

# Emergency Connectivity in Ad-Hoc Networks with Selfish Nodes<sup>\*</sup>

George Karakostas<sup>1,\*\*</sup> and Euripides Markou<sup>2,\*\*\*</sup>

<sup>1</sup> Department of Computing & Software, School of Computational Engineering & Science.  
McMaster University, 1280 Main Street West, Hamilton, Ontario L8S 4K1, Canada.

<sup>2</sup> Department of Computer Science and Biomedical Informatics,  
University of Central Greece, 2-4 Papasiopoulou str., Lamia 35100, Greece.  
E-mail: karakos@mcmaster.ca, emarkou@ucg.gr

**Abstract.** Inspired by the CONFIDANT protocol [1], we define and study a basic reputation-based protocol in multihop wireless networks with selfish nodes. Its reputation mechanism is implemented through the ability of any node to define a threshold of tolerance for any of its neighbors, and to cut the connection to any of these neighbors that refuse to forward an amount of flow above that threshold. The main question we would like to address is whether one can set the initial conditions so that the system reaches an equilibrium state where a non-zero amount of *every* commodity is routed. This is important in emergency situations, where all nodes need to be able to communicate even with a small bandwidth. Following a standard approach, we model this protocol as a game, and we give necessary and sufficient conditions for the existence of non-trivial Nash equilibria. Then we enhance these conditions with extra conditions that give a set of necessary and sufficient conditions for the existence of connected Nash equilibria. We note that it is not always necessary for all the flow originating at a node to reach its destination at equilibrium. For example, a node may be using unsuccessful flow in order to effect changes in a distant part of the network that will prove quite beneficial to it. We show that we can decide in polynomial time whether there exists a (connected) equilibrium without unsuccessful flows. In that case we calculate (in polynomial time) initial values that impose such an equilibrium on the network. On the negative side, we prove that it is NP-hard to decide whether a connected equilibrium exists in general (i.e., with some nodes using unsuccessful flows at equilibrium).

## 1 Introduction

In recent years there has been a great effort in designing robust and efficient wireless networks of devices that take upon themselves certain network responsibilities that used to be the responsibilities of a central network designer in traditional network design. For example, in ad-hoc networks the topology of the network is the result of cooperation amongst the nodes themselves: in a multihop wireless network, a successful transmission between a pair of nodes requires the cooperation of intermediate nodes in order for the transmitted packets to reach their destination. While this may be guaranteed in networks with a central authority forcing the nodes to cooperate, in the absence of such an authority cooperation may not be guaranteed. This is due to the *selfishness* of each node, i.e., the effort by the node to maximize its own utility without caring about the results of its actions on the overall network-wide outcome. For example, if battery life is a valuable resource for a node, forwarding packages between two other nodes consumes energy that doesn't result in any kind of pay-off for this

---

<sup>\*</sup> A preliminary version of this paper appeared in the Proc. 8th Latin American Theoretical Informatics Symposium (LATIN' 08), April 2008, Rio de Janeiro, Brazil, LNCS 4957, pp. 350-361.

<sup>\*\*</sup> Research supported by an NSERC Discovery Grant and MITACS.

<sup>\*\*\*</sup> Research supported by MITACS. This work was done during this author's stay at the School of Computational Engineering & Science of the McMaster University, as a postdoctoral fellow.

node, and as a result it may decide to stop cooperating in forwarding packages for others. If this behavior prevails throughout the whole network, it may eventually result in zero throughput for everybody, a phenomenon better known as the “Tragedy of the Commons” [5]. To cope with this problem one can offer *incentives* to nodes such as rewards for their cooperation or punishment for non-cooperation.

The two most commonly proposed forms of incentives are micro-payments, and reputation-based mechanisms. One of the main motivation for developing them is the desire of the network designer to not permanently punish a misbehaving node, but ‘re-socialize’ it if it changes its uncooperative behavior.

Micro-payment schemes are based on the concept of distribution of credit to nodes, so that nodes are compensated for their cooperation by (virtual) credit payments, that they can then use to pay intermediate nodes for forwarding their own traffic. Hence if a node is consistently uncooperative, it will run out of credit and will have to stop transmitting. Usually, the distribution and/or the expenditure of credit is controlled by a central authority. Examples of such protocols are [2, 10, 11, 3].

Reputation-based systems are based on lists that the nodes keep on the *reputation* of their neighbors, i.e., the fraction of packets forwarded by them. They use this information in order to decide how much traffic they should forward towards their neighbors. This may be decided in a *Tit-for-Tat* fashion, i.e., when a node has to relay a packet on behalf of a neighbor, it does so with the same probability with which this neighbor forwards its own packets (see [8, 9] for examples of such mechanisms). Or, the amount to forward can be decided according to (centralized or local) *ratings tables*, that give the nodes an indication of the behavior of other nodes; if a node’s rating of another node falls below a certain threshold, then the latter cannot be trusted to forward traffic, and therefore nothing is forwarded to it by the former, i.e., the edge connecting the two nodes is *cut* by the first node. An example of such a mechanism that actually distributes the reputation information so that each node can form its own ratings table is the CONFIDANT protocol [1]. More recent protocols [6, 7] limit the distribution of reputation information only to one-hop neighbors.

**Our results:** In this work we address the connectivity issues arising in such reputation-based systems. More specifically, we would like to study whether it is possible in such a selfish environment to lead all nodes towards an equilibrium with good connectivity properties. In fact, we are very ambitious: we are looking for driving them towards an equilibrium that permits a non-zero quantity of *every* traffic demand to be satisfied. The reason for such a strict requirement is the fact that in an emergency situation police, firemen, emergency medical personnel, etc. should be able to communicate with each other even if the achieved bandwidth is very small (but still enough for emergency signals to be able to travel through the network). From the above, it is not at all obvious whether such a goal can be achieved, given the fact that each network node is autonomously playing a protocol game, after it’s been set in its initial condition. Given the game-theoretic nature of such protocols, it is only natural to study them in terms of their (Nash) equilibrium states. Under this light, and given the rules of the game, i.e., the protocol, the most appropriate (indeed, in some cases the only) time a network designer can intervene in order to control the outcome is during the setting of the initial conditions, or, equivalently, by ‘rebooting’ the protocol with new initial values. This can be achieved by a separate broadcasting channel that all nodes are listening (‘snooping’) in, and whose packets are of the highest priority. Obviously, this is a

very intruding method, and it would defy the purpose of selfishness if it were to be applied very frequently. But one does not (hopefully) expect catastrophic emergency situations to arise that frequently. Therefore broadcasting will not be used often.

Inspired by the CONFIDANT mechanism, we study a basic reputation-based system. The strategy of every node consists of the amount of traffic flow it sends to its various receivers, the routing of this flow, the amount of flow it forwards for every commodity in which it doesn't participate as a sender or a receiver, and a non-negative *threshold* value for each outgoing edge. The latter set of values is an abstraction of the reputation mechanism: if the amount of flow that is forwarded by node  $x$  to node  $y$  (including flow that originates at  $x$ ), but is cut by  $y$  is more than the threshold value  $x$  has for  $y$ , then  $x$  disconnects edge  $(x, y)$ . Later on,  $y$  may end up cutting flow that is less than the current threshold value of  $x$  for  $(x, y)$ , in which case  $(x, y)$  reappears. The utility for every node increases with the flow originating at or destined for this node and reaches its destination, while decreases with the flow sent out or forwarded by this node (because, for example, the node has to spend battery energy to transmit).

The main drawback of this protocol is the assumption that every node has to make its strategy known to every other node. But at the same time, this complete knowledge of the game state gives great potential power to each node to affect parts of the network that are very far away, even in counter-intuitive ways, e.g., by sending flow whose sole purpose is to affect the current topology and discourage the flow of other nodes. Hence, this assumption may make our demand for complete connectivity even harder to achieve, and it may mean that things can be easier in a more restricted setting. For example, in a model where each user has a limited knowledge of other users' strategies, a user might not be able to calculate how her utility changes when she changes her strategy. Such a user has fewer options than in our model for a profitable strategy and the same or less power to affect distant parts of the network.

As a first step towards achieving connectivity, we are able to characterize the complexity of computing initial values that lead to a connected Nash equilibrium in our protocol. We do that, by giving necessary and sufficient conditions for the existence of *non-trivial* Nash equilibria. Then we enhance these conditions with extra conditions that give a set of necessary and sufficient conditions for the existence of *connected* Nash equilibria. Note that it is not always necessary for all the flow originating at a node to reach its destination at equilibrium. As mentioned above, a node may be using such *unsuccessful* flow in order to effect changes in a distant part of the network that will prove quite beneficial to it. We show that we can decide in polynomial time the existence of a connected Nash equilibrium *without unsuccessful flows*, and we can calculate (in polynomial time) initial values that impose such an equilibrium (should it exist) on the network using linear programming. On the other hand, we prove that it is NP-hard to decide whether a connected equilibrium exists in general (i.e., with some nodes using unsuccessful flows at equilibrium).

Our results are derived using game-theoretic concepts, which is the standard approach for analyzing such protocols, modeled as games. But we emphasize that, other than the assump-

tions mentioned above, we don't impose any restrictions on the network topology, or any statistical distribution on the nodes' decisions.<sup>3</sup>

## 2 Model and Terminology

In this section we describe our model for the network and the protocol the nodes follow. The set of connections that can be realized is given by a directed graph  $G(V, E)$ . We emphasize that, depending on the current state of the game, not all these edges may be present. For every origin-destination pair (commodity)  $(u, v)$ ,  $u, v \in V$  there is a demand  $d_{(u,v)}$  that  $u$  wants to send to  $v$ . The flow is splittable, and  $u$  decides how to split and route this flow. Again, the current state of the game may not allow  $u$  to send all of  $d_{(u,v)}$ , so the latter serves more as an upper bound on the flow actually sent. We denote by  $\mathcal{P}_i$  the set of paths connecting the  $i$ -th origin-destination pair in  $G$ , and let  $\mathcal{P} := \cup_i \mathcal{P}_i$ .

The current state of the network, together with the nodes' strategies are described by the following set of variables:

- $F_{(u,e,e',v)}^y$  **with**  $e, e' \in E, e = (x, y), e' = (y, z), u, v, x, y, z \in V$  **and**  $y \neq u, v$ : This is the flow of commodity  $(u, v)$  that  $y$  receives through  $e$ , and forwards further through  $e'$ .
- $f_{(u,e,e',v)}^y$  **with**  $e, e' \in E, e = (x, y), e' = (y, z), u, v, x, y, z \in V$  **and**  $y \neq u, v$ : This is the *decision* variable of  $y$  that sets an upper bound on the amount of flow  $\sum_{g=(w,x)} F_{(u,g,e,v)}^x$  routed through  $e'$  that  $y$  actually forwards through  $e'$ , i.e.,  $F_{(u,e,e',v)}^y = \min\{f_{(u,e,e',v)}^y, \sum_{g=(w,x)} F_{(u,g,e,v)}^x\}$  (notice that edge  $e'$  can be disconnected; in that case, what is being forwarded by  $y$  through  $e'$  is simply lost). We emphasize that  $f_{(u,e,e',v)}^y$  is just the  $y$ 's decision variable that determines what  $y$  will do *if* there is flow from  $u$  to  $v$  which has been forwarded from  $x$  to  $y$  and needs to be forwarded through  $e' = (y, z)$ , while  $\sum_{g=(w,x)} F_{(u,g,e,v)}^x$  is the actual flow that comes to  $y$  from  $x$  through  $e$ . So  $y$  maintains such a variable  $f_{(u,e,e',v)}^y$ , for every incoming edge  $e = (x, y)$  and every outgoing edge  $e' = (y, z)$ , and every commodity  $(u, v)$ .
- $O_{(u,e,v)}^y$  **with**  $e \in E, e = (x, y), u, v, x, y \in V$  **and**  $y \neq u, v$ : This is an auxiliary variable, defined as  $O_{(u,e,v)}^y = \sum_{e'=(y,z)} F_{(u,e,e',v)}^y$ . It is simply the total flow of commodity  $(u, v)$  coming to  $y$  through edge  $e$ , and being forwarded by  $y$  through all its outgoing edges  $e' = (y, z)$ .
- $I_{(u,e',v)}^y$  **with**  $e' \in E, e' = (y, z), u, v, y, z \in V$  **and**  $y \neq v$ : This is also an auxiliary variable, defined as  $I_{(u,e',v)}^y = \sum_{e=(x,y)} F_{(u,e,e',v)}^y$ . It is simply the total flow of commodity  $(u, v)$  coming to  $y$  through all its incoming edges  $e = (x, y)$ , and being forwarded by  $y$  through edge  $e'$ . Note that  $I_{(y,e',v)}^y$  is the flow originated at  $y$  and routed through  $e'$  with destination  $v$ .

<sup>3</sup> We don't assume any kind of synchronization amongst the nodes, but we do assume that the decision variables changes are instantaneous. Note that the game modeling the protocol is *not* a repeated game, and there isn't any notion of *rounds*.

- $\epsilon_x^y$ : This auxiliary variable is defined as  $\epsilon_x^y = \sum_{com.(u,v), v \neq y} (I_{(u,e,v)}^x - O_{(u,e,v)}^y)$ , i.e., as the part of the total flow that comes to  $y$  through  $e$  and is being blocked by  $y$ .
- $s_{(u,P,v)}^u$ : This is the *decision* variable of  $u$  that determines how much flow of commodity  $(u, v)$  node  $u$  routes through path  $P$  (whether this flow amount eventually reaches  $v$  or not).
- $THR_x(y)$ : This is the *decision* variable of node  $x$  that defines an upper bound on the flow forwarded by  $x$  and cut by  $y$  that  $x$  can tolerate before it cuts edge  $(x, y)$ . We consider edge  $(x, y)$  disconnected when  $\epsilon_x^y > 0$  AND  $THR_x(y) \leq \epsilon_x^y$ . Hence edge  $(x, y)$  exists in the network provided  $\epsilon_x^y = 0$  OR  $THR_x(y) > \epsilon_x^y$ .

The following definition will be used repeatedly throughout this paper:

**Definition 1.** An edge  $(x, y)$  is connected if  $\epsilon_x^y = 0$  OR  $THR_x(y) > \epsilon_x^y$ , and disconnected otherwise.

Therefore, the *strategy* of a node  $x$  is determined by the vector  $(\mathbf{s}^x, \mathbf{THR}_x, \mathbf{f}^x)$ . At this point we need to make the following observation<sup>4</sup> that is important for the communication bit complexity of the protocol: The number of the  $s^x$  variables that  $x$  must decide can be as large as the number of paths in the network, i.e., it grows exponentially on the size of the network. But, in fact, for what follows, it is enough for every  $x$  to determine *edge* flows  $s_{(x,e,y)}^x$  for all O-D pairs  $(x, y)$  and all edges  $e \in E$ . As is well known, edge flows may correspond to more than one path flow patterns, and we may have many Nash equilibria with the same edge flows but different path flows. In the following discussion, any of these patterns can be used as the path flows interchangeably, i.e., our results are *independent* of the path flow pattern and they depend only on the edge flows. This is easy to see in most cases (e.g., it is easy to see that the same amount of flow is unsuccessful in all path patterns, so Definition 3 below can be applied to each one of them with the same results for our paper), and we explain it a little more when we describe linear programs (LP-S) and (LP-C). Hence, every node/player  $x$  needs to decide the values of  $O(nm)$  numbers for  $s_{(x,e,y)}^x$ ,  $O(n)$  numbers for  $THR_x(y)$ , and  $O(n^4)$  numbers for  $f_{(u,e,e',v)}^x$ , or  $O(n^4)$  values overall, which is polynomial on the size of the network  $n = |V|$ ,  $m = |E|$ . For purposes of clarity, in what follows we revert to the path variables  $s^x$  as defined above, although this representation is extremely redundant.

Note that the routing of the flow node  $x$  sends out is incorporated in the values for  $\mathbf{s}^x$ . Therefore  $x$  decides the following:

- Threshold  $THR_x(y) \geq 0$ , and hence decides whether edge  $(x, y)$  is connected or not.
- Variables  $\epsilon_w^x$ , by deciding  $f_{(u,e,e',v)}^x$  which, in turn, change the flows  $F_{(u,e,e',v)}^x$ . As a result,  $x$  decides whether edge  $e = (w, x)$  is connected or not.
- The routing of the flow originating at  $x$  and its quantity, by deciding  $s_{(x,P,y)}^x$  for any path  $P$  connecting  $x$  to  $y$ . But always  $\sum_P s_{(x,P,y)}^x \leq d_{(x,y)}$ .

We repeat that every node sees all decision variables of all other nodes, we don't assume any kind of synchronization amongst the nodes, but we do assume that the decision variables changes are instantaneous.

<sup>4</sup> We thank an anonymous reviewer for helping to clarify this.

**Definition of the utility function:** Every node plays in a selfish way, i.e., so that its utility (defined below) is maximized. At any time  $t$ , we denote by  $C_y^-, C_y^+, D_y^-, D_y^+$  the sets of connected incoming, connected outgoing, disconnected incoming and disconnected outgoing edges respectively, adjacent to node  $y$ . Then, for every node  $y$  its utility function is defined as follows:

$$util_t(y) = \begin{array}{c} \text{flow sent by } y \\ \text{and reached its} \\ \text{destination} \end{array} + \begin{array}{c} \text{flow received} \\ \text{by } y \end{array} - \begin{array}{c} \text{flow forwarded} \\ \text{by } y \end{array} - \begin{array}{c} \text{flow sent by } y \\ \text{and didn't reach} \\ \text{its destination} \end{array}.$$

More specifically,

$$util_t(y) = \sum_{e \in C_y^+} S_e^y + \sum_{e \in C_y^-} R_e^y - \sum_{e' \in C_y^+ \cup D_y^+} \sum_{u \neq y, v \in G} I_{(u, e', v)}^y - \left( \sum_{e' \in C_y^+ \cup D_y^+} \sum_{v \in G} I_{(y, e', v)}^y - \sum_{e \in C_y^+} S_e^y \right) \quad (1)$$

where

- $S_e^y$  is the flow which has been sent by  $y$  (i.e. originated at  $y$ ) through edge  $e$  and has reached its destination,
- $R_e^y$  is the flow which has been received by  $y$  through edge  $e$ ,
- $I_{(u, e', v)}^y$  is the flow of commodity  $(u, v)$  with  $y \neq v$ , and node  $y$  attempts to forward (or sent, if  $u = y$ ) through edge  $e'$  (note that  $e'$  may be disconnected).

The intuition behind this definition of utility (which is very similar to the definition used in [1]), is that a node exchanges resource units (e.g., battery energy) for information units (i.e., packets received or sent successfully). Our assumption is that the correspondence is one for one. Different weighting of resources and information is a generalization left for future work.

Throughout this work, we use the standard definition of Nash equilibria, i.e., at equilibrium, no node gains an increase of its utility by changing its decision variables (strategy), while the other nodes maintain their own strategies. We will focus on *non-trivial* equilibria.

**Definition 2.** A trivial equilibrium is any equilibrium with  $\mathbf{f}^x = \mathbf{0}, \forall x$ , and with  $s_{(u, (u, v), v)}^u = d_{(u, v)}, \forall$  commodities  $(u, v)$  s.t.  $(u, v) \in E$  and  $s_{(u, P, v)}^u = 0$  otherwise.

So from now on, whenever we write ‘equilibrium’ we mean ‘non-trivial equilibrium’, unless otherwise stated. We also assume that there is always at least one demand between non-adjacent nodes in  $G$ , since otherwise a trivial equilibrium is a connected one, and this case is not very interesting.

**Definition 3.** An amount of flow with origin a node  $u$  and destination a node  $v$  routed through a path  $P$  is successful if it reaches node  $v$ , otherwise it is unsuccessful.

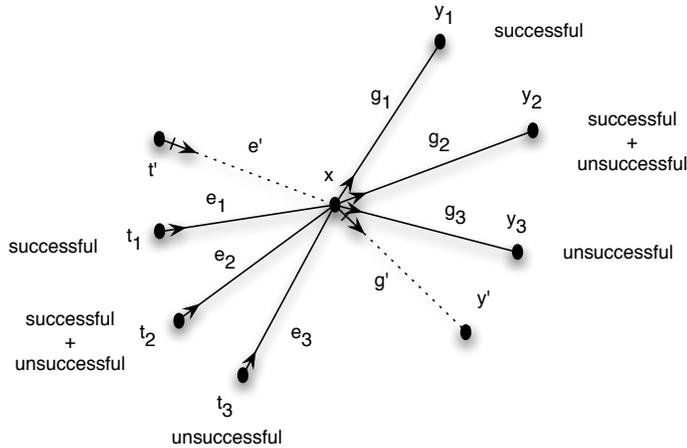


Fig. 1. A node  $x$  in the network. The groups of edges  $e', g'$  are disconnected.

### 3 Characterization of Nash equilibria

In this section we give necessary and sufficient conditions for the existence of an equilibrium. Our hope will be that these conditions (probably together with additional ones) will simplify the study of connected equilibria.

**Definition 4.** An unsuccessful flow  $\Phi$  which has been routed through a (disconnected) edge  $e$  is responsible for disconnecting edge  $e$  if  $e$  would be connected without  $\Phi$ .

We group the (non-disconnected) incoming and outgoing edges for a node  $x$  as follows (see Figure 1):

- group 1: these edges transfer only successful flows,
- group 2: these edges transfer successful and unsuccessful flows,
- group 3: these edges transfer only unsuccessful flows.

**Lemma 1.** Consider a network at a Nash equilibrium. Suppose that there is an edge  $e = (x, y)$  which is disconnected. Then a) there is at least one unsuccessful flow  $\Phi$  routed through  $e$  (and reaches node  $x$ ) which is responsible for disconnecting edge  $e$  and, b) every unsuccessful flow routed through  $e$  (and reaches node  $x$ ) is responsible for disconnecting edge  $e$ .

*Proof.* Since edge  $e = (x, y)$  is disconnected it holds  $\epsilon_x^y > 0$ . Therefore there is at least one unsuccessful flow  $\Phi$  which has been routed through  $e$  (and reaches node  $x$ ). If  $\Phi$  is the only such unsuccessful flow, then obviously  $\Phi$  is responsible for disconnecting  $e$ . Suppose there is another unsuccessful flow  $\Phi_1$  which has been routed through  $e$  (and reaches node  $x$ ). If  $\Phi_1$  is not responsible for disconnecting  $e$ , then the node  $u_1$  which sends  $\Phi_1$  could stop sending it and clearly increase its utility since nothing else changes on the network that could affect  $u_1$ 's utility. But this would mean that the network is not at a Nash equilibrium.  $\square$

**Theorem 1.** *The game is at an equilibrium if and only if for any node  $x$  the following conditions hold:*

1.  $\epsilon_x^y = 0$ , where  $g = (x, y) \in C_x^+$  (i.e., node  $y$  does not cut any flow forwarded by  $x$  through the connected edge  $g$ ),
2. if there is a successful flow between nodes  $u, v \neq y$  routed through edge  $g = (x, y)$ , then  $THR_x(y) = 0$ ,
3. if there is no unsuccessful flow going through an edge  $e = (t, x)$ , then  $R_e^x \geq \sum_{u \neq x, v} \sum_{g=(x, y)} F_{(u, e, g, v)}^x$   
(i.e., the flow that node  $x$  receives through edge  $e = (t, x)$  is not less than the total flow which is coming through  $e$  and  $x$  has to forward, if all this latter flow is successful),
4. for any disconnected edge  $g' = (x, y') \in D_x^+$  it holds that  $THR_x(y') = \epsilon_x^{y'} > 0$ , node  $x$  does not send any flow through  $g'$ , and the (unsuccessful) flows which are responsible for disconnecting  $g'$  are being sent by at least two nodes, other than  $x$ ,
5. let  $e = (t, x)$  be an incoming connected edge to  $x$  such that all unsuccessful flows which pass through  $e$ , have been routed through outgoing disconnected edges  $g' = (x, y'_i) \in D_x^+$  of  $x$ ; then:
  - $THR_t(x) = 0$ ,
  - $R_e^x \geq \sum_{u \neq x, v} \sum_{g' \in D_x^+} F_{(u, e, g', v)}^x + \sum_{u \neq x, v} \sum_{g \in C_x^+} F_{(u, e, g, v)}^x$ ,
6. the flow that node  $x$  sends successfully through all of its (connected) outgoing edges  $\Phi(x)$  is maximized over all possible routings  $\mathbf{s}^x$ ,
7. any combination of the following possible actions taken by  $x$  cannot increase its utility:
  - (a) disconnecting a number of edges of group 2,
  - (b) decreasing the unsuccessful flow that  $x$  lets go through edges of group 3,
  - (c) connecting edges  $e' = (t', x) \in D_x^-$ ,
  - (d) sending successful and unsuccessful flow through the outgoing edges of  $x$ ,
  - (e) increasing thresholds

*Proof.* We first prove that if the game is at a Nash equilibrium then the above conditions are true.

For the first condition, if  $\sum_{u, v \neq y} I_{(u, g, v)}^x = 0$ , i.e., if no flow is being forwarded from  $x$  to  $y$  through edge  $g$  with destination different than  $y$ , then, of course, nothing can be cut by  $y$  and thus  $\epsilon_x^y = 0$ . Now suppose that  $\epsilon_x^y > 0$  when  $\sum_{u, v \neq y} I_{(u, g, v)}^x > 0$ . This means that there is an unsuccessful flow sent by some node  $u$  to some node  $v$  through edge  $g$ , and this flow is blocked by  $y$ . But then node  $u$  can increase its utility by not sending this flow, which contradicts the definition of a Nash equilibrium. Therefore  $\epsilon_x^y = 0$ .

For the second condition, if  $THR_x(y) > \epsilon_x^y = 0$  when there is a successful flow  $\Phi$  through  $g = (x, y)$ , then node  $y$  could increase its utility by cutting flow forwarded by  $x$ , thus increasing  $\epsilon_x^y$  up to  $\min\{THR_x(y) - \delta, \Phi\}$ , for some  $\delta > 0$  without violating  $THR_x(y) > \epsilon_x^y = 0$  which is a contradiction.

For condition (3), if  $R_e^x < \sum_{u \neq x, v} \sum_{g=(x, y)} F_{(u, e, g, v)}^x$ , then  $x$  could profit by cutting all flow coming through  $e$ .

For condition (4), first we prove that  $x$  cannot send all the unsuccessful flow responsible for disconnecting  $g'$  by itself. Indeed, suppose that  $x$  sends all the unsuccessful flow  $\Phi$  that causes the disconnection of  $g'$  (due to the fact that for node  $y'$  we have  $THR_x(y') \leq \epsilon_x^{y'}$  and  $\epsilon_x^{y'} > 0$ ). Since the system is at a Nash equilibrium, no other node sends something through  $g'$  (if there was a node which was sending a flow through  $g'$  then it would be a clear profit to its utility to stop sending that flow). Thus  $x$  can safely connect edge  $g'$  by not sending the flow  $\Phi$  and profit, which is a contradiction. For the same reason any node  $t$  cannot send all by itself the unsuccessful flow  $\Phi$  that disconnects  $g'$ . Thus there are at least two nodes which send unsuccessful flows responsible for the disconnection of  $g'$ . Suppose that one of them is  $x$ . Now if  $THR_x(y') < \epsilon_x^{y'}$  then at least one of these two flows can be safely decreased without connecting  $g'$ , a contradiction. Hence  $THR_x(y') = \epsilon_x^{y'} > 0$ . But then  $x$  can decrease it to  $THR_x(y') = 0$  and not send its flow, thus profiting, a contradiction. Therefore there are at least two nodes which send unsuccessful flows responsible for disconnecting  $g'$  and node  $x$  is not one of them.  $THR_x(y') = \epsilon_x^{y'} > 0$  for the same reason as before.

For condition (5), if  $THR_t(x) > 0$  for some  $t$ , then node  $x$  could increase  $\epsilon_t^x$ , thus decreasing  $\epsilon_x^{y'_i}$  for some  $i$  and at the same time decreasing  $THR_x(y'_i)$ , so that  $THR_x(y'_i) \leq \epsilon_x^{y'_i}$  still holds, thus profiting, a contradiction. If  $R_e^x < \sum_{u \neq x, v} \sum_{g' \in D_x^+} F_{(u, e, g', v)}^x + \sum_{u \neq x, v} \sum_{g \in C_x^+} F_{(u, e, g, v)}^x$  for some  $e$  and given that  $THR_t(x) = 0$ , node  $x$  could profit by increasing  $\epsilon_t^x$  and cutting edge  $e$ , a contradiction.

Finally, the proof for conditions (6), (7) is straight-forward, since if they do not hold then node  $x$  can have a course of action that increases its utility, a contradiction.

Now we prove that if the above conditions hold, then no node can increase its utility by changing something and thus the game is at a Nash equilibrium.

For the sake of contradiction, suppose that there is a node  $x$  which can increase its utility. This means that  $x$  succeeds to change one or a combination of the following and this yields to an increase in its utility:

- (i) increase the flow that sends successfully, or
- (ii) increase the flow that receives, or
- (iii) decrease the flow that forwards, or
- (iv) decrease the unsuccessful flow that sends.

We will show that when a node manages to change any of the above, its utility function does not increase, no matter what other actions it may take.

For case (i), since (6) holds, node  $x$  has to connect at least one previously disconnected edge. Such an edge will be either adjacent to  $x$  or not. First suppose it is an adjacent edge  $g' = (x, y')$ . For connecting  $g'$ , node  $x$  should either increase  $THR_x(y')$  or increase  $\epsilon_t^x$  of an edge  $e = (t, x)$  through which an unsuccessful flow passes that cuts edge  $g'$ . In view of conditions (5), (7)b, (7)d, (7)e both these cases lead to a loss for  $x$ . Another case for  $x$  is to send an unsuccessful flow to cut an edge  $w$  which will end up (possibly after connecting and disconnecting several edges) to cut an edge which is crossed by the unsuccessful flow which cuts  $g'$ . This is also non-profitable because of condition (7)c, (7)d. Now consider a disconnected edge  $g'$  which is not adjacent to  $x$ . Either  $x$  has to send less unsuccessful flow

through  $g'$ , or increase an  $\epsilon_t^x$  of an edge  $e = (t, x)$  through which there is an unsuccessful flow that cuts edge  $g'$  or to send an unsuccessful flow to cut an edge  $w$  which will end up (possibly after connecting and disconnecting several edges) to cut an edge which is crossed by the unsuccessful flow which cuts  $g'$ . Again all these are not profitable because of condition (7)c, (7)d, (7)b.

For case (ii), the only case in which node  $x$  could receive more flow than before is by connecting a previously disconnected not-adjacent edge  $e'_x$  through which there was an unsuccessful flow with target  $x$ . For this purpose, node  $x$  has to send an unsuccessful flow to cut an edge  $w$  which will end up (possibly after connecting and disconnecting several edges) to cut an edge which is crossed by the unsuccessful flow which cuts  $e'_x$ . This is not profitable because of condition (7)b, (7)c, (7)d.

For case (iii), i.e., node  $x$  succeeds to forward less flow than before, we note that  $x$  could do this either by decreasing some variables  $f_{(u,e,g,v)}^x > 0$ , where  $I_{(u,e,v)}^t > 0, v \neq x$ , or by sending an unsuccessful flow to cut an edge  $w$  which, in turn, and possibly after connecting and disconnecting several edges, will cut an edge crossed by the flow which  $x$  forwards.

- For the first case, and in view of conditions (1), (2), if any  $f_{(u,e,g,v)}^x > 0$  of a successful flow gets decreased,  $\epsilon_t^x$  will get increased, and edge  $e$  will be disconnected. But because of condition (3),  $x$ 's utility will decrease. If node  $x$  tries to decrease the unsuccessful flow that it forwards, then because of (4), (5), (7)a, (7)b this action is not profitable.
- The last remaining case for  $x$  is again not profitable because of conditions (7)a, (7)b, (7)c, (7)d.

For case (iv), condition (7)d implies that if  $x$  decreases the unsuccessful flow that was sending then it will be a loss.  $\square$

Theorem 1 is essentially a codification of all the conditions that happen simultaneously at equilibrium. But showing that such a (non-trivial) equilibrium exists (or, even more, compute it) is non-trivial. In fact, we will show that deciding the existence of an equilibrium is NP-hard. But it turns out it is much easier to check whether there is a non-trivial equilibrium with only successful flows; this can be reduced to the solution of a simple LP.

For every edge  $e = (u, v)$ , we set  $d(e)$  equal to  $d_{(u,v)}$  if commodity  $(u, v)$  exists, and 0 otherwise. Let  $D := \sum_{e \in E} d(e)$ . We will use the following notation:

- $e \in^* P$ , when edge  $e \in P$  is *not* the last edge of  $P$ ,
- $e \in^0 P$ , when edge  $e \in P$  is the last edge of  $P$ .

In the following LP, variables  $x(P)$  represent the amount of flow sent along path  $P$ :

$$\begin{aligned}
& \max \sum_{P \in \mathcal{P}} x(P) \quad \text{s.t.} & \text{(LP-S)} \\
& \sum_{P: e \in^* P} x(P) - \sum_{P: e \in^0 P} x(P) \leq 0 & \forall e \in E \\
& \sum_{P \in \mathcal{P}_i} x(P) \leq d_{(u_i, v_i)} & \forall i \\
& x(P) \geq 0 & \forall P \in \mathcal{P}
\end{aligned}$$

This linear program has an exponential number of variables, although the number of constraints is polynomial on the size of the input. We can replace it with an equivalent (in terms of having exactly the same optimal solution, and feasibility) and standard *edge flow* formulation that uses variables  $x_e^{(u_i, v_i)}$  for every edge  $e \in E$  and commodity  $i$  with O-D pair  $(u_i, v_i)$  to indicate the flow of commodity  $i$  that crosses edge  $e$ . Then the first group of constraints can be written as

$$\sum_{i:v \neq v_i} x_{(u,v)}^{(u_i, v_i)} - \sum_{i:v = v_i} x_{(u,v)}^{(u_i, v_i)} \leq 0, \quad \forall (u, v) \in E,$$

and the other constraints are replaced with the standard constraints of flow conservation and demand satisfaction. Obviously, the new formulation can be solved in polynomial time; if it is infeasible then the original formulation (LP-S) is also infeasible, otherwise its solution can be transformed into a path flow solution  $x(P)$  of the same total demand, with only polynomially (on  $|V|, |E|$ ) many non-zero  $x(P)$ 's. These non-zero flow paths are going to be used below.

**Theorem 2.** *A non-trivial equilibrium with only successful flows exists if and only if (LP-S) has a solution  $x(P)$  with  $\sum_{P \in \mathcal{P}} x(P) > D$ .*

*Proof.* First we note that the trivial equilibrium is a solution of (LP-S) (therefore (LP-S) is always feasible) that achieves an objective value of  $D$ . Also, note that if an origin-destination pair  $(u_i, v_i)$  is connected by edge  $e = (u_i, v_i)$  in  $E$ , then at any equilibrium the whole demand  $d_{(u_i, v_i)}$  is routed through  $e$ , otherwise  $u_i$  could have increased its utility by routing more of this commodity. Hence, the total flow routed by a non-trivial equilibrium with only successful flows is always greater than  $D$ .

For the 'if' direction, let  $x(P)$  be a solution to (LP-S). Consider an edge  $e = (x, y)$ . Notice that  $\sum_{P: e \in P} x(P) = \sum_{u, v, g} F_{(u, e, g, v)}^y$  and  $\sum_{P: e \in P} x(P) = R_e^y$ . Then the first constraint guarantees that  $\sum_{u, v, g} F_{(u, e, g, v)}^y \leq R_e^y$ . We call this property (i). We will show that the user strategies produced by the following procedure *Equil*<sup>5</sup> satisfy the theorem:

**Procedure Equil**

**Initialize**  $(s^x, THR_x, f^x) = \mathbf{0}, \forall x \in V;$

$s_{(u_i, P, v_i)}^{u_i} := x(P), \forall P \in \mathcal{P}_i$  with  $x(P) > 0, \forall i;$

**For each** commodity  $(u_i, v_i)$  **do**

$TotFlow(u_i, v_i) := 0;$

**For each** edge  $e_1 = (u_i, x_1)$  **do**

**For each** edge  $e_2 = (x_1, x_2)$  **do**

$$f_{(u_i, e_1, e_2, v_i)}^{x_1} := \sum_{P \in \mathcal{P}_i: x(P) > 0 \wedge (e_1, e_2 \in P)} s_{(u_i, P, v_i)}^{u_i};$$

$$TotFlow(u_i, v_i) := TotFlow(u_i, v_i) + f_{(u_i, e_1, e_2, v_i)}^{x_1};$$

**For each** edge  $e_j = (x_{j-1}, x_j)$  not incident to  $u_i, v_i$  **do**

**For each** edge  $e_{j+1} = (x_j, x_{j+1})$  not incident to  $u_i$  **do**

$$f_{(u_i, e_j, e_{j+1}, v_i)}^{x_j} := TotFlow(u_i, v_i);$$

<sup>5</sup> Note that the running time of *Equil* is polynomial in  $|V|, |E|$  and the number of commodities.

We prove that the above construction is already at equilibrium. Consider a commodity  $(u, v)$ . Notice that in any path  $\langle u \xrightarrow{e_1} x_1 \xrightarrow{e_2} x_2 \xrightarrow{e_3} \dots \xrightarrow{e_{k-1}} x_{k-1} \xrightarrow{e_k} v \rangle$  from  $u$  to  $v$  in the network, we have

$$\sum_{P \in \mathcal{P}(u,v): x(P) > 0 \wedge (e_1, e_2 \in P)} s_{(u,P,v)}^u = f_{(u,e_1,e_2,v)}^{x_1} \leq f_{(u,e_2,e_3,v)}^{x_2} = f_{(u,e_i,e_{i+1},v)}^{x_i} = \text{TotFlow}(u, v),$$

where  $3 \leq i \leq k-1$ . The total flow that can pass from  $e_i, e_{i+1}$  going from  $u$  to  $v$ , is at most  $f_{(u,e_i,e_{i+1},v)}^{x_i}$ . Even if all this flow passes also from  $e_j, e_{j+1}$  for  $j > i$ , it will not exceed  $f_{(u,e_j,e_{j+1},v)}^{x_j}$ . Therefore, if edge  $e_j = (x_{j-1}, x_j)$  is already connected ( $THR_{x_{j-1}}(x_j) > \epsilon_{x_{j-1}}^{x_j}$  OR  $\epsilon_{x_{j-1}}^{x_j} = 0$ ), it cannot get disconnected by an unsuccessful flow from  $u$  to  $v$  which has been routed through  $e_i, e_{i+1}, e_j, e_{j+1}$ , since  $\epsilon_{x_{j-1}}^{x_j}$  cannot increase. This means that there is no unsuccessful flow in the network and moreover, node  $u$  cannot send an unsuccessful flow using any path to  $v$ , cutting an edge  $e_i$ , where  $i > 1$ . We call this property (ii).

We verify that the conditions of Theorem 1 hold. Since there is no unsuccessful flow in the network, condition (1) holds. Condition (2) has been explicitly forced in the first line of the Procedure *Equil*. Conditions (4), (5) hold trivially since there is no unsuccessful flow in the network. Condition (3) also holds because of property (i). For condition (6), since no edge is disconnected, node  $u$  could send more flow to node  $v$  only by increasing either some term(s) of  $\sum_{P \in \mathcal{P}(u,v): x(P) > 0 \wedge (e_1, e_2 \in P)} s_{(u,P,v)}^u$  or some  $s_{(u,P,v)}^u$ , where  $e_1, e_2 \in P$  and  $x(P) = 0$ . In

the first case, edge  $e_1$  would be disconnected since  $\sum_{P \in \mathcal{P}(u,v): x(P) > 0 \wedge (e_1, e_2 \in P)} s_{(u,P,v)}^u = f_{(u,e_1,e_2,v)}^{x_1}$

and hence node  $u$  fails to send more flow to node  $v$ . In the second case, edge  $e_1$  would be again disconnected since in that case  $f_{(u,e_1,e_2,v)}^{x_1} = 0$ . In view of property (ii), condition (7) becomes equivalent to ‘sending any unsuccessful flow does not increase its utility’. Because of property (ii), the only unsuccessful flow that  $u$  can send, can cut only edge  $e_1$ . But this is clearly not profitable. Thus condition (7) also holds and the network is at equilibrium.

For the ‘only if’ direction, suppose that a (non-trivial) equilibrium with all flows being successful exists. Set  $x(P)$  to be the amount of flow through path  $P \in \mathcal{P}_i, \forall i$ . Obviously, the second constraint of (LP-S) is satisfied. Condition 3 of Theorem 1 implies that  $\sum_{u,v,g} F_{(u,e,g,v)}^y \leq$

$R_e^y$ , for every edge  $e$ . But  $\sum_{u,v,g} F_{(u,e,g,v)}^y = \sum_{P: e \in P} x(P)$  and  $R_e^y = \sum_{P: e \in P} x(P)$ . Therefore the

first constraint of (LP-S) holds as well. Hence,  $x(P)$  is feasible, and since the equilibrium is non-trivial,  $\sum_{P \in \mathcal{P}} x(P) > D$  holds, as explained at the beginning.  $\square$

The solution of (LP-S) by standard techniques [4] implies the following:

**Corollary 1.** *We can compute in polynomial time user strategies that are at equilibrium with only successful flows, if such an equilibrium exists.*

## 4 Connected equilibria

In this section we study the following question: given an underlying network topology along with a set of demands between nodes, is it possible to assign values to the decision variables, so that the game converges to a connected equilibrium, when such an equilibrium exists?

Recall that we call the network *connected* iff a non-zero amount of every commodity reaches its destination. Therefore, if, in addition to being at equilibrium, we want the network to be connected, we have to add to Theorem 1 the condition that for every commodity  $(u, v)$ , there is a successful non zero flow sent from  $u$  to  $v$  through a path  $P$  in the network. This translates to the following condition for every edge  $e = (x, y)$  in path  $P$ :  $THR_x(y) \geq \epsilon_x^y = 0$  AND  $I_{(u,e,v)}^x > 0$  (especially when  $y \neq v$ , it must hold  $THR_x(y) = \epsilon_x^y = 0$ , as follows from condition 2 of Theorem 1).

**Theorem 3.** *A network is at a connected equilibrium if and only if in addition to the Theorem 1 conditions, for every commodity  $(u, v)$ , either edge  $(u, v)$  is connected or there is a path connecting  $u, v$ , so that for every edge  $e = (x, y)$  in the path it holds that  $I_{(u,e,v)}^x > 0$  AND  $\epsilon_x^y = 0$ .*

It is easy to see that there are cases in which it is impossible for a game to converge to a connected equilibrium. For example, suppose that there is an edge  $e = (x, y)$  in the network such that node  $x$  is neither a source nor a sink, and there is a commodity  $(u, v)$  such that all paths between  $u$  and  $v$  pass through  $e$ . Then it is easy to see that, in any equilibrium, there will be no flow from  $u$  to  $v$ . Indeed, suppose that there is a connected equilibrium. Hence there should be an edge  $e = (t, x)$  in the network which carries some successful flow. If  $e$  carries *only* successful flow then the condition 3 of Theorem 1 would be violated. On the other hand if  $e$  carries successful *and* unsuccessful flow condition 7(a) would be violated since  $x$  would have a profit to disconnect edge  $e$  and gain in its utility.

As mentioned in the Introduction, the proof of existence, and the computation of strategies that lead to connected equilibria is, in general, very difficult, since we will prove in the next section that it is an NP-hard problem. But, building on the results of the previous section, we can prove the existence (or not) of a connected equilibrium with only successful flows in polynomial time, and compute strategies that achieve it. Using the characterization of such equilibria by Theorem 3, we can reduce this computation to the solution of the following extension of (LP-S):

$$\begin{aligned}
 & \max \quad w && \text{s.t.} && \text{(LP-C)} \\
 & \sum_{P:e \in P} x(P) - \sum_{P:e \in 0P} x(P) \leq 0 && \forall e \in E \\
 & \sum_{P \in \mathcal{P}_i} x(P) \leq d_{(u_i, v_i)} && \forall i \\
 & \sum_{P \in \mathcal{P}_i} x(P) \geq w && \forall i \\
 & x(P) \geq 0 && \forall P \in \mathcal{P} \\
 & w \geq 0
 \end{aligned}$$

The polynomial time solvability of (LP-C) can be shown in the same way as for (LP-S).

Similarly to Theorem 2, we can prove the following

**Theorem 4.** *A connected equilibrium with only successful flows exists if and only if (LP-C) has a solution  $x(P), w$  with  $w > 0$ .*

Again, the solution of (LP-C) by standard techniques [4] implies the following

**Corollary 2.** *We can compute in polynomial time user strategies that induce a connected equilibrium with only successful flows, if such an equilibrium exists.*

## 5 NP-hardness of Existence of a Connected Nash Equilibrium

Suppose a network is given together with a set of demands. In this section we prove that it is NP-hard to decide whether there exists a connected equilibrium, i.e., whether there exist values for the decision variables of the nodes so that the game converges to an equilibrium with the network being connected. We prove this by showing a reduction from the satisfiability problem (SAT).

**Sketch of the reduction:** Given an instance  $I$  of the SAT problem we construct (in polynomial time on the number of the boolean variables) a network and a set of demands between nodes. The basic element of the construction is the *variable-subgraph* (Figure 2) which corresponds to a boolean variable of  $I$  and it is constructed in such a way, so that in any connected Nash equilibrium, in exactly one of its edges there is no successful flow at all. We then show that there is a truth assignment which satisfies an instance  $I$  of SAT if and only if there is a Nash equilibrium in the constructed network with the network being connected (i.e., for any demand there is a flow being delivered). We prove this by giving explicitly values to decision variables of the nodes so that the network is connected at a Nash equilibrium. We show that if a boolean variable  $A$  has value *FALSE* in the truth assignment and appears as  $\neg A$  in a literal of  $I$  (negative literal) then the corresponding subgraph (Figure 10) is connected at a Nash equilibrium with only successful flows. If variable  $A$  has value *TRUE* in the truth assignment and appears as  $A$  in a literal of  $I$  (positive literal) then the corresponding subgraph (Figure 9) is connected at a Nash equilibrium with successful and unsuccessful flows.

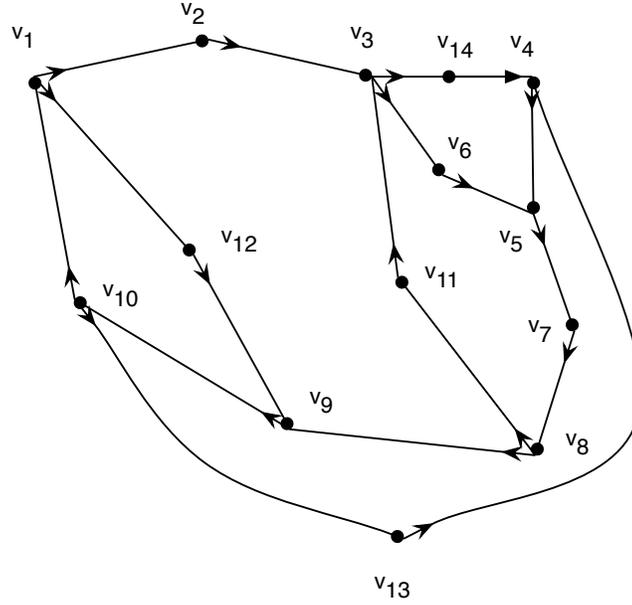
### 5.1 Construction of the reduction

Let  $I$  be an instance of the SAT problem. We remind the reader that a literal  $L(A)$  in  $I$  of the SAT problem is the appearance of the boolean variable  $A$  in its positive ( $A$ ) or negative ( $\neg A$ ) form. A clause  $C$  in  $I$  is a disjunction of literals and the instance  $I$  is a conjunction of clauses.

For every variable  $A \in I$  we construct a *variable-subgraph* as shown in Figure 2. We select numbers  $\delta, \alpha, \lambda$  where  $\delta \gg \alpha \gg \lambda > 0$  and we assign demands between nodes in that subgraph as illustrated in Figure 3.

For every clause  $C \in I$  we add two nodes  $u_C, v_C$ . We also assign a demand  $\delta$  between  $u_C$  and  $v_C$ . We connect these nodes with the variable-subgraphs described earlier as follows:

- If variable  $A$  appears as literal  $A$  (positive form) in clause  $C$  then we add the edges  $(u_C, v_1)$  and  $(v_2, v_C)$  and we assign a demand  $\delta$  between nodes  $u_C, v_1$  (see Figure 4a).
- If variable  $A$  appears as literal  $\neg A$  (negative form) in clause  $C$  then we add the edges  $(u_C, v_4)$  and  $(v_5, v_C)$  and we assign a demand  $\delta$  between nodes  $u_C, v_4$  (see Figure 4b).



**Fig. 2.** A variable-subgraph.

We call these subgraphs *literal-subgraphs*.

The above construction can be done in polynomial time on the number of the boolean variables. An example of a clause with 3 literals and its respective construction, which we call a *clause-subgraph* is shown in Figure 5. In Figure 6, an example of a boolean formula with 3 clauses-3 variables and its respective construction is shown.

## 5.2 Transformation of a truth assignment

We will now see how to transform in polynomial time a solution of the SAT problem to a solution of our problem. The general idea is the following:

- If a variable  $A$  of the SAT problem has value *TRUE*, then we set suitable values to decision variables of the corresponding variable-subgraph so that to force node  $v_2$  to forward some successful flow. To that end node  $v_1$  and node  $v_3$  send unsuccessful flows which disconnect edge  $(v_4, v_5)$ . In particular node  $v_1$  sends an unsuccessful flow through nodes  $\langle v_1, v_2, v_3, v_{14}, v_4, v_5, v_7, v_8, v_9 \rangle$  to node  $v_9$  and node  $v_3$  sends an unsuccessful flow through nodes  $\langle v_3, v_{14}, v_4, v_5, v_7, v_8, v_9 \rangle$  to node  $v_9$ . These flows are such that if one of them decreases, then edge  $(v_4, v_5)$  gets connected again. Moreover, in such a case (i.e., should edge  $(v_4, v_5)$  be connected again), the flow that would pass through edge  $(v_4, v_5)$  would disconnect edge  $(v_5, v_7)$ . Because of this, node  $v_2$  would stop sending a flow to node  $v_9$ . In other words, by this technique, node  $v_2$  will be forced to forward the unsuccessful flow from node  $v_1$  and keep edge  $(v_1, v_2)$  connected. We can then pass a successful flow through edge  $(v_1, v_2)$ , namely the flow which satisfies the demand between nodes  $u_c, v_c$  which nodes correspond to the clause in which the variable  $A$  appears in its

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1									$\delta$			$\delta$		
2			$3\delta + 4\alpha$						$2\delta + \alpha$					
3					$\delta$	$4\delta + 2\alpha$			$2\alpha$					$2\alpha$
4														
5	$\delta + 2\alpha$		$\delta$				$4\delta + 4\alpha$							
6					$2\delta + 3\alpha$									
7								$4\delta + 4\alpha$						
8									$\delta$		$\delta$			
9										$\delta + 2\alpha$				
10	$\delta$				$\delta$									$\delta$
11			$\delta$											
12									$\delta$					
13				$\delta$										
14				$2\alpha$										

**Fig. 3.** Demands for the nodes of the variable-subgraph in Figure 2. The rows contain the sources and the columns contain the sinks. For example the box at row 3 and column 9 contains the value  $2\alpha$  which means that the demand from node  $v_3$  to node  $v_9$  is  $2\alpha$ . Empty boxes represent zero demands between the corresponding nodes.

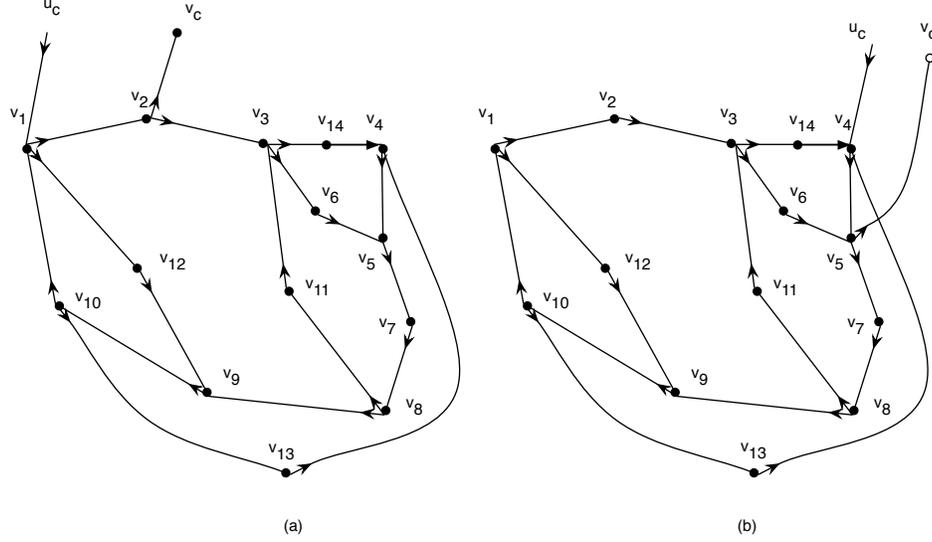
positive form (if there exists such a clause). In this case edge  $(v_4, v_5)$  is the only one in the variable-subgraph with no successful flow at all.

- If a variable  $A$  of the SAT problem has value *FALSE*, then we set suitable values to decision variables of the corresponding variable-subgraph so that there is a successful flow routed through edge  $(v_4, v_5)$ , namely the flow which satisfies the demand between nodes  $u_c, v_c$  which nodes correspond to the clause in which the variable  $A$  appears in its negative form (if there exists such a clause). In this case there are no unsuccessful flows in the network and edge  $(v_1, v_2)$  is the only one in the variable-subgraph with no successful flow at all.
- We then show that the network is connected (i.e., for any demand there is a flow being delivered) and the system is at a Nash equilibrium.

We now set the values of the decision variables of the nodes of the constructed graph. We are doing so by first setting the decision variables of the nodes of each variable-subgraph. For simplicity, we use instead of notation  $v_i$ , just the number  $i$  (i.e.,  $i$  stands for node  $v_i$ ). We first list the values for the nodes decision variables of a variable-subgraph  $G_A$  which are common in every variable-subgraph (no-matter whether the corresponding boolean variable  $A$  has been assigned a value *TRUE* or *FALSE*) and then we complete the initialization for those two cases. We only list the decision variables to which we assign non-zero values (i.e., for any decision variable not listed below, we assign a zero value). We assign the values as follows:

**Common values:**

- Node 1:  $f_{(10,(10,1),(1,2),5)}^1 = \delta, s_{\langle 1,12 \rangle}^1 = \delta$ .
- Node 2:  $s_{\langle 2,3 \rangle}^2 = 3\delta + 4\alpha, s_{\langle 2,3,6,5,7,8,9 \rangle}^2 = 2\delta + \alpha$ .
- Node 3:  $f_{(10,(2,3),(3,6),5)}^3 = \delta, f_{(1,(2,3),(3,14),9)}^3 = \alpha, f_{(2,(2,3),(3,6),9)}^3 = 2\delta + \alpha, s_{\langle 3,6 \rangle}^3 = 4\delta + 2\alpha, s_{\langle 3,6,5 \rangle}^3 = \delta, s_{\langle 3,14 \rangle}^3 = 2\alpha, s_{\langle 3,6,5,7,8,9 \rangle}^3 = \alpha$ .
- Node 5:  $f_{(10,(6,5),(5,7),9)}^5 = 2\delta + \alpha, f_{(3,(6,5),(5,7),9)}^5 = \alpha, s_{\langle 5,7,8,9,10,1 \rangle}^5 = \delta + 2\alpha, s_{\langle 5,7,8,11,3 \rangle}^5 = \delta, s_{\langle 5,7 \rangle}^5 = 4\delta + 4\alpha$ .



**Fig. 4.** In any literal-subgraph the demand between nodes  $u_c$ ,  $v_c$  is  $\delta$ . a) A positive literal-subgraph. The demand between nodes  $u_c$ ,  $v_1$  is  $\delta$ . b) A negative literal-subgraph. The demand between nodes  $u_c$ ,  $v_4$  is  $\delta$ .

- Node 6:  $f_{(2,(3,6),(6,5),9)}^6 = 2\delta + \alpha$ ,  $f_{(3,(3,6),(6,5),5)}^6 = \delta$ ,  $f_{(3,(3,6),(6,5),9)}^6 = \alpha$ ,  $f_{(10,(3,6),(6,5),5)}^6 = \delta$ ,  $s_{(<6,5>)}^6 = 2\delta + 3\alpha$ .
- Node 7:  $f_{(2,(5,7),(7,8),9)}^7 = 2\delta + \alpha$ ,  $f_{(3,(5,7),(7,8),9)}^7 = \alpha$ ,  $f_{(5,(5,7),(7,8),3)}^7 = \delta$ ,  $f_{(5,(5,7),(7,8),1)}^7 = \delta + 2\alpha$ ,  $s_{(<7,8>)}^7 = 4\delta + 4\alpha$ .
- Node 8:  $f_{(2,(7,8),(8,9),9)}^8 = 2\delta + \alpha$ ,  $f_{(3,(7,8),(8,9),9)}^8 = \alpha$ ,  $f_{(5,(7,8),(8,9),1)}^8 = \delta + 2\alpha$ ,  $f_{(5,(7,8),(8,11),3)}^8 = \delta$ ,  $s_{(<8,9>)}^8 = \delta$ ,  $s_{(<8,11>)}^8 = \delta$ .
- Node 9:  $f_{(5,(8,9),(9,10),1)}^9 = \delta + 2\alpha$ ,  $s_{(<9,10>)}^9 = \delta + 2\alpha$ .
- Node 10:  $f_{(5,(9,10),(10,1),1)}^{10} = \delta + 2\alpha$ ,  $s_{(<10,1>)}^{10} = \delta$ ,  $s_{(<10,13>)}^{10} = \delta$ .
- Node 11:  $f_{(5,(8,11),(11,3),3)}^{11} = \delta$ ,  $s_{(<11,3>)}^{11} = \delta$ .
- Node 12:  $f_{(1,(1,12),(12,9),9)}^{12} = \delta$ ,  $s_{(<12,9>)}^{12} = \delta$ .
- Node 13:  $f_{(10,(10,13),(13,4),5)}^{13} = \delta$ ,  $s_{(<13,4>)}^{13} = \delta$ .
- Node 14:  $f_{(1,(3,14),(14,4),9)}^{14} = \alpha$ ,  $s_{(<14,4>)}^{14} = 2\alpha$ .

According to whether the boolean variable  $A$  has value *TRUE* or *FALSE* we complete the initialization as follows:

If variable  $A$  of the SAT problem has value *TRUE* then we complete the *common values* with the values of the following decision variables.

#### Additional values for *TRUE* value transformation

- Node 1:  $s_{(<1,2,3,14,4,5,7,8,9>)}^1 = \alpha$ ,  $s_{(<1,12,9>)}^1 = \delta - \alpha$ .
- Node 2:  $f_{(10,(1,2),(2,3),5)}^2 = \delta$ ,  $f_{(1,(1,2),(2,3),9)}^2 = \alpha$ .
- Node 3:  $s_{(<3,14,4,5,7,8,9>)}^3 = \alpha$ .
- Node 4:  $THR_4(5) = 2\lambda$ ,  $f_{(1,(14,4),(4,5),9)}^4 = \alpha$ ,  $f_{(3,(14,4),(4,5),9)}^4 = \alpha$ .
- Node 5:  $f_{(3,(4,5),(5,7),9)}^5 = \alpha - \lambda$ ,  $f_{(1,(4,5),(5,7),9)}^5 = \alpha - \lambda$ .

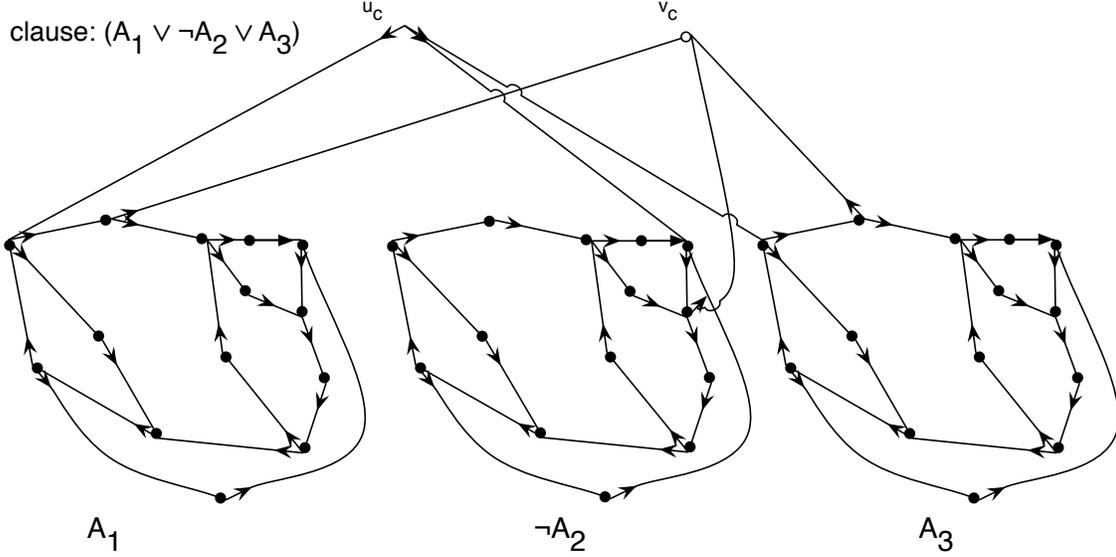


Fig. 5. A clause  $C = A_1 \vee \neg A_2 \vee A_3$  and its respective constructed clause-subgraph.

- Node 10:  $s_{(\langle 10,1,2,3,6,5 \rangle)}^{10} = \delta$ .
- Node 14:  $f_{(3,(3,14),(14,4),9)}^{14} = \alpha$ .

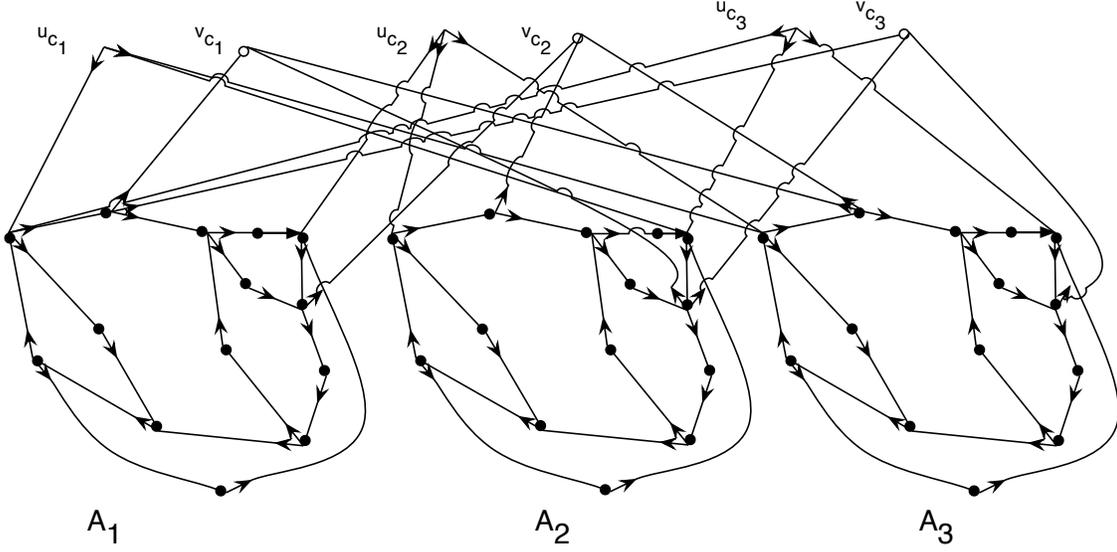
The above values together with the *common values* consist the **TRUE value transformation**. In this case, where the variable-subgraph corresponds to a boolean variable with value *TRUE*, the flow-paths have been routed as shown in Figure 7.

**Lemma 2.** Consider a stage where the decision variables of the nodes of a variable-subgraph have the values of the **TRUE value transformation**. Then: a) edge  $(v_4, v_5)$  is disconnected and b) if one of the (unsuccessful) flows which have been routed through edge  $(v_4, v_5)$  decreases then edge  $(v_5, v_7)$  will get disconnected.

*Proof.* Node  $v_4$  forwards  $\alpha$  unsuccessful flow coming from  $v_1$  ( $s_{(\langle 1,2,3,14,4,5,7,8,9 \rangle)}^1 = f_{(1,(1,2),(2,3),9)}^2 = f_{(1,(2,3),(3,14),9)}^3 = f_{(1,(3,14),(14,4),9)}^{14} = f_{(1,(14,4),(4,5),9)}^4 = \alpha$ ) and  $\alpha$  unsuccessful flow coming from  $v_3$  ( $s_{(\langle 3,14,4,5,7,8,9 \rangle)}^3 = f_{(3,(3,14),(14,4),9)}^{14} = f_{(3,(14,4),(4,5),9)}^4 = \alpha$ ). Node  $v_5$  allows  $\alpha - \lambda$  flow from  $v_1$  ( $f_{(1,(4,5),(5,7),9)}^5 = \alpha - \lambda$ ) and  $\alpha - \lambda$  flow from  $v_3$  coming from edge  $(4, 5)$  ( $f_{(3,(4,5),(5,7),9)}^5 = \alpha - \lambda$ ). Therefore  $\epsilon_4^5 = 2\lambda$  (node  $v_4$  does not forward any other flow). Since  $THR_4(5) = 2\lambda$ , we have that  $THR_4(5) = \epsilon_4^5 = 2\lambda$ . Therefore edge  $(v_4, v_5)$  is disconnected. If one of these two unsuccessful flows gets decreased, then we will have  $\epsilon_4^5 < 2\lambda$  and  $\epsilon_4^5 < THR_4(5)$  which means that edge  $(v_4, v_5)$  will get connected. Then the other flow  $(\alpha - \lambda)$  will go through edge  $(v_5, v_7)$ . Notice that there is already  $\alpha$  successful flow coming from node  $v_3$  which node  $v_7$  forwards. Since  $f_{(1,(5,7),(7,8),9)}^7 = 0$  and  $f_{(3,(5,7),(7,8),9)}^7 = \alpha$ , we will have that  $\epsilon_5^7 \geq \alpha - \lambda > 0$  and since  $THR_5(7) = 0$ , we will have  $THR_5(7) < \epsilon_5^7$  which means that edge  $(v_5, v_7)$  gets disconnected.  $\square$

If variable  $A$  of the SAT problem has value *FALSE* then we complete the *common values* with the values of the following decision variables.

A boolean formula:  $(A_1 \vee \neg A_2 \vee A_3) \wedge (\neg A_1 \vee A_2 \vee A_3) \wedge (A_1 \vee \neg A_2 \vee \neg A_3)$



**Fig. 6.** A boolean formula  $(A_1 \vee \neg A_2 \vee A_3) \wedge (\neg A_1 \vee A_2 \vee A_3) \wedge (A_1 \vee \neg A_2 \vee \neg A_3)$  and its respective constructed subgraph.

### Additional values for *FALSE* value transformation

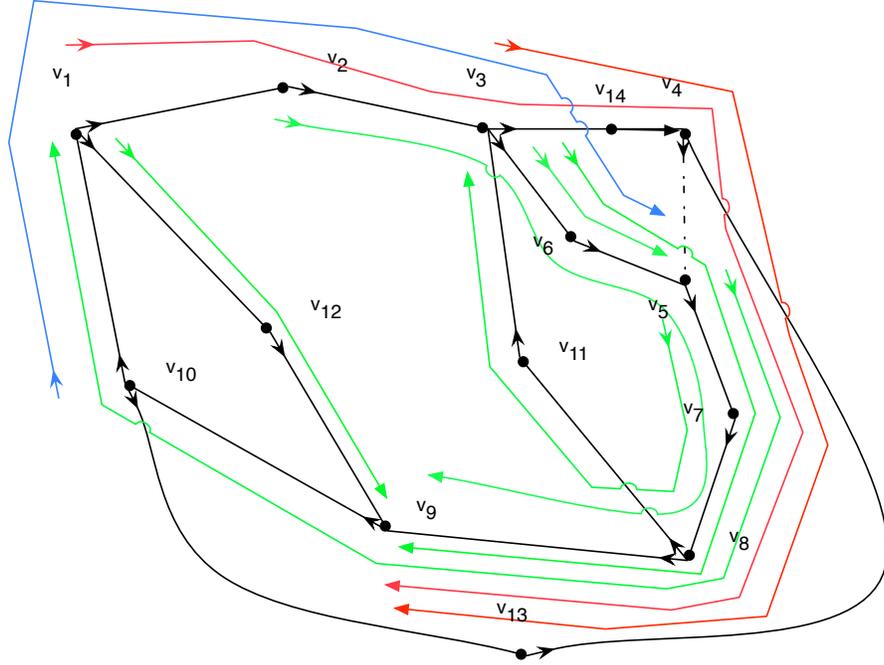
- Node 1:  $s_{\langle 1,12,9 \rangle}^1 = \delta$ .
- Node 10:  $s_{\langle 10,13,4,5 \rangle}^{10} = \delta$ .

The above values together with the *common values* consist the *FALSE value transformation*. In this case, where the variable-subgraph corresponds to a boolean variable with value *FALSE*, the flow-paths have been routed as shown in Figure 8.

### 5.3 Analysis of the reduction

**Lemma 3.** *In a variable-subgraph  $G_A$ , in any connected equilibrium, there is exactly one edge with no successful flow at all. This edge is either  $(v_1, v_2)$  or  $(v_4, v_5)$ .*

*Proof.* Suppose for the sake of contradiction, that there is a connected equilibrium in  $G_A$  and all edges have successful flows. This means that all edges are connected. We first show that in this case the network has no unsuccessful flows at all: if there was a node which was sending an unsuccessful flow then it would be clearly profitable for that node to stop sending this unsuccessful flow since this action would not introduce other changes to the network. Thus all edges carry only successful flows. Consider the edge  $(v_1, v_2)$ . Node  $v_2$  does not receive a flow since there is no demand with sink node  $v_2$ . However there is a successful flow that node  $v_2$  needs to forward. But then condition (3) of Theorem 1 does not hold for the edge  $(v_1, v_2)$  which would mean that the system is not at a Nash equilibrium.



**Fig. 7.** A variable-subgraph which corresponds to a variable with value *TRUE*, with routed flow-paths. Only paths between non-adjacent nodes are shown. There are two unsuccessful flows routed through edge  $(v_4, v_5)$  and disconnect it. These flows are such that if any one of them decreases then edge  $(v_4, v_5)$  gets connected and edge  $(v_5, v_7)$  gets disconnected. Apart from those two, all other flow-paths are successful.

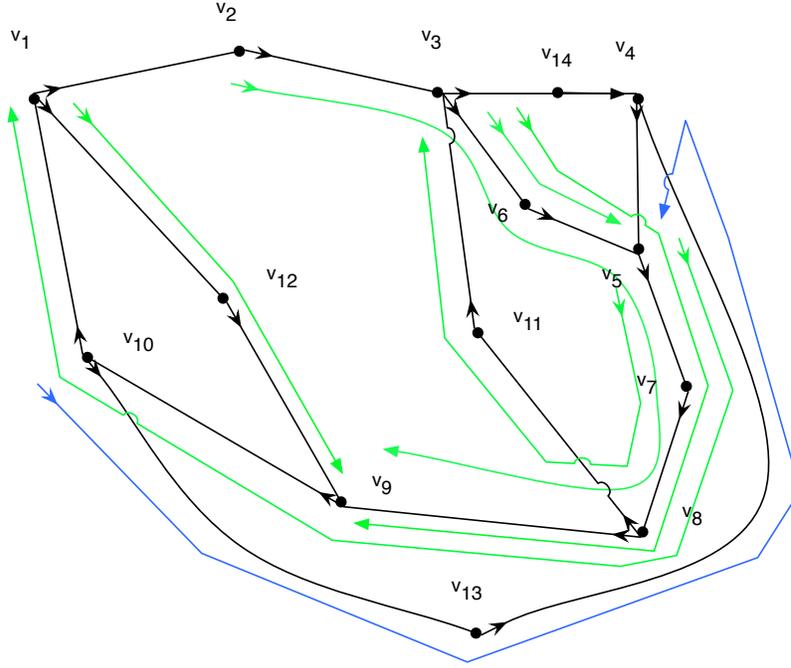
Thus there is at least one edge which does not carry any successful flow. It is easily checkable that any edge apart from  $(v_1, v_2)$  and  $(v_4, v_5)$  should carry successful flow, otherwise there is always a demand for which no flow is being delivered. Now suppose that none of the edges  $(v_1, v_2)$  and  $(v_4, v_5)$  carry successful flow. But then no flow is being delivered for the demand  $(v_{10}, v_5)$ .  $\square$

**Lemma 4.** *A variable-subgraph  $G_A$  where the values to its nodes' decision variables have been assigned according to the *TRUE* value transformation is at a connected Nash equilibrium.*

*Proof.* We prove that if we fix the values of the decision variables as in the *TRUE* value transformation then the  $G_A$  is already at a connected Nash equilibrium. To that end we verify one by one the conditions of Theorem 1.

For the first condition all we have to do is to examine the incoming connected edges of every node:

- $v_1$  forwards flow i) from node  $v_{10}$  to node  $v_5$  through edge  $(v_{10}, v_1)$  and  $s_{(<10,1,2,3,6,5>}^{10} = f_{(10,(10,1),(1,2),5)}^1 = \delta$ ,
- $v_2$  forwards flow i) from node  $v_{10}$  to node  $v_5$  through edge  $(v_1, v_2)$  and  $s_{(<10,1,2,3,6,5>}^{10} = f_{(10,(1,2),(2,3),5)}^2 = \delta$ , ii) from node  $v_1$  to node  $v_9$  through edge  $(v_1, v_2)$  and  $s_{(<1,2,3,14,4,5,7,8,9>}^1 = f_{(1,(1,2),(2,3),9)}^2 = \alpha$ ,



**Fig. 8.** A variable-subgraph which corresponds to a variable with value *FALSE*, with routed flow-paths. Only paths between non-adjacent nodes are shown. All paths carry only successful flows. Although the edge  $(v_1, v_2)$  is not disconnected, there is no flow going through it.

- $v_3$  forwards flow i) from node  $v_{10}$  to node  $v_5$  through edge  $(v_2, v_3)$  and  $s_{\langle 10,1,2,3,6,5 \rangle}^{10} = f_{(10,(2,3),(3,6),5)}^3 = \delta$ , ii) from node  $v_1$  to node  $v_9$  through edge  $(v_2, v_3)$  and  $s_{\langle 1,2,3,14,4,5,7,8,9 \rangle}^1 = f_{(1,(2,3),(3,14),9)}^3 = \alpha$ , iii) from node  $v_2$  to node  $v_9$  through edge  $(v_2, v_3)$  and  $s_{\langle 2,3,6,5,7,8,9 \rangle}^2 = f_{(2,(2,3),(3,6),9)}^3 = 2\delta + \alpha$ ,
- $v_{14}$  forwards flow i) from node  $v_1$  to node  $v_9$  through edge  $(v_3, v_{14})$  and  $s_{\langle 1,2,3,14,4,5,7,8,9 \rangle}^1 = f_{(1,(3,14),(14,4),9)}^{14} = \alpha$ , ii) from node  $v_3$  to node  $v_9$  through edge  $(v_3, v_{14})$  and  $s_{\langle 3,14,4,5,7,8,9 \rangle}^3 = f_{(3,(3,14),(14,4),9)}^{14} = \alpha$ ,
- $v_4$  forwards flow i) from node  $v_1$  to node  $v_9$  through edge  $(v_{14}, v_4)$  and  $s_{\langle 1,2,3,14,4,5,7,8,9 \rangle}^1 = f_{(1,(14,4),(4,5),9)}^4 = \alpha$ , ii) from node  $v_3$  to node  $v_9$  through edge  $(v_{14}, v_4)$  and  $s_{\langle 3,14,4,5,7,8,9 \rangle}^3 = f_{(3,(14,4),(4,5),9)}^4 = \alpha$ ,
- $v_5$  forwards flow i) from node  $v_2$  to node  $v_9$  through edge  $(v_6, v_5)$  and  $s_{\langle 2,3,6,5,7,8,9 \rangle}^2 = f_{(2,(6,5),(5,7),9)}^5 = 2\delta + \alpha$ , ii) from node  $v_3$  to node  $v_9$  through edge  $(v_6, v_5)$  and  $s_{\langle 3,6,5,7,8,9 \rangle}^3 = f_{(3,(6,5),(5,7),9)}^5 = \alpha$ , (notice that in view of Lemma 2 edge  $(v_4, v_5)$  is disconnected, so the first condition of Theorem 1 does not apply for this edge),
- $v_6$  forwards flow i) from node  $v_{10}$  to node  $v_5$  through edge  $(v_3, v_6)$  and  $s_{\langle 10,1,2,3,6,5 \rangle}^{10} = f_{(10,(3,6),(6,5),5)}^6 = \delta$ , ii) from node  $v_2$  to node  $v_9$  through edge  $(v_3, v_6)$  and  $s_{\langle 2,3,6,5,7,8,9 \rangle}^2 = f_{(2,(3,6),(6,5),9)}^6 = 2\delta + \alpha$ , iii) from node  $v_3$  to node  $v_5$  through edge  $(v_3, v_6)$  and  $s_{\langle 3,6,5 \rangle}^3 = f_{(3,(3,6),(6,5),5)}^6 = \delta$ , iv) from node  $v_3$  to node  $v_9$  through edge  $(v_3, v_6)$  and  $s_{\langle 3,6,5,7,8,9 \rangle}^3 = f_{(3,(3,6),(6,5),9)}^6 = \alpha$ ,

- $v_7$  forwards flow i) from node  $v_2$  to node  $v_9$  through edge  $(v_5, v_7)$  and  $s_{\langle 2,3,6,5,7,8,9 \rangle}^2 = f_{(2,(5,7),(7,8),9)}^7 = 2\delta + \alpha$ , ii) from node  $v_3$  to node  $v_9$  through edge  $(v_5, v_7)$  and  $s_{\langle 3,6,5,7,8,9 \rangle}^3 = f_{(3,(5,7),(7,8),9)}^7 = \alpha$  (notice that in view of Lemma 2 edge  $(v_4, v_5)$  is disconnected, and therefore node  $v_7$  does not receive flow from node  $v_3$  through that edge), iii) from node  $v_5$  to node  $v_3$  through edge  $(v_5, v_7)$  and  $s_{\langle 5,7,8,11,3 \rangle}^5 = f_{(5,(5,7),(7,8),3)}^7 = \delta$ , iv) from node  $v_5$  to node  $v_1$  through edge  $(v_5, v_7)$  and  $s_{\langle 5,7,8,9,10,1 \rangle}^5 = f_{(5,(5,7),(7,8),1)}^7 = \delta + 2\alpha$ , (notice that since edge  $(v_4, v_5)$  is disconnected, node  $v_7$  does not receive any flow from node  $v_1$ ),
- $v_8$  forwards flow i) from node  $v_2$  to node  $v_9$  through edge  $(v_7, v_8)$  and  $s_{\langle 2,3,6,5,7,8,9 \rangle}^2 = f_{(2,(7,8),(8,9),9)}^8 = 2\delta + \alpha$ , ii) from node  $v_3$  to node  $v_9$  through edge  $(v_7, v_8)$  and  $s_{\langle 3,6,5,7,8,9 \rangle}^3 = f_{(3,(7,8),(8,9),9)}^8 = \alpha$  (notice that since edge  $(v_4, v_5)$  is disconnected, node  $v_8$  does not receive flow from node  $v_3$  through that edge), iii) from node  $v_5$  to node  $v_3$  through edge  $(v_7, v_8)$  and  $s_{\langle 5,7,8,11,3 \rangle}^5 = f_{(5,(7,8),(8,11),3)}^8 = \delta$ , iv) from node  $v_5$  to node  $v_1$  through edge  $(v_7, v_8)$  and  $s_{\langle 5,7,8,9,10,1 \rangle}^5 = f_{(5,(7,8),(8,9),1)}^8 = \delta + 2\alpha$ , (notice that since edge  $(v_4, v_5)$  is disconnected, node  $v_8$  does not receive flow from node  $v_1$ ),
- $v_9$  forwards flow from node  $v_5$  to node  $v_1$  through edge  $(v_8, v_9)$  and  $s_{\langle 5,7,8,9,10,1 \rangle}^5 = f_{(5,(8,9),(9,10),1)}^9 = \delta + 2\alpha$ ,
- $v_{10}$  forwards flow from node  $v_5$  to node  $v_1$  through edge  $(v_9, v_{10})$  and  $s_{\langle 5,7,8,9,10,1 \rangle}^5 = f_{(5,(9,10),(10,1),1)}^{10} = \delta + 2\alpha$ ,
- $v_{11}$  forwards flow from node  $v_5$  to node  $v_3$  through edge  $(v_8, v_{11})$  and  $s_{\langle 5,7,8,11,3 \rangle}^5 = f_{(5,(8,11),(11,3),3)}^{11} = \delta$ ,
- $v_{12}$  forwards flow from node  $v_1$  to node  $v_9$  through edge  $(v_1, v_{12})$  and  $s_{\langle 1,12,9 \rangle}^1 = \delta - \alpha < f_{(1,(1,12),(12,9),9)}^{12} = \delta$ ,
- $v_{13}$  does not forward any flow since node  $v_{10}$  does not send any flow through edge  $(v_{10}, v_{13})$ .

Therefore for any edge  $(x, y)$  different than  $(v_4, v_5)$  it holds  $\epsilon_x^y = 0$ . Since all thresholds apart from  $THR_4(5)$  are 0 we have that for any edge  $(x, y)$  apart from  $(v_4, v_5)$ ,  $THR_x(y) = \epsilon_x^y = 0$  and  $(x, y)$  is connected. Thus the first two conditions of Theorem 1 hold.

For the third condition we have:

- $v_1$ : receives  $\delta$  flow from node  $v_{10}$  and  $\delta + 2\alpha$  flow from node  $v_5$  through edge  $(v_{10}, v_1)$ ,
- $v_2$ : this condition does not apply for edge  $(v_1, v_2)$  since there is unsuccessful flow in that edge,
- $v_3$ : i) this condition does not apply for edge  $(v_2, v_3)$  since there is unsuccessful flow in that edge, ii) receives  $\delta$  flow from node  $v_{11}$  and  $\delta$  flow from node  $v_5$  through edge  $(v_{11}, v_3)$ ,
- $v_{14}$ : i) this condition does not apply for edge  $(v_3, v_{14})$  since there is unsuccessful flow in that edge,
- $v_4$ : i) this condition does not apply for edge  $(v_{14}, v_4)$  since there is unsuccessful flow in that edge, ii) receives  $\delta$  flow from node  $v_{13}$  through edge  $(v_{13}, v_4)$ ,
- $v_5$  receives i)  $\delta$  flow from node  $v_3$ ,  $2\delta + 3\alpha$  flow from node  $v_6$  and  $\delta$  flow from node  $v_{10}$  through edge  $(v_6, v_5)$ ,
- $v_6$  receives  $4\delta + 2\alpha$  flow from node  $v_3$ ,
- $v_7$  receives  $4\delta + 4\alpha$  flow from node  $v_5$ ,
- $v_8$  receives  $4\delta + 4\alpha$  flow from node  $v_7$ ,
- $v_9$  receives  $2\delta + \alpha$  flow from node  $v_2$ ,  $\alpha$  flow from node  $v_3$ , and  $\delta$  flow from node  $v_8$  through edge  $(v_8, v_9)$ , ii)  $\delta$  flow from node  $v_{12}$  through edge  $(v_{12}, v_9)$ ,

- $v_{10}$  receives  $\delta + 2\alpha$  flow from node  $v_9$  through edge  $(v_9, v_{10})$ ,
- $v_{11}$  receives  $\delta$  flow from node  $v_8$  through edge  $(v_8, v_{11})$ ,
- $v_{12}$  receives  $\delta$  flow from node  $v_1$  through edge  $(v_1, v_{12})$ ,
- $v_{13}$  receives  $\delta$  flow from node  $v_{10}$  through edge  $(v_{10}, v_{13})$ ,

Therefore for every edge  $(x, y)$  carrying only successful flow condition 3 of Theorem 1 holds.

For condition 4 of Theorem 1, the only disconnected edge is  $(v_4, v_5)$ . It holds  $THR_4(5) = 2\lambda > 0$ , node  $v_4$  does not send any flow through this edge and nodes  $v_1, v_3$  send unsuccessful flows through this edge.

For condition 5 of Theorem 1, the only edge for which the condition applies is edge  $(v_{14}, v_4)$ . Indeed, it holds that  $THR_{14}(4) = 0$  and node  $v_4$  receives  $2\alpha$  flow from node  $v_{14}$  which is equal to the (unsuccessful) flow that forwards through edge  $(v_4, v_5)$ .

For condition 6 of Theorem 1, the only demands that are not completely satisfied are between:

- $v_1$  and  $v_9$ , but in order for  $v_1$  to increase what it sends it needs to decrease the unsuccessful flow that sends to  $v_9$ .
- $v_3$  and  $v_9$ , but in order for  $v_3$  to increase what it sends it needs to decrease the unsuccessful flow that sends to  $v_9$ .

Both these cases will be taken care by condition 7d.

For condition 7 of Theorem 1 we have:

- Node  $v_1$ : i) cannot send any unsuccessful flow to cut the flow received from  $v_{10}$  which  $v_1$  needs to forward, ii) cannot stop sending the unsuccessful flow to  $v_9$ , since it will profit  $2\alpha$  more flow in its utility (by rerouting this flow to  $v_9$ ) but lose  $\delta + 2\alpha$  flow that receives from node  $v_5$  since edge  $(v_5, v_7)$  will be disconnected (see Lemma 2),
- Node  $v_2$ : i) cannot send any unsuccessful flow to cut the flow received from  $v_{10}$  which  $v_2$  needs to forward, ii) cannot stop forward the (unsuccessful and successful) flows, since edge  $(v_1, v_2)$  will get disconnected, it will profit  $\delta + \alpha$  flow but lose the  $2\delta + \alpha$  flow that sends to node  $v_9$  since edge  $(v_5, v_7)$  will be disconnected (see Lemma 2),
- Node  $v_3$ : i) cannot send any unsuccessful flow to cut the flows received from  $v_1, v_2, v_{10}$  which  $v_3$  needs to forward, ii) cannot stop forward the (unsuccessful and successful) flows, since edge  $(v_2, v_3)$  will get disconnected, it will profit  $3\delta + 4\alpha - \lambda$  flow (by sending  $\alpha - \lambda$  flow to  $v_9$  through edge  $(v_4, v_5)$ ) but lose  $3\delta + 4\alpha$  flow that receives from node  $v_2$ , iii) cannot stop sending only the unsuccessful flow  $\alpha$  to node  $v_9$  through edge  $(v_4, v_5)$ , since in that case edge  $(v_5, v_7)$  will be disconnected (see Lemma 2) and it will lose  $\alpha$  flow that was sending to node  $v_9$  through edge  $(v_6, v_5)$ ,
- Node  $v_{14}$ : i) cannot send any unsuccessful flow to cut the flows received from  $v_1, v_3$  which  $v_{14}$  needs to forward, ii) cannot stop forward the unsuccessful flows, since edge  $(v_3, v_{14})$  will get disconnected, it will profit  $2\alpha$  flow but lose the same flow that receives from node  $v_3$ , iii) has not a profit to increase its threshold for node  $v_4$ ,
- Node  $v_4$ : i) cannot send any flow, ii) cannot stop forward the unsuccessful flows, since edge  $(v_{14}, v_4)$  will get disconnected, it will profit  $2\alpha$  flow but lose the same flow that receives from node  $v_{14}$ , iii) has not a profit to increase its threshold for node  $v_5$ ,
- Node  $v_5$ : i) cannot send any unsuccessful flow to cut the flows received from  $v_2, v_3$  which needs to forward,

- all other nodes cannot send any unsuccessful flow to cut the flows which need to forward.

Notice also that for all demands of Figure 3 there is a flow delivered. Therefore the equilibrium is connected.  $\square$

**Lemma 5.** *A variable-subgraph  $G_A$  where the values to its nodes' decision variables have been assigned according to the FALSE value transformation is at a connected Nash equilibrium.*

*Proof.* We prove that if we fix the values of the decision variables as in the FALSE value transformation then  $G_A$  is already at a connected Nash equilibrium. To that end we verify one by one the conditions of Theorem 1.

For the first condition all we have to do is to examine the incoming edges of every node:

- $v_1$  does not forward any flow,
- $v_2$  does not forward any flow,
- $v_3$  forwards flow from node  $v_2$  to node  $v_9$  through edge  $(v_2, v_3)$  and  $s_{\langle 2,3,6,5,7,8,9 \rangle}^2 = f_{(2,(2,3),(3,6),9)}^3 = 2\delta + \alpha$ ,
- $v_4$  forwards flow from node  $v_{10}$  to node  $v_5$  through edge  $(v_{13}, v_4)$  and  $s_{\langle 10,13,4,5 \rangle}^{10} = f_{(10,(13,4),(4,5),5)}^4 = \delta$ ,
- $v_5$  forwards flow i) from node  $v_2$  to node  $v_9$  through edge  $(v_6, v_5)$  and  $s_{\langle 2,3,6,5,7,8,9 \rangle}^2 = f_{(2,(6,5),(5,7),9)}^5 = 2\delta + \alpha$ , ii) from node  $v_3$  to node  $v_9$  through edge  $(v_6, v_5)$  and  $s_{\langle 3,6,5,7,8,9 \rangle}^3 = f_{(3,(6,5),(5,7),9)}^5 = \alpha$ ,
- $v_6$  forwards flow i) from node  $v_2$  to node  $v_9$  through edge  $(v_3, v_6)$  and  $s_{\langle 2,3,6,5,7,8,9 \rangle}^2 = f_{(2,(3,6),(6,5),9)}^6 = 2\delta + \alpha$ , ii) from node  $v_3$  to node  $v_5$  through edge  $(v_3, v_6)$  and  $s_{\langle 3,6,5 \rangle}^3 = f_{(3,(3,6),(6,5),5)}^6 = \delta$ , iii) from node  $v_3$  to node  $v_9$  through edge  $(v_3, v_6)$  and  $s_{\langle 3,6,5,7,8,9 \rangle}^3 = f_{(3,(3,6),(6,5),9)}^6 = \alpha$ ,
- $v_7$  forwards flow i) from node  $v_2$  to node  $v_9$  through edge  $(v_5, v_7)$  and  $s_{\langle 2,3,6,5,7,8,9 \rangle}^2 = f_{(2,(5,7),(7,8),9)}^7 = 2\delta + \alpha$ , ii) from node  $v_3$  to node  $v_9$  through edge  $(v_5, v_7)$  and  $s_{\langle 3,6,5,7,8,9 \rangle}^3 = f_{(3,(5,7),(7,8),9)}^7 = \alpha$ , iii) from node  $v_5$  to node  $v_3$  through edge  $(v_5, v_7)$  and  $s_{\langle 5,7,8,11,3 \rangle}^5 = f_{(5,(5,7),(7,8),3)}^7 = \delta$ , iv) from node  $v_5$  to node  $v_1$  through edge  $(v_5, v_7)$  and  $s_{\langle 5,7,8,9,10,1 \rangle}^5 = f_{(5,(5,7),(7,8),1)}^7 = \delta + 2\alpha$ ,
- $v_8$  forwards flow i) from node  $v_2$  to node  $v_9$  through edge  $(v_7, v_8)$  and  $s_{\langle 2,3,6,5,7,8,9 \rangle}^2 = f_{(2,(7,8),(8,9),9)}^8 = 2\delta + \alpha$ , ii) from node  $v_3$  to node  $v_9$  through edge  $(v_7, v_8)$  and  $s_{\langle 3,6,5,7,8,9 \rangle}^3 = f_{(3,(7,8),(8,9),9)}^8 = \alpha$ , iii) from node  $v_5$  to node  $v_3$  through edge  $(v_7, v_8)$  and  $s_{\langle 5,7,8,11,3 \rangle}^5 = f_{(5,(7,8),(8,11),3)}^8 = \delta$ , iv) from node  $v_5$  to node  $v_1$  through edge  $(v_7, v_8)$  and  $s_{\langle 5,7,8,9,10,1 \rangle}^5 = f_{(5,(7,8),(8,9),1)}^8 = \delta + 2\alpha$ ,
- $v_9$  forwards flow from node  $v_5$  to node  $v_1$  through edge  $(v_8, v_9)$  and  $s_{\langle 5,7,8,9,10,1 \rangle}^5 = f_{(5,(8,9),(9,10),1)}^9 = \delta + 2\alpha$ ,
- $v_{10}$  forwards flow from node  $v_5$  to node  $v_1$  through edge  $(v_9, v_{10})$  and  $s_{\langle 5,7,8,9,10,1 \rangle}^5 = f_{(5,(9,10),(10,1),1)}^{10} = \delta + 2\alpha$ ,
- $v_{11}$  forwards flow from node  $v_5$  to node  $v_3$  through edge  $(v_8, v_{11})$  and  $s_{\langle 5,7,8,11,3 \rangle}^5 = f_{(5,(8,11),(11,3),3)}^{11} = \delta$ ,

- $v_{12}$  forwards flow from node  $v_1$  to node  $v_9$  through edge  $(v_1, v_{12})$  and  $s_{(<1,12,9>)}^1 = f_{(1,(1,12),(12,9),9)}^{12} = \delta$ ,
- $v_{13}$  forwards flow from node  $v_{10}$  to node  $v_5$  through edge  $(v_{10}, v_{13})$  and  $s_{(<10,13,4,5>)}^{10} = f_{(10,(10,13),(13,4),5)}^{13} = \delta$ ,

Therefore for any edge  $(x, y)$  it holds  $\epsilon_x^y = 0$ .

For the second condition of the Theorem 1, all thresholds are 0.

For the third condition we have:

- $v_1$ : receives  $\delta$  flow from node  $v_{10}$  and  $\delta + 2\alpha$  flow from node  $v_5$  through edge  $(v_{10}, v_1)$ ,
- $v_2$ : does not receive any flow,
- $v_3$ : receives i)  $3\delta + 3\alpha$  flow from node  $v_2$  through edge  $(v_2, v_3)$ , ii)  $\delta$  flow from node  $v_{11}$  and  $\delta$  flow from node  $v_5$  through edge  $(v_{11}, v_3)$ ,
- $v_{14}$ : receives  $2\alpha$  flow from node  $v_3$  through edge  $(v_3, v_{14})$ ,
- $v_4$ : receives i)  $2\alpha$  flow from node  $v_{14}$  through edge  $(v_{14}, v_4)$ , ii)  $\delta$  flow from node  $v_{13}$  through edge  $(v_{13}, v_4)$ ,
- $v_5$ : receives i)  $\delta$  flow from node  $v_3$ ,  $2\delta + 3\alpha$  flow from node  $v_6$ , through edge  $(v_6, v_5)$ , ii)  $\delta$  flow from node  $v_{10}$  through edge  $(v_4, v_5)$ ,
- $v_6$ : receives  $4\delta + 2\alpha$  flow from node  $v_3$  through edge  $(v_3, v_6)$ ,
- $v_7$ : receives  $4\delta + 4\alpha$  flow from node  $v_5$  through edge  $(v_5, v_7)$ ,
- $v_8$ : receives  $4\delta + 4\alpha$  flow from node  $v_7$  through edge  $(v_7, v_8)$ ,
- $v_9$ : receives i)  $2\delta + \alpha$  flow from node  $v_2$ ,  $\alpha$  flow from node  $v_3$ ,  $\delta$  flow from node  $v_8$  through edge  $(v_8, v_9)$ , ii)  $\delta$  flow from node  $v_1$  through edge  $(v_{12}, v_9)$ ,
- $v_{10}$  receives  $\delta + 2\alpha$  flow from node  $v_9$  through edge  $(v_9, v_{10})$ ,
- $v_{11}$  receives  $\delta$  flow from node  $v_8$  through edge  $(v_8, v_{11})$ ,
- $v_{12}$  receives  $\delta$  flow from node  $v_1$  through edge  $(v_1, v_{12})$ ,
- $v_{13}$  receives  $\delta$  flow from node  $v_{10}$  through edge  $(v_{10}, v_{13})$ ,

Therefore condition 3 of Theorem 1 holds in every edge.

Conditions 4, 5 of Theorem 1 do not apply here since there are no disconnected edges and unsuccessful flows.

For condition 6 of Theorem 1, the only demand which has not been completely satisfied is between nodes  $v_3$  and  $v_9$ . However node  $v_3$  cannot send more flow than  $\alpha$  by any path.

For condition 7 of Theorem 1, since there are no disconnected edges only  $7d$  applies here. We have:

- Node  $v_1$  does not have a profit to send any unsuccessful flow,
- Node  $v_2$  does not have a profit to send any unsuccessful flow,
- Node  $v_3$  cannot send any unsuccessful flow to cut the flow received from  $v_2$  which  $v_3$  needs to forward,
- Node  $v_{14}$  does not have a profit to send any unsuccessful flow,
- Node  $v_4$  cannot send any flow,
- Node  $v_5$  cannot send any unsuccessful flow to cut the flows received from  $v_2, v_3$  which  $v_5$  needs to forward,

- all other nodes cannot send any unsuccessful flow to cut the flows which they need to forward.

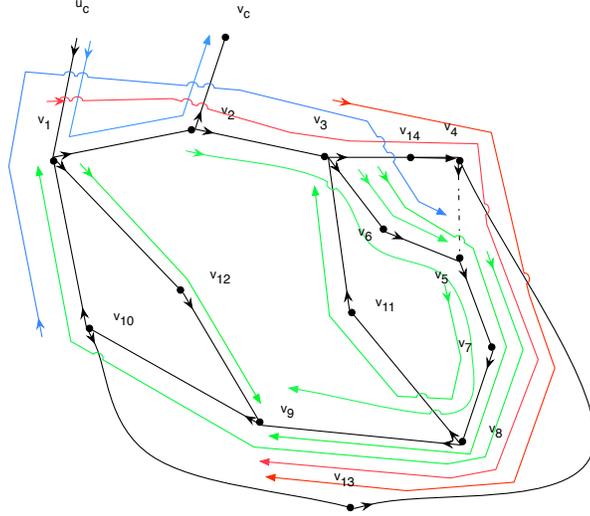
Notice also that for all demands of Figure 3 there is a flow delivered. Therefore the equilibrium is connected.  $\square$

**Lemma 6.** *If the instance  $I$  of the SAT problem is satisfiable then the constructed network has a connected Nash equilibrium.*

*Proof.* Let  $\Phi$  be a truth assignment which satisfies  $I$ . Let  $m$  be the maximum number of clauses satisfied by the same literal in  $\Phi$  and let  $k$  be the maximum number of literals which satisfy the same clause. Consider a variable-subgraph  $G_A$  which corresponds to a boolean variable  $A$  together with the nodes and edges which correspond to the clauses  $B_i$  that have been satisfied by  $A$  in  $\Phi$ .

If  $A$  has value *TRUE* in the satisfying truth assignment  $\Phi$  then it must appear as a positive literal in every  $B_i$  and therefore nodes  $u_{b_i}$  and nodes  $v_{b_i}$  which correspond to those clauses  $B_i$  are connected with  $G_A$  through edges  $(u_{b_i}, v_1)$  and  $(v_2, v_{b_i})$  respectively. The remaining clauses  $C_j$  (if any) in which  $A$  appears as a negative literal correspond to nodes  $u_{c_j}$  and nodes  $v_{c_j}$  that are connected with  $G_A$  through edges  $(u_{c_j}, v_4)$  and  $(v_5, v_{c_j})$  respectively. We show that the enhanced subgraph (i.e., the subgraph consisting of  $G_A$  together with the nodes  $u_{b_i}, v_{b_i}$  and the edges that connect those nodes with  $G_A$ ) has a connected Nash equilibrium. We assign values to the decision variables according to the *TRUE value transformation* and we additionally set  $s_{\langle u_b, 1, 2, v_b \rangle}^{u_b} = f_{(u_b, (u_b, 1), (1, 2), v_b)}^1 = f_{(u_b, (1, 2), (2, v_b), v_b)}^2 = \frac{\delta}{mk}$ ,  $\forall b$  among  $b_i$ . We show that in view of Lemma 4 the system is at a connected Nash equilibrium. We verify that the conditions of Theorem 1 still hold.

- For the first condition of Theorem 1 the differences are:
  - $v_1$  additionally forwards flow from each node  $u_b$  among  $u_{b_i}$  to one node  $v_b$  among  $v_{b_i}$  through edge  $(u_b, v_1)$  and  $s_{\langle u_b, 1, 2, v_b \rangle}^{u_b} = f_{(u_b, (u_b, 1), (1, 2), v_b)}^1 = \frac{\delta}{mk}$ ,
  - $v_2$  additionally forwards flow from each node  $u_b$  among  $u_{b_i}$  to one node  $v_b$  among  $v_{b_i}$  through edge  $(v_1, v_2)$  and  $s_{\langle u_b, 1, 2, v_b \rangle}^{u_b} = f_{(u_b, (1, 2), (2, v_b), v_b)}^2 = \frac{\delta}{mk}$ ,
Therefore the first condition still holds.
- For the second condition there is no difference.
- For the third condition the difference is that  $v_1$  additionally receives  $\frac{\delta}{mk}$  flow from each node  $u_b$  among  $u_{b_i}$  through edge  $(u_b, v_1)$ . Hence the condition still holds.
- For conditions 4, 5 there is no difference.
- For condition 6 of Theorem 1, the demand between each node  $u_b$  among  $u_{b_i}$  and one node  $v_b$  among  $v_{b_i}$ , probably could not be completely satisfied. However node  $u_b$  cannot send more flow by any path. Thus this condition also holds.
- For condition 7 of Theorem 1, i) nodes  $v_1, v_2$  cannot send any unsuccessful flow to cut the new flow from nodes  $u_{b_i}$  which should forward, ii) node  $u_b$  among nodes  $u_{b_i}$  does not have a profit to send any unsuccessful flow and iii) node  $v_2$  still does not have a profit to disconnect edge  $(v_1, v_2)$ , since in that case it could profit at most  $\delta$  flow than before the introduction of nodes  $u_{b_i}$  raising the total to a  $2\delta + \alpha$  flow (see the second bullet in the paragraph for the condition 1 in the proof of Lemma 4) but still losing the same flow that was sending to node  $v_9$  (see the second bullet in the paragraph for the condition 7 in the proof of Lemma 4). Hence this condition also holds.



**Fig. 9.** A positive-literal-subgraph which corresponds to a variable with value *TRUE*, with routed flow-paths.

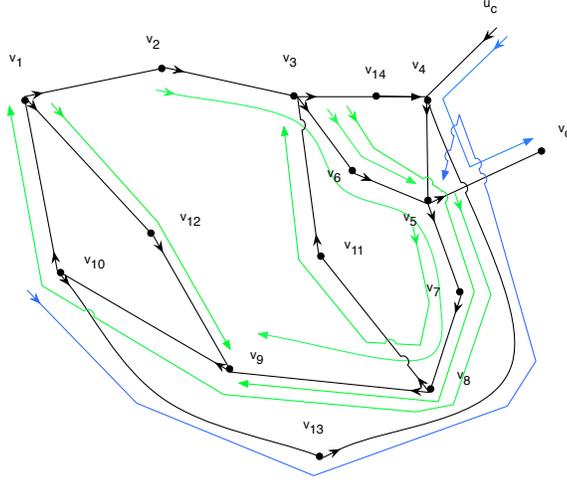
Therefore the system is at a Nash equilibrium. Notice that nodes  $u_{c_j}, v_{c_j}$  cannot distract this equilibrium since  $f_{(u_{c_j},(u_{c_j},4),(4,5),v_c)}^4 = 0, \forall c$  among  $c_j$  and therefore the subgraph including those nodes is still at a Nash equilibrium. Moreover all demands between nodes of  $G_A$  and all demands between a node  $u_b$  among nodes  $u_{b_i}$  and the respected node  $v_b$  among nodes  $v_{b_i}$  are satisfied (i.e., there is a flow delivered). The situation is shown in Figure 9.

If  $A$  has value *FALSE* in the satisfying truth assignment  $\Phi$  then it must appear as a negative literal in every  $B_i$  and therefore nodes  $u_{b_i}$  and nodes  $v_{b_i}$  which correspond to those clauses  $B_i$  are connected with  $G_A$  through edges  $(u_{b_i}, v_4)$  and  $(v_5, v_{b_i})$  respectively. The remaining clauses  $C_j$  (if any) in which  $A$  appears as a positive literal correspond to nodes  $u_{c_j}$  and nodes  $v_{c_j}$  that are connected with  $G_A$  through edges  $(u_{c_j}, v_1)$  and  $(v_2, v_{c_j})$  respectively. We show that the enhanced subgraph (i.e., the subgraph consisting of  $G_A$  together with the nodes  $u_{b_i}, v_{b_i}$  and the edges that connect those nodes with  $G_A$ ) has a connected Nash equilibrium. We assign values to the nodes decision variables according to the *FALSE value transformation* and we additionally set  $s_{\langle u_b,4,5,v_b \rangle}^{u_b} = f_{(u_b,(u_b,4),(4,5),v_b)}^4 = f_{(u_b,(4,5),(5,v_b),v_b)}^5 = \frac{\delta}{mk}, \forall b$  among  $b_i$ . We show that in view of Lemma 5 the system is at a connected Nash equilibrium. We verify that the conditions of Theorem 1 still hold.

- For the first condition of Theorem 1 the differences are:
  - $v_4$  additionally forwards flow from each node  $u_b$  among  $u_{b_i}$  to one node  $v_b$  among  $v_{b_i}$  through edge  $(u_b, v_4)$  and  $s_{\langle u_b,4,5,v_b \rangle}^{u_b} = f_{(u_b,(u_b,4),(4,5),v_b)}^4 = \frac{\delta}{mk}$ ,
  - $v_5$  additionally forwards flow from each node  $u_b$  among  $u_{b_i}$  to one node  $v_b$  among  $v_{b_i}$  through edge  $(v_4, v_5)$  and  $s_{\langle u_b,4,5,v_b \rangle}^{u_b} = f_{(u_b,(4,5),(5,v_b),v_b)}^5 = \frac{\delta}{mk}$ ,

Therefore the first condition still holds.

- For the second condition there is no difference.
- For the third condition the differences are: i) Node  $v_4$  additionally receives  $\frac{\delta}{mk}$  flow from each node  $u_b$  among  $u_{b_i}$  through edge  $(u_b, v_4)$  which is equal to what additionally forwards through that edge. ii) Node  $v_5$  additionally forwards  $\frac{\delta}{mk}$  flow from each node  $u_b$  among  $u_{b_i}$  through edge  $(v_4, v_5)$  which raises the total flow which needs to forward through edge



**Fig. 10.** A negative-literal-subgraph which corresponds to a variable with value *FALSE*, with routed flow-paths.

$(v_4, v_5)$  to at most  $\delta$ . But it also receives a flow  $\delta$  from node  $v_{10}$  through this edge (see the fifth bullet in the paragraph for the condition 3 in the proof of Lemma 5). Hence the condition still holds.

- For conditions 4, 5 there is no difference.
- For condition 6 of Theorem 1, the demand between each node  $u_b$  among  $u_{b_i}$  and one node  $v_b$  among  $v_{b_i}$ , probably could not be completely satisfied. However node  $u_b$  cannot send more flow by any path. Thus this condition also holds.
- For condition 7 of Theorem 1, i) nodes  $v_4, v_5$  cannot send any unsuccessful flow to cut the new flow from nodes  $u_{b_i}$  which should forward, and ii) node  $u_b$  among nodes  $u_{b_i}$  does not have a profit to send any unsuccessful flow. Hence this condition also holds.

Therefore the system is at a Nash equilibrium. Notice that nodes  $u_{c_j}, v_{c_j}$  cannot distract this equilibrium since  $f_{(u_c, (u_c, 1), (1, 2), v_c)}^1 = 0, \forall c$  among  $c_j$  and therefore the subgraph including those nodes is still at a Nash equilibrium. Moreover for every demand between nodes of  $G_A$  and for every demand between a node  $u_b$  among nodes  $u_{b_i}$  and the respected node  $v_b$  among nodes  $v_{b_i}$  there is a flow delivered. The situation is shown in Figure 10.

Notice that there is no demand that could be satisfied by a flow-path routed through edges of two different literal-subgraphs. Hence since each literal-subgraph is at a Nash equilibrium, the whole network is at a Nash equilibrium. Moreover all additional demands between nodes that correspond to clauses are satisfied since each such pair of nodes has to appear as  $u_b, v_b$  connected to some variable-subgraph  $G_A$ , namely this  $G_A$  for which the boolean variable  $A$  satisfies clause  $b$ . Therefore the constructed network is at a connected Nash equilibrium.  $\square$

**Lemma 7.** *Let  $I$  be an instance of the SAT problem. If the constructed  $\pi(I)$  instance of the network game has a connected Nash equilibrium then instance  $I$  of the SAT problem is satisfiable.*

*Proof.* Suppose that the constructed  $\pi(I)$  instance of the network game has a connected equilibrium. This means that for every clause-subgraph represented by a demand between

nodes  $u_c, v_c$  there is at least one literal-subgraph in which there is a flow delivered for this demand (i.e., there is a non-zero successful flow from  $u_c$  to  $v_c$  routed through an edge of that literal-subgraph). If the flow delivered for this demand has been routed through an edge of a positive-literal subgraph then we set the value of the corresponding boolean variable to be *TRUE*. If the flow delivered has been routed through an edge of a negative-literal subgraph then we set the value of the corresponding boolean variable to be *FALSE*. If at the end of this procedure there are still boolean variables with no value assigned then we assign any value to them. We show now that this is a consistent truth assignment. For the sake of contradiction, suppose that during this truth assignment, there is a boolean variable in which both values *TRUE* and *FALSE* have been assigned. This would mean that there is a literal-subgraph in which a demand between nodes  $u_{c_1}, v_{c_1}$  (representing clause  $C_1$ ) has been satisfied (i.e., there is a flow delivered) while the subgraph serves as a positive-literal-subgraph (i.e., the flow from  $u_{c_1}$  to  $v_{c_1}$  has been routed through edge  $(v_1, v_2)$  of the literal-subgraph) and at the same time there is another demand between nodes  $u_{c_2}, v_{c_2}$  (representing clause  $C_2$ ) which has been satisfied while the subgraph serves as a negative-literal-subgraph (i.e., the flow from  $u_{c_2}$  to  $v_{c_2}$  has been routed through edge  $(v_4, v_5)$  of the literal-subgraph). But this would mean that in the corresponding variable-subgraph, both edges  $(v_1, v_2)$  and  $(v_4, v_5)$  have successful flows. But in view of Lemma 3, such a stage in a variable-subgraph cannot be a Nash equilibrium. Hence the truth assignment is consistent and satisfies the boolean formula.  $\square$

From Lemmas 6, 7 we get the following theorem:

**Theorem 5.** *Given a network and a set of demands between nodes, it is NP-hard to decide whether there exist values for the decision variables of the nodes so that the game converges to a connected Nash equilibrium.*

## 6 Conclusion

The question of inducing Nash equilibria with specific attributes is a very general one, and applies to any protocol. In this work we study the property of connectivity, but other natural goals are the maximization of total utility, the maximization of the minimum demand satisfied (similar to concurrent multicommodity flow problems), the maximization of total bandwidth etc. We focused on a basic reputation-based model for ad-hoc networks, but the achievement of most of these goals remains open for this model as well. On the other hand, we were able to characterize the Nash equilibria for it in a way that allowed us to study connectivity properties in a very general setting, i.e., for general topologies and multiple commodities. We would like to combine these properties with additional ones, e.g., maximization of the minimum demand. This would involve network design decisions at the level of setting-up the topology, since there are simple examples with throughput (i.e. the minimum (over all commodities) fraction of satisfied demand) equal to  $\frac{d_{min}}{(k-1)d_{max}}$ , where  $d_{min}, d_{max}$  are the minimum, maximum demands respectively, and  $k$  is the number of commodities. Hence, a natural extension of our results would be to study these extra network design decisions when the installation of every new edge incurs a cost. Another natural extension would be the study of a minimal subset of nodes whose setting of initial values induces an equilibrium with the desired properties. Note that in our results we set the initial values for all nodes, thus inducing an equilibrium ‘in one shot’.

## Acknowledgements

We would like to thank the anonymous reviewer who pointed out that our results hold even when the nodes decide on (and then communicate) their edge flows, instead of deciding their path flows, as described in Section 2.

## References

1. S. Buchegger and J.-Y. Le Boudec. Performance Analysis of the CONFIDANT Protocol: Cooperation Of Nodes Fairness In Dynamic Ad-hoc NeTworks. In *Proceedings of MOBIHOC02*, pp. 226–236, 2002.
2. L. Buttyan and J.-P. Hubaux. Stimulating Cooperation in Self-Organizing Mobile Ad Hoc Networks. In *Proceedings ACM/Kluwer Mobile Networks and Applications*, vol. 8(5), pp. 579–592, 2003.
3. S. Eidenbenz, G. Resta, and P. Santi. COMMIT: a sender-centric truthful and energy-efficient routing protocol for ad hoc networks with selfish nodes. In *Proceedings of 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2005.
4. M. Grötschel, L. Lovász, and A. Schrijver. Geometric Algorithms and Combinatorial Optimization. Springer Verlag, Berlin 1993.
5. G. Hardin. The Tragedy of the Commons. *Science*, Vol. 162, No. 3859, pp. 1243–1248, December 1968.
6. Q. He, D. Wu, and P. Khosla. SORI: A Secure and Objective Reputation-based Incentive Scheme for Ad-hoc Networks. In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC2004)*, pp. 825–830, 2004.
7. R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Sustaining Cooperation in Multihop Wireless Networks. In *Proceedings of Second USENIX Symposium on Networked System Design and Implementation (NSDI’05)*, pp. 231–244, 2005.
8. F. Milan, J. J. Jaramillo, and R. Srikant. Achieving cooperation in multihop wireless networks of selfish nodes. In *Proceedings of the 2006 workshop on Game theory for communications and networks (GameNets)*, 2006.
9. V. Srinivasan, P. Nuggehalli, C.-F. Chiasserini, and R. Rao. Cooperation in wireless ad-hoc networks. In *Proceedings of IEEE INFOCOM 03*, pp. 808–817, 2003.
10. S. Zhong, J. Chen, and Y. R. Yang. Sprite: A simple, Cheat-proof, Credit-based System for Mobile Ad-hoc Networks. In *Proceedings of IEEE INFOCOM03*, pp. 1987–1997, 2003.
11. S. Zhong, L. E. Li, Y. G. Liu, and Y. R. Yang. On designing incentive-compatible routing and forwarding protocols in wireless ad-hoc networks: an integrated approach using game theoretic and cryptographic techniques. In *Wireless Networks*, Vol. 13(6), pp. 799–816, 2007.