl[r_node] = 1'. This is why, only when both marked[cur_node] = 1 and cmpl[cur_node] = 0 are encountered during the search, a directed loop of nonzero flow has been discovered. The directed loop of flow is subsequently retrieved and eliminated by the procedure ***retrieve_directed_flow_loop***(cur_node). The array pred[] records the predecessors of the visited nodes and helps retrieve the identified directed flow loop. The procedure ***retrieve_directed_flow_loop***(cur_node) retrieves the discovered loop of flow by starting with the statement 'k = cur_node' followed by a loop, where the statement k = pred[k] is repeatedly executed and followed by a check whether an already encountered node has been encountered again and whether node *k* is a descendent of the 'cur_node'. These checks are used for identifying the node which closes the identified directed loop carrying nonzero flow. After discovering the directed flow loop, the procedure determines the edge from the loop carrying the smallest amount of flow and subtracts this flow from the flows of all edges belonging to the loop. As a result, at least one edge in the directed loop becomes empty. Once an edge becomes empty, it remains empty until the end of the procedure for removing directed flow loops. This flow loop will not be discovered again during subsequent searches.

The proposed algorithm has been tested on a benchmark network which has the shape of a lattice (Fig.11). The lattice-type network has been selected because it is a natural network, often encountered in real applications. Because the lattice network is easily scalable, it provides an opportunity to isolate the impact of the size of the network on the algorithm's performance. To increase the number of loops, alternating directions of the flows from the sources s_i to the destinations d_i has been selected (Fig.11).

The same flow of 10 units per unit time has been assumed along each source-destination path.

Six different sizes of lattice-type network have been constructed and the network loops have been removed by the proposed algorithm, implemented in *C* and run on a computer with a processor *Intel(R) Core(TM) 2 Duo CPU T9900 @ 3.06 GHz*.

According to the number of intersecting horizontal and vertical paths, the following network sizes were tested: 2x2, 3x3, 4x4, 5x5, 6x6 and 7x7. The number of nodes corresponding to the 6 test-networks was: 12, 22, 32, 45, 60 and 77, correspondingly.

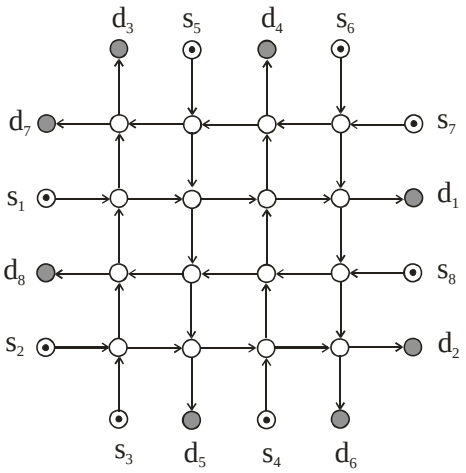

Fig. 11. Lattice networks used for testing the proposed algorithm.

The running time of the algorithm versus the size (the number of nodes) of the lattice network are shown in Fig.12. As can be seen, the running time of the algorithm is approximately proportional to the size of the lattice network.

It needs to be pointed out that identifying all directed cycles in the network, before removing the bottleneck flow from any of them, is not a feasible approach. To show why this is the case, consider a complete network where any two nodes (i,j) are connected with directed edges (Fig.13 shows a complete network with 4 nodes). The number of directed cycles in this network is equal to the number all possible subsets of 2 nodes, 3 nodes,...,n nodes. Consequently, the number of directed cycles is $2^N - N - 1$ and determining all possible cycles is a task of exponential complexity, a task which is practically impossible even for not very large *n*. In the proposed approach, identifying a directed loop of flow with the dfs-procedure and subtracting the bottleneck flow from the edges, has a worst-case complexity O(*m*), where *m* is the number of edges in the network. After each flow subtraction, at least a single edge remains empty. Therefore, after at most *m* steps, all directed loops of flow will be discovered and removed. As a result, the procedure for removing all directed loops of flow in a network has a worst-case running time O(m^2).

Finally, it can be shown [6] that maximising the throughput flow at a minimum cost, leaves no directed loops of flow in the network. The worst-case running time of maximising the throughput flow at a minimum cost however is significantly larger than the worst-case running time of the described procedure.

Similar to the case related to directed flow loops, the probability of existence of almost-directed loops of flow in networks can be estimated and an algorithm related to discovering and removing almost-closed flow loops can be developed. The developments related to almost-directed flow loops will be published elsewhere.

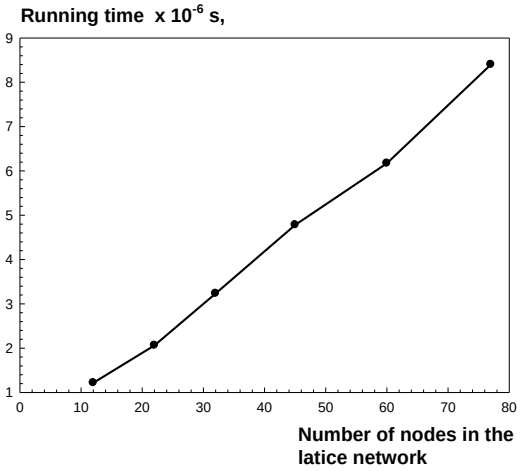

Fig. 12. Performance of the proposed algorithm for a different size of the lattice network.

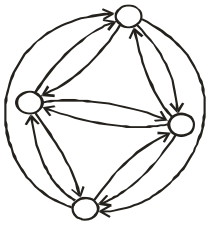

Fig. 13. Complete network with four nodes

V. CONCLUSIONS

1) Directed loops of flow can appear in networks with interchangeable commodity even if no transported commodity physically travels along a closed contour.

2) The necessary and sufficient condition for a directed flow loop in a network defined by randomly oriented straight-line flow paths, is the non-existence of a half-plane where all direction vectors reside.

3) A closed-form expression has been obtained for the probability of a directed flow loop for intersecting, randomly oriented flow paths, in a plane. Even for a relatively small number of intersecting flow paths, the probability of a directed flow loop is very large.

4) A theorem has been stated and proved regarding the existence of a directed flow loop in a network with feasible flows. On the basis of this theorem, a simple and efficient recursive algorithm has been proposed for discovering and removing directed loops of flow in networks. The algorithm discovers and removes a directed flow loop in linear time in the size of the network.

5) A new concept referred to as 'almost-directed flow loop' has been introduced and its basic properties formulated. It has been shown that the removal of an almost-directed flow loop results in decreasing the losses in the network.

6) It is shown, that the process of removal of almost-directed flow loops terminates after a finite number of steps. Furthermore, if no directed flow loops exist in the network, the removal of an almost-directed flow loop cannot create a directed flow loop.

7) The shortest-path strategy for optimising the throughput flow between sources and destinations does not guarantee that there will be no undesirable directed loops of flow in the optimised networks.

8) The directed and almost-directed flow loops in real networks are associated with wastage of energy and resources and increased levels of congestion. In real networks (transportation networks, manufacturing networks, electrical networks, computer networks, etc.), which include many intersecting flow paths, the existence of directed and almost-directed flow loops and the associated wastage of energy for maintaining these loops is practically inevitable. Consequently, optimising real networks by removing directed and almost-directed flow loops has the potential to save a significant amount of wasted resources for the world economy.

REFERENCES

[1] U. Hengartner, S. Moon, R. Mortier, and C. Diot. Detection and analysis of routing loops in packet traces. *Sprint ATL Technical Report TR02-ATL-051001*, May 2002.

[2] V. Paxson. End-to-end routing behavior in the Internet. *IEEE/ACM Transactions on Networking*, vol. 5(5), pp.610–615, 1997.

[3] Tanenbaum A.S., Computer networks, 4th ed., Pearson Education International, 2003.

[4] Sharir M., A strong-connectivit;y algorithm and its applications in data flow analysis, Computers and Mathematics with Applications, vol.7, pp.67-72, 1981.

[5] Todinov M.T., The dual network theorem for static flow networks and its application for maximising the throughput Flow, Artificial Intelligence Research, vol.2 (1), pp.81-106, 2013.

[6] Todinov M.T., Flow Networks, Elsevier, 2013.

[7] Ahuja R.K., T.L.Magnanti, J.B.Orlin, Network flows: Theory, Algorithms and Applications, Prentice Hall, 1993.

[8] Asano T., Y.Asano, Journal of the Operations Research Society of Japan, vol.43 (1), 2000.

[9] Goodrich M.T. and R.Tamassia, Algorithm design, John Wiley & Sons, 2002.

[10] Hu T.C., Integer programming and network flows, Addison-Wesley Publ.Company, Reading, Mass., 1969.

[11] Cormen T.H., T.C.E.Leiserson, R.L.Rivest, and C.Stein, Introduction to Algorithms, 2nd ed., MIT Press and McGraw-Hill, 2001.

[12] Tarjan R.E., Data structures and network algorithms, SIAM, Philadelphia, 1983.

[13] Kleinberg J. and E.Tardos, Algorithm design, Addison Wesley, 2006.

[14] Lawler E., Combinatorial Optimisation, Dover publications, Inc, New York, 1976.

[15] Goldberg A.V., E.Tardos and R.E.Tarjan, Network flow algorithms, in Algorithms and Combinatorics, v.9. Paths, Rows and VLSI-Layout, Sprmger-Verlag Berlin, Heidelberg, 1990.

[16] Papadimitriou C.H., K.Steiglitz, Combinatorial Optimisation: Algorithms and complexity, Dover publications, Inc., New York, 1998.

[17] Dinic E.A., "Algorithm for solution of a problem of maximum flow in a network with power estimation", Soviet Math. Doklady, vol. 11(8), pp.1277-1280, 1970.

[18] Edmonds J. and R.M.Karp, Theoretical improvements in algorithmic efficiency for network flow problems, Journal of the ACM, vol.19(2), pp.248-264, 1972.

[19] Elias P., A.Feinstein and C.E.Shannon, Note on maximum flow through a network, IRE Transactions on Information Theory, IT2, pp.117-119, 1956.

[20] Ford L.R. and D.R. Fulkerson, Maximal flow through a network, Canadian Journal of Mathematics, vol.**8**(5), pp.399-404, 1956.

[21] Karzanov A., Determining the maximal flow in a network by the method of preflows., Sov.Math.Dokl., 1974.

[22] Goldberg A.V. and R. E. Tarjan, A new approach to the maximum flow problem, Journal of ACM, vol.**35**, pp.921-940, 1988.

[23] Sleator D.D. and R.E.Tarjan. An O(*nm* log *n*) algorithm for maximum network flow. Technical report STAN-CS-80-831, Department of Computer Science, Stanford University, Stanford, CA, 1980.