Graph Theoretic Detection of Inefficiencies in Network Models

Ivano Salvo, Daniele Gorla, and Pietro Cenciarelli
Sapienza University of Rome, Dpt. of Computer Science

Abstract. We present graph-theoretic characterisations of three notions of inefficiency arising in network models: edge-weakness in flow networks, node-weakness in depletable channels, and vulnerability in traffic networks. Our characterisations lead to three polynomial algorithms that check these forms of inefficiency. Furthermore, checking vulnerability also leads to an advancement on the subgraph homeomorphism problem.

1 Introduction

In this paper, we summarise a bunch of works [4, 3, 6, 5, 9] devoted to graph-theoretically characterise three notions of inefficiency arising in three network models: flow networks, depletable channels, and traffic networks.

In the standard model of flow networks, where flows are constrained by edge capacities, and in depletable channels, where flows are constrained by node charges, inefficiency is related to the existence of non-maximum saturating flows. Thus, a directed graph is said to be *edge-weak* (resp. *node-weak*) if there exists a capacity (resp. charge) assignment to edges (resp. nodes) such that the resulting flow network (resp. depletable channel) admits non-maximum saturating flows.

In the traffic network model for selfish routing, inefficiency is related to the fact that removing edges can lead to networks with a lower latency for agents at the Nash equilibrium (usually referred to as Wardrop equilibrium in this setting). A directed graph is said *vulnerable* if there exists a latency function on edges such that the resulting traffic network suffers from the so-called *Braess-paradox*: the delay of agents in traveling from a given source node to the target node at the Wardrop equilibrium can be reduced by removing edges. Interestingly, also edge-weak and node-weak graphs can be made efficient by removing edges.

Here, we recall our graph theoretic characterisations edge-, node-weakness, and vulnerability. These notions lead to three different classes of graphs in the general case, whereas, in the acyclic case, node-weak graphs are a subset of edge-weak graphs, that in turn coincide with vulnerable graphs. Our characterisations lead to three polynomial time algorithms that check whether a directed graph is edge-weak, node-weak, or vulnerable.

2 Inefficiencies in Network Models

In the following, we will always consider *st*-directed simple graphs, i.e., graphs without self-loops and parallel edges, equipped with a *source* node s (without incoming edges) and a *target* node t (without outgoing edges).

In the standard model of *flow networks* [1], edges are endowed with capacities and admissible flows can not exceed edge capacities. In the graph of Fig. 1(b), two flow units can go from s to t, provided that one of them takes the path $s \to u \to t$ and the other one takes the path $s \to v \to t$. However, a flow of one

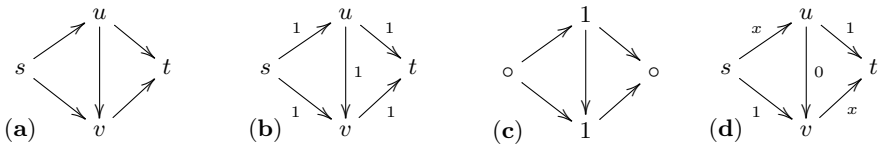

Fig. 1. The Wheatstone graph $\mathcal{W}$ (a) and three models built upon $\mathcal{W}$: a flow network (b), a depletable channel (c), and a traffic network (d).

unit along the path $s \to u \to v \to t$ would saturate the net (i.e., no more flow can be sent from s to t). This is inefficient, since not all possible flow is delivered. We call *edge-weak* those graphs that, as $\mathcal{W}$ in Fig. 1(a), admit non-maximum saturating flows for some capacity-to-edge assignments, as in Fig. 1(b).

In *depletable channels* [3], a model for energy consumption in wireless networks, each node u is equipped with an integer $\eta(u) \geq 0$ representing its depletable charge. Node charges constrain admissible flows: in the channel of Fig. 1(c) (built upon $\mathcal{W}$), nodes are labelled with their charge ('$\circ$' stands for 'unlimited'). As before, we can transmit two information units, or saturate the net with one unit only. We call *node-weak* those graphs (as $\mathcal{W}$) that admit non-maximum saturating flows for some charge-to-node assignment (Fig. 1(c)).

In *traffic networks* [2], edges are labelled with latency functions, that models agent delay along an edge in terms of its congestion, that is the flow on that edge. Autonomous selfish users choose the faster path and the traffic stabilises to an equilibrium of a noncooperative game, (*Wardrop equilibrium*). The traffic network in Fig. 1(d) is built on top of $\mathcal{W}$: two edges ($u \to t$ and $s \to v$) cause a delay of 1 regardless the flow on them; the edge ($u \to v$) causes no delay; two edges ($v \to t$ and $s \to u$) cause a delay linear in the flow on them. Selfish users control a negligible part of traffic ε and choose the quickest path $p : s \to u \to v \to t$, because there they experience a delay of $2\,\varepsilon$ instead of $1 + \varepsilon$ experienced in other paths However, this leads to the congestion of p: in a flow of value 1, at the Wardrop equilibrium, all agents choose the path p and their delay is 2. This is an equilibrium, since agents have no interest in deviating from their choice. Paradoxically, if we remove the 'ideal' edge $u \to v$, half of users will choose the path $s \to u \to t$ and half the path $s \to v \to t$ resulting in a delay of 3/2. This phenomenon has been known for a long time as *Braess's paradox* [2], that occurs when the equilibrium cost may be reduced by removing edges.

The property of a graph (as $\mathcal{W}$ in Fig. 1(a)) to lead, under some latency function (as the one in Fig. 1(d)), to the possibility of experiencing the Braess's paradox has been called *vulnerability* in [11].

3 A Graph Theoretic Perspective

We have characterised node-weakness [4] and edge-weakness [5] in terms of minimal separators, and vulnerability [6] in terms of subgraph embedding.

A *minimal vertex separator* (*mvs* for short) [10] of an st–graph is a minimal set of nodes whose removal disconnects s from t.

Theorem 1 ([4]). *A directed st-graph G is node-weak if and only if there exists a path from s to t touching some mvs of G at least twice.*

As discussed in Sect. 2, the graph $\mathcal{W}$ in Fig. 1(a) is node-weak: indeed, the mvs $\{u, v\}$ is touched twice by the path $s \to u \to v \to t$. By contrast, the graph C in Fig. 2 is not node-weak, since its saturating flows have always value $\min\{\eta(v_1) + \eta(v_2), \eta(v_3) + \eta(v_4)\}$: no st-path touches any of its mvs's twice.

Similarly, a *minimal edge separator* (*mes* for short) of an st–graph is a minimal set of edges whose removal disconnects s from t.

Theorem 2 ([5]). *A directed st-graph G is edge-weak if and only if there exists a path from s to t touching some mes of G at least twice.*

For example, the graph C in Fig. 2 is edge-weak, because if we assign capacity 0 to one of its two diagonal edges ($v_1 \to v_4$ or $v_2 \to v_3$) and 1 to all other edges, we essentially obtain a flow network with the same flows as that of Fig. 1(b). Indeed, the path $s \to v_1 \to v_4 \to t$ touches twice the mes $\{s \to v_1, v_2 \to v_3, v_4 \to t\}$.

We have characterised vulnerable graphs as those graphs in which the pattern graph $\mathcal{W}$ can be embedded. An *st-embedding* of $\mathcal{W}$ into an st-graph G is a injective map ψ from nodes of $\mathcal{W}$ to nodes of G such that: (1) for all edges $u \to v$ of $\mathcal{W}$, there exists a path from $\psi(u)$ to $\psi(v)$ in G, (2) $\psi(u) \rightsquigarrow \psi(v)$ and $\psi(u') \rightsquigarrow \psi(v')$ are node disjoint paths (apart for their extremes) whenever $u \neq u'$ or $v \neq v'$, and (3) there exist two (possibly empty) node-disjoint paths in G from its source to $\phi(s)$ and from $\phi(t)$ to its target.

Theorem 3 ([6]). *A directed st-graph G is vulnerable iff there is an st-embedding of $\mathcal{W}$ into G. If G is vulnerable, then G is edge-weak.*

The embedding ψ of $\mathcal{W}$ into the graph C in Fig. 2 that maps source into source, target into target, u into v_1, and v into v_4 induces the following mapping of edges of $\mathcal{W}$ to disjoint paths of C (we denote $s_{\mathcal{W}}$ and $t_{\mathcal{W}}$ the source and target of $\mathcal{W}$, and the same for s_C and t_C): $\{(s_{\mathcal{W}} \to u, s_c \to v_1), (s_{\mathcal{W}} \to v, s_c \to v_2 \to v_4), (u \to v, v_1 \to v_4), (u \to t_{\mathcal{W}}, v_1 \to v_3 \to t_C), (v \to t_{\mathcal{W}}, v_4 \to t_c)\}$. Here, the paths from s_C to $\psi(s_{\mathcal{W}})$ and from $\psi(t_{\mathcal{W}})$ to t_C are both empty.

As shown in Sect. 2, the graph $\mathcal{W}$ is edge-weak, node-weak, and vulnerable. Let us now consider graphs in Fig. 2. A is node-weak, because its mvs $\{u\}$ is touched twice by the path $s \to u \to v \to u \to t$. By contrast, A is not edge-weak (and so neither vulnerable): its only mes's are $\{s \to u\}$ and $\{u \to t\}$ and no path passes twice through them. The graph B is node-weak and edge-weak because of the path $s \to u \to v \to u \to v \to t$ that touches twice both the mvs $\{u\}$ and the mes $\{u \to v\}$; however, it is not vulnerable, since there is no st-embedding of $\mathcal{W}$ into it. As said above, the graph C is vulnerable and thus edge-weak, but not node-weak. Finally, the graph D is not node-weak (its mvs's are $\{s\}$, $\{u\}$, $\{v\}$, and $\{t\}$ and no one of them is touched twice by a path) nor vulnerable (there is no st-embedding of $\mathcal{W}$ into it), but it is edge-weak, because the path $s \to u \to x \to y \to x \to y \to v \to t$ passes twice through the mes $\{u \to v, x \to y\}$.

In the general case, the inclusion diagram is depicted top-left in Fig. 2; if we restrict our attention to directed acyclic graphs (DAG), the inclusion diagram is depicted top-right in Fig. 2, as a consequence of the following result.

Theorem 4 ([5]). *If an st-graph G is a DAG, then the following statements are equivalent: (1) G is vulnerable; (2) G is edge-weak; (3) G is not two–terminal series–parallel; (4) G contains a node-weak subgraph.*

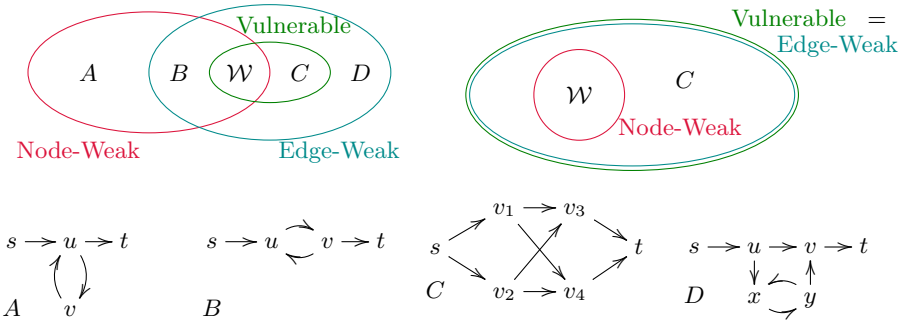

Fig. 2. The inclusion diagram for general directed graphs (top-left) and directed acyclic graphs (top-right). W is the Wheatstone graph from Fig. 1(a).

4 Polynomial Detection of Net Inefficiency

Detecting Braess-paradox in a traffic network is an *NP*-complete problem [11]. We proved [4, 3] that computing the minimum saturating flow in depletable channels is *NP*-hard (and the associated decisional problem is *NP*-complete). This result can be easily adapted to flow networks. Stemming from our results of Sect. 3, we succeeded in finding polynomial algorithms to check edge-weakness, node-weakness, and vulnerability.

A trivial algorithm to check if a graph G is node-weak is to generate all its mvs's [10] and, for each mvs S check if there exists a path between two nodes of S. However, the number of mvs's can be exponential in $|V|$. In [4], we present a polynomial algorithm that checks weakness by checking at worst $\mathcal{O}(|V|^2)$ mvs.

Theorem 5 ([4]). *Checking whether an st-graph $G = (V, E)$ is node-weak can be solved in time $\mathcal{O}(|V|^2 \cdot |E|)$.*

The same algorithm to detect node-weakness can be adapted to edge-weakness, by letting mes play the role of mvs, thus proving the following.

Theorem 6 ([9]). *Checking whether a directed st-graph $G = (V, E)$ is edge-weak can be solved in time $\mathcal{O}(|E|^3)$.*

An *st*-graph is *redundant* if it contains redundant edges, i.e., edges that do not belong to any simple path from s to t. For example, the graph B in Fig. 2 is redundant because of the edge $u \to v$. Redundant edges play no role in Wardrop equilibria: removing a redundant edge from a graph G leads to a graph G' that is vulnerable if and only if G is itself vulnerable.

In [7], it was proved that vulnerable and not redundant graphs coincide with two-terminal series-parallel graphs, and this implies the existence of efficient algorithms to check if a non-redundant *st*-graph is vulnerable [12]. However, we proved [6] that checking if a graph is redundant is itself *NP*-complete and thus, checking preconditions of the algorithm proposed by [7] is computationally hard.

In [6], we presented an algorithm that works on *any* directed *st*-graph G. At each main step, we analyse a cycle in G and either: (1) we remove a redundant edge, thus breaking a cycle; or (2) we find an *st*-embedding of $\mathcal{W}$. This process always ends finding an *st*-embedding of $\mathcal{W}$ or in a DAG G' that is vulnerable iff G is vulnerable; and the vulnerability of G' can be checked in linear time.

Theorem 7 ([6]). *Checking whether a directed st-graph $G = (V, E)$ is vulnerable can be solved in time $\mathcal{O}(|V| \cdot |E|^2)$.*

The algorithm for checking vulnerability can be adapted to the homeomorphic subgraph problem without node mapping. In general, this problem is *NP*-complete [8], except when, as $\mathcal{W}$, the pattern graph has all nodes with degree at most 3. In this case, some pattern graphs are known for which this problem is polynomial. We have added the pattern graph $\mathcal{W}$ to this family.

Theorem 8 ([6]). *Checking whether a directed graph $G = (V, E)$ contains a subgraph homeomorphic to $\mathcal{W}$ can be solved in time $\mathcal{O}(|V|^3 \cdot |E|^2)$.*

5 Open Problems

We conclude by singling out some interesting open problems. First of all, it would be interesting to investigate if our graph theoretic characterisations and the corresponding polynomial algorithms can be extended to the multi-commodity model, where several source and target nodes are present.

Another issue is efficiency of our algorithms. Are they optimal? Which are lower-bounds of checking node-, edge-weakness, and vulnerability?

Finally, it would be interesting investigating a general algorithm for checking the subgraph homeomorphism problem for all pattern graphs with at most 3 as node degree, at least for patterns that are, as $\mathcal{W}$, *st*-graphs.

References

1. R. Ahuja, T. Magnanti, and J. Orlin. *Network Flows, theory, algorithms, and applications*. Prentice-Hall, New Jersey, 1993.
2. M. Beckmann, C. B. McGuire, and C. B. Winsten. *Studies in the Economics of Transportation*. Yale University Press, 1956.
3. P. Cenciarelli, D. Gorla, and I. Salvo. Depletable channels: Dynamics and behaviour. In *Proc. of FCT09*, volume 5690 of *LNCS*, pages 50–61. Springer, 2009.
4. P. Cenciarelli, D. Gorla, and I. Salvo. Depletable channels: Dynamics, behaviour, and efficiency in network design. *Acta Informatica*, 2018. Accepted.
5. P. Cenciarelli, D. Gorla, and I. Salvo. Inefficiencies in network models: A graph theoretic perspective. *Information Processing Letters*, 131:44–50, 2018.
6. P. Cenciarelli, D. Gorla, and I. Salvo. A polynomial-time algorithm for checking the possibility of braess paradox in directed graphs. *Algorithmica*, 2018. In press.
7. X. Chen, Z. Diao, and X. Hu. Network characterizations for excluding braess's paradox. *Theory of Computing Systems*, pages 1–34, 2016.
8. S. Fortune, J. Hopcroft, and J. Wyllie. The directed subgraph homeomorphism problem. *Theoretical Computer Science*, 10(2):111 – 121, 1980.
9. F. Granese. Polynomially recognising graphs where saturating flows are always maximum. BSc Thesis, Sapienza University of Rome., October 2017.
10. T. Kloks and D. Kratsch. Listing all minimal separators of a graph. *SIAM J. Comput.*, 27(3):605–613, 1998.
11. T. Roughgarden. On the severity of Braess's Paradox: Designing networks for selfish users is hard. *J. Comput. Syst. Sci.*, 72(5):922–953, 2006.
12. J. Valdes, R. Tarjan, and E. Lawler. The recognition of series-parallel digraphs. *SIAM Journal of Computing*, 11:298–313, 1982.