Resolving Braess's Paradox in Random Networks*

Dimitris Fotakis[1], Alexis C. Kaporis[2], Thanasis Lianeas[1], and Paul G. Spirakis[3,4]

[1] Electrical and Computer Engineering, National Technical University of Athens, Greece
[2] Information and Communication Systems Dept., University of the Aegean, Samos, Greece
[3] Department of Computer Science , University of Liverpool, UK
[4] Computer Technology Institute and Press – Diophantus, Patras, Greece
fotakis@cs.ntua.gr, kaporisa@gmail.com, tlianeas@mail.ntua.gr,
P.Spirakis@liverpool.ac.uk, spirakis@cti.gr

Abstract. Braess's paradox states that removing a part of a network may improve the players' latency at equilibrium. In this work, we study the approximability of the best subnetwork problem for the class of random $\mathcal{G}_{n,p}$ instances proven prone to Braess's paradox by (Roughgarden and Valiant, RSA 2010) and (Chung and Young, WINE 2010). Our main contribution is a polynomial-time approximation-preserving reduction of the best subnetwork problem for such instances to the corresponding problem in a simplified network where all neighbors of s and t are directly connected by 0 latency edges. Building on this, we obtain an approximation scheme that for any constant $\varepsilon > 0$ and with high probability, computes a subnetwork and an ε-Nash flow with maximum latency at most $(1+\varepsilon)L^* + \varepsilon$, where L^* is the equilibrium latency of the best subnetwork. Our approximation scheme runs in polynomial time if the random network has average degree $O(\text{poly}(\ln n))$ and the traffic rate is $O(\text{poly}(\ln \ln n))$, and in quasipolynomial time for average degrees up to $o(n)$ and traffic rates of $O(\text{poly}(\ln n))$.

1 Introduction

An instance of a (non-atomic) *selfish routing* game consists of a network with a source s and a sink t, and a traffic rate r divided among an infinite number of players. Every edge has a non-decreasing function that determines the edge's latency caused by its traffic. Each player routes a negligible amount of traffic through an $s-t$ path. Observing the traffic caused by others, every player selects an $s-t$ path that minimizes the sum of edge latencies. Thus, the players reach a *Nash equilibrium* (a.k.a., a *Wardrop equilibrium*), where all players use paths of equal minimum latency. Under some general assumptions on the latency functions, a Nash equilibrium flow (or simply a *Nash flow*) exists and the common players' latency in a Nash flow is essentially unique (see e.g., [14]).

Previous Work. It is well known that a Nash flow may not optimize the network performance, usually measured by the *total latency* incurred by all players. Thus, in the

* This research was supported by the project Algorithmic Game Theory, co-financed by the European Union (European Social Fund - ESF) and Greek national funds, through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: THALES, investing in knowledge society through the European Social Fund, by the ERC project RIMACO, and by the EU FP7/2007-13 (DG INFSO G4-ICT for Transport) under Grant Agreement no. 288094 (Project eCompass).

Y. Chen and N. Immorlica (Eds.): WINE 2013, LNCS 8289, pp. 188–201, 2013.

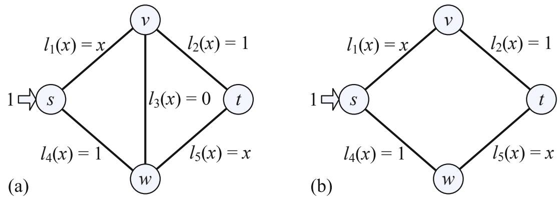

Fig. 1. (a) The optimal total latency is $3/2$, achieved by routing half of the flow on each of the paths (s, v, t) and (s, w, t). In the (unique) Nash flow, all traffic goes through the path (s, v, w, t) and has a latency of 2. (b) If we remove the edge (v, w), the Nash flow coincides with the optimal flow. Hence the network (b) is the *best subnetwork* of network (a).

last decade, there has been a significant interest in quantifying and understanding the performance degradation due to the players' selfish behavior, and in mitigating (or even eliminating) it using several approaches, such as introducing economic disincentives (tolls) for the use of congested edges, or exploiting the presence of centrally coordinated players (Stackelberg routing), see e.g., [14] and the references therein.

A simple way to improve the network performance at equilibrium is to exploit Braess's paradox [3], namely the fact that removing some edges may improve the latency of the Nash flow[1] (see e.g., Fig. 1 for an example). Thus, given an instance of selfish routing, one naturally seeks for the *best subnetwork*, i.e. the subnetwork minimizing the common players' latency at equilibrium. Compared against Stackelberg routing and tolls, edge removal is simpler and more appealing to both the network administrator and the players (see e.g., [6] for a discussion).

Unfortunately, Roughgarden [15] proved that it is NP-hard not only to find the best subnetwork, but also to compute any meaningful approximation to its equilibrium latency. Specifically, he proved that even for linear latencies, it is NP-hard to approximate the equilibrium latency of the best subnetwork within a factor of $4/3 - \varepsilon$, for any $\varepsilon > 0$, i.e., within any factor less than the worst-case Price of Anarchy for linear latencies. On the positive side, applying Althöfer's Sparsification Lemma [1], Fotakis, Kaporis, and Spirakis [6] presented an algorithm that approximates the equilibrium latency of the best subnetwork within an additive term of ε, for any constant $\varepsilon > 0$, in time that is subexponential if the total number of $s - t$ paths is polynomial, all paths are of polylogarithmic length, and the traffic rate is constant.

Interestingly, Braess's paradox can be dramatically more severe in networks with multiple sources and sinks. More specifically, Lin et al. [8] proved that for networks with a single source-sink pair and general latency functions, the removal of at most k edges cannot improve the equilibrium latency by a factor greater than $k + 1$. On the other hand, Lin et al. [8] presented a network with two source-sink pairs where the removal of a single edge improves the equilibrium latency by a factor of $2^{\Omega(n)}$. As for

[1] Due to space constraints, we have restricted the discussion of related work to the most relevant results on the existence and the elimination of Braess's paradox. There has been a large body of work on quantifying and mitigating the consequences of Braess's paradox on selfish traffic, especially in the areas of Transportation Science and Computer Networks. The interested reader may see e.g., [15,12] for more references.

the impact of the network topology, Milchtaich [11] proved that Braess's paradox does not occur in series-parallel networks, which is precisely the class of networks that do not contain the network in Fig. 1.a as a topological minor.

Recent work actually indicates that the appearance of Braess's paradox is not an artifact of optimization theory, and that edge removal can offer a tangible improvement on the performance of real-world networks (see e.g., [7,13,14,16]). In this direction, Valiant and Roughgarden [17] initiated the study of Braess's paradox in natural classes of random networks, and proved that the paradox occurs with high probability in dense random $\mathcal{G}_{n,p}$ networks, with $p = \omega(n^{-1/2})$, if each edge e has a linear latency $\ell_e(x) = a_e x + b_e$, with a_e, b_e drawn independently from some reasonable distribution. The subsequent work of Chung and Young [4] extended the result of [17] to sparse random networks, where $p = \Omega(\ln n/n)$, i.e., just greater than the connectivity threshold of $\mathcal{G}_{n,p}$, assuming that the network has a large number of edges e with small additive latency terms b_e. In fact, Chung and Young demonstrated that the crucial property for Braess's paradox to emerge is that the subnetwork consisting of the edges with small additive terms is a good expander (see also [5]). Nevertheless, the proof of [4,17] is merely existential; it provides no clue on how one can actually find (or even approximate) the best subnetwork and its equilibrium latency.

Motivation and Contribution. The motivating question for this work is whether in some interesting settings, where the paradox occurs, we can efficiently compute a set of edges whose removal significantly improves the equilibrium latency. From a more technical viewpoint, our work is motivated by the results of [4,17] about the prevalence of the paradox in random networks, and by the knowledge that in random instances some hard (in general) problems can actually be tractable.

Departing from [4,17], we adopt a purely algorithmic approach. We focus on the class of so-called *good* selfish routing instances, namely instances with the properties used by [4,17] to demonstrate the occurrence of Braess's paradox in random networks with high probability. In fact, one can easily verify that the random instances of [4,17] are good with high probability. Rather surprisingly, we prove that, in many interesting cases, we can efficiently approximate the best subnetwork and its equilibrium latency. What may be even more surprising is that our approximation algorithm is based on the expansion property of good instances, namely the very same property used by [4,17] to establish the prevalence of the paradox in good instances! To the best of our knowledge, our results are the first of theoretical nature which indicate that Braess's paradox can be efficiently eliminated in a large class of interesting instances.

Technically, we present essentially an approximation scheme. Given a good instance and any constant $\varepsilon > 0$, we compute a flow g that is an ε-Nash flow for the subnetwork consisting of the edges used by it, and has a latency of $L(g) \leq (1 + \varepsilon)L^* + \varepsilon$, where L^* is the equilibrium latency of the best subnetwork (Theorem 1). In fact, g has these properties with high probability. Our approximation scheme runs in polynomial time for the most interesting case that the network is relatively sparse and the traffic rate r is $O(\mathrm{poly}(\ln \ln n))$, where n is the number of vertices. Specifically, the running time is polynomial if the good network has average degree $O(\mathrm{poly}(\ln n))$, i.e., if $pn = O(\mathrm{poly}(\ln n))$, for random $\mathcal{G}_{n,p}$ networks, and quasipolynomial for average degrees up to $o(n)$. As for the traffic rate, we emphasize that most work on selfish routing and selfish network design problems assumes that $r = 1$, or at least that r does not increase with the network's size (see e.g., [14] and the references therein). So, we can approximate, in polynomial-time, the best subnetwork for a large class of instances that, with high probability, include exponentially many $s - t$ paths and $s - t$ paths of length $\Theta(n)$. For such instances, a direct application of [6, Theorem 3] gives an exponential-time algorithm.

The main idea behind our approximation scheme, and our main technical contribution, is a polynomial-time approximation-preserving reduction of the best subnetwork problem for a good network G to a corresponding best subnetwork problem for a 0-*latency simplified network* G_0, which is a layered network obtained from G if we keep only s, t and their immediate neighbors, and connect all neighbors of s and t by direct edges of 0 latency. We first show that the equilibrium latency of the best subnetwork does not increase when we consider the 0-latency simplified network G_0 (Lemma 1). Although this may sound reasonable, we highlight that decreasing edge latencies to 0 may trigger Braess's paradox (e.g., starting from the network in Fig. 1.a with $l'_3(x) = 1$, and decreasing it to $l_3(x) = 0$ is just another way of triggering the paradox). Next, we employ Althöfer's Sparsification Lemma [1] (see also [9,10] and [6, Theorem 3]) and approximate the best subnetwork problem for the 0-latency simplified network.

The final (and crucial) step of our approximation preserving reduction is to start with the flow-solution to the best subnetwork problem for the 0-latency simplified network, and extend it to a flow-solution to the best subnetwork problem for the original (good) instance. To this end, we show how to "simulate" 0-latency edges by low latency paths in the original good network. Intuitively, this works because due to the expansion properties and the random latencies of the good network G, the intermediate subnetwork of G, connecting the neighbors of s to the neighbors of t, essentially behaves as a complete bipartite network with 0-latency edges. This is also the key step in the approach of [4,17], showing that Braess's paradox occurs in good networks with high probability (see [4, Section 2] for a detailed discussion). Hence, one could say that to some extent, the reason that Braess's paradox exists in good networks is the very same reason that the paradox can be efficiently resolved. Though conceptually simple, the full construction is technically involved and requires dealing with the amount of flow through the edges incident to s and t and their latencies. Our construction employs a careful grouping-and-matching argument, which works for good networks with high probability, see Lemmas 4 and 5.

We highlight that the reduction itself runs in polynomial time. The time consuming step is the application of [6, Theorem 3] to the 0-latency simplified network. Since such networks have only polynomially many (and very short) $s - t$ paths, they escape the hardness result of [15]. The approximability of the best subnetwork for 0-latency simplified networks is an intriguing open problem arising from our work.

Our result shows that a problem, that is NP-hard to approximate, can be very closely approximated in random (and random-like) networks. This resembles e.g., the problem of finding a Hamiltonian path in Erdös-Rényi graphs, where again, existence and construction both work just above the connectivity threshold, see e.g., [2]. However, not all hard problems are easy when one assumes random inputs (e.g., consider factoring or the hidden clique problem, for both of which no such results are known in full depth).

2 Model and Preliminaries

Notation. For an event E in a sample space, $\mathbb{P}[E]$ denotes the probability of E happening. We say that an event E occurs *with high probability*, if $\mathbb{P}[E] \geq 1 - n^{-\alpha}$, for some constant $\alpha \geq 1$, where n usually denotes the number of vertices of the network G to which E refers. We implicitly use the union bound to account for the occurrence of more than one low probability events.

Instances. A *selfish routing instance* is a tuple $\mathcal{G} = (G(V,E), (\ell_e)_{e\in E}, r)$, where $G(V,E)$ is an undirected network with a source s and a sink t, $\ell_e : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is a non-decreasing latency function associated with each edge e, and $r > 0$ is the traffic rate. We let $\mathcal{P}$ (or $\mathcal{P}_G$, whenever the network G is not clear from the context) denote the (non-empty) set of simple $s-t$ paths in G. For brevity, we usually omit the latency functions, and refer to a selfish routing instance as (G, r).

We only consider linear latencies $\ell_e(x) = a_e x + b_e$, with $a_e, b_e \geq 0$. We restrict our attention to instances where the coefficients a_e and b_e are randomly selected from a pair of distributions $\mathcal{A}$ and $\mathcal{B}$. Following [4,17], we say that $\mathcal{A}$ and $\mathcal{B}$ are *reasonable* if:

- $\mathcal{A}$ has bounded range $[A_{\min}, A_{\max}]$ and $\mathcal{B}$ has bounded range $[0, B_{\max}]$, where $A_{\min} > 0$ and $A_{\max}$, $B_{\max}$ are constants, i.e., they do not depend on r and $|V|$.
- There is a closed interval $I_{\mathcal{A}}$ of positive length, such that for every non-trivial subinterval $I' \subseteq I_{\mathcal{A}}$, $\mathbb{P}_{a\sim\mathcal{A}}[a \in I'] > 0$.
- There is a closed interval $I_{\mathcal{B}}$, $0 \in I_{\mathcal{B}}$, of positive length, such that for every non-trivial subinterval $I' \subseteq I_{\mathcal{B}}$, $\mathbb{P}_{b\sim\mathcal{B}}[b \in I'] > 0$. Moreover, for any constant $\eta > 0$, there exists a constant $\delta_\eta > 0$, such that $\mathbb{P}_{b\sim\mathcal{B}}[b \leq \eta] \geq \delta_\eta$.

Subnetworks. Given a selfish routing instance $(G(V,E), r)$, any subgraph $H(V', E')$, $V' \subseteq V$, $E' \subseteq E$, $s, t \in V'$, obtained from G by edge and vertex removal, is a *subnetwork* of G. H has the same source s and sink t as G, and the edges of H have the same latencies as in G. Every instance $(H(V', E'), r)$, where $H(V', E')$ is a subnetwork of $G(V,E)$, is a *subinstance* of $(G(V,E), r)$.

Flows. Given an instance (G, r), a (feasible) *flow* f is a non-negative vector indexed by $\mathcal{P}$ such that $\sum_{q\in\mathcal{P}} f_q = r$. For a flow f, let $f_e = \sum_{q:e\in q} f_q$ be the amount of flow that f routes on edge e. Two flows f and g are *different* if there is an edge e with $f_e \neq g_e$. An edge e is used by flow f if $f_e > 0$, and a path q is used by f if $\min_{e\in q}\{f_e\} > 0$. We often write $f_q > 0$ to denote that a path q is used by f. Given a flow f, the latency of each edge e is $\ell_e(f_e)$, the latency of each path q is $\ell_q(f) = \sum_{e\in q} \ell_e(f_e)$, and the latency of f is $L(f) = \max_{q:f_q>0} \ell_q(f)$. We sometimes write $L_G(f)$ when the network G is not clear from the context. For an instance $(G(V,E), r)$ and a flow f, we let $E_f = \{e \in E : f_e > 0\}$ be the set of edges used by f, and $G_f(V, E_f)$ be the corresponding subnetwork of G.

Nash Flow. A flow f is a *Nash (equilibrium) flow*, if it routes all traffic on minimum latency paths. Formally, f is a Nash flow if for every path q with $f_q > 0$, and every path q', $\ell_q(f) \leq \ell_{q'}(f)$. Therefore, in a Nash flow f, all players incur a common latency $L(f) = \min_q \ell_q(f) = \max_{q:f_q>0} \ell_q(f)$ on their paths. A Nash flow f on a network $G(V,E)$ is a Nash flow on any subnetwork $G'(V', E')$ of G with $E_f \subseteq E'$.

Every instance (G, r) admits at least one Nash flow, and the players' latency is the same for all Nash flows (see e.g., [14]). Hence, we let $L(G, r)$ be the players' latency in some Nash flow of (G, r), and refer to it as the equilibrium latency of (G, r). For linear latency functions, a Nash flow can be computed efficiently, in strongly polynomial time, while for strictly increasing latencies, the Nash flow is essentially unique (see e.g., [14]).

ε-Nash flow. The definition of a Nash flow can be naturally generalized to that of an "almost Nash" flow. Formally, for some $\varepsilon > 0$, a flow f is an ε-Nash flow if for every path q with $f_q > 0$, and every path q', $\ell_q(f) \leq \ell_{q'}(f) + \varepsilon$.

Best Subnetwork. Braess's paradox shows that there may be a subinstance (H, r) of an instance (G, r) with $L(H, r) < L(G, r)$ (see e.g., Fig. 1). The *best subnetwork* H^* of (G, r) is a subnetwork of G with the minimum equilibrium latency, i.e., H^* has $L(H^*, r) \leq L(H, r)$ for any subnetwork H of G. In this work, we study the approximability of the *Best Subnetwork Equilibrium Latency* problem, or $\mathrm{BestSubEL}$ in short. In $\mathrm{BestSubEL}$, we are given an instance (G, r), and seek for the best subnetwork H^* of (G, r) and its equilibrium latency $L(H^*, r)$.

Good Networks. We restrict our attention to undirected $s - t$ networks $G(V, E)$. We let $n \equiv |V|$ and $m \equiv |E|$. For any vertex v, we let $\Gamma(v) = \{u \in V : \{u, v\} \in E\}$ denote the set of v's neighbors in G. Similarly, for any non-empty $S \subseteq V$, we let $\Gamma(S) = \bigcup_{v \in S} \Gamma(v)$ denote the set of neighbors of the vertices in S, and let $G[S]$ denote the subnetwork of G induced by S. For convenience, we let $V_s \equiv \Gamma(s)$, $E_s \equiv \{\{s, u\} : u \in V_s\}$, $V_t \equiv \Gamma(t)$, $E_t \equiv \{\{v, t\} : v \in V_t\}$, and $V_m \equiv V \setminus (\{s, t\} \cup V_s \cup V_t)$. We also let $n_s = |V_s|$, $n_t = |V_t|$, $n_+ = \max\{n_s, n_t\}$, $n_- = \min\{n_s, n_t\}$, and $n_m = |V_m|$. We sometimes write $V(G)$, $n(G)$, $V_s(G)$, $n_s(G)$, ..., if G is not clear from the context.

It is convenient to think that the network G has a layered structure consisting of s, the set of s's neighbors V_s, an "intermediate" subnetwork connecting the neighbors of s to the neighbors of t, the set of t's neighbors V_t, and t. Then, any $s - t$ path starts at s, visits some $u \in V_s$, proceeds either directly or through some vertices of V_m to some $v \in V_t$, and finally reaches t. Thus, we refer to $G_m \equiv G[V_s \cup V_m \cup V_t]$ as the *intermediate subnetwork* of G. Depending on the structure of G_m, we say that:

- G is a *random* $\mathcal{G}_{n,p}$ network if (i) n_s and n_t follow the binomial distribution with parameters n and p, and (ii) if any edge $\{u, v\}$, with $u \in V_m \cup V_s$ and $v \in V_m \cup V_t$, exists independently with probability p. Namely, the intermediate network G_m is an Erdös-Rényi random graph with $n - 2$ vertices and edge probability p, except for the fact that there are no edges in $G[V_s]$ and in $G[V_t]$.
- G is *internally bipartite* if the intermediate network G_m is a bipartite graph with independent sets V_s and V_t. G is *internally complete bipartite* if every neighbor of s is directly connected by an edge to every neighbor of t.
- G is 0-*latency simplified* if it is internally complete bipartite and every edge e connecting a neighbor of s to a neighbor of t has latency function $\ell_e(x) = 0$.

The 0-*latency simplification* G_0 of a given network G is a 0-latency simplified network obtained from G by replacing $G[V_m]$ with a set of 0-latency edges directly connecting every neighbor of s to every neighbor of t. Moreover, we say that a 0-latency simplified network G is *balanced*, if $|n_s - n_t| \leq 2n_-$.

Algorithm 1. Approximation Scheme for $\mathrm{BestSubEL}$ in Good Networks

Input: Good network $G(V, E)$, rate $r > 0$, approximation guarantee $\varepsilon > 0$
Output: Subnetwork H of G and ε-Nash flow g in H with $L(g) \leq (1+\varepsilon)L(H^*, r) + \varepsilon$
1 if $L(G, r) < \varepsilon$, return G and a Nash flow of (G, r) ;
2 create the 0-latency simplification G_0 of G ;
3 if $r \geq (B_{\max} n_+)/(\varepsilon A_{\min})$, then let $H_0 = G_0$ and let f be a Nash flow of (G_0, r) ;
4 else, let H_0 be the subnetwork and f the $\varepsilon/6$-Nash flow of Thm. 2 applied with error $\varepsilon/6$;
5 let H be the subnetwork and let g be the ε-Nash flow of Lemma 5 starting from H_0 and f ;
6 return the subnetwork H and the ε-Nash flow g ;

We say that a network $G(V, E)$ is (n, p, k)-*good*, for some integer $n \leq |V|$, some probability $p \in (0, 1)$, with $pn = o(n)$, and some constant $k \geq 1$, if G satisfies that:

1. The maximum degree of G is at most $3np/2$, i.e., for any $v \in V$, $|\Gamma(v)| \leq 3np/2$.
2. G is an *expander graph*, namely, for any set $S \subseteq V$, $|\Gamma(S)| \geq \min\{np|S|, n\}/2$.
3. The edges of G have random reasonable latency functions distributed according to $\mathcal{A} \times \mathcal{B}$, and for any constant $\eta > 0$, $\mathbb{P}_{b \sim \mathcal{B}}[b \leq \eta / \ln n] = \omega(1/np)$.
4. If $k > 1$, we can compute in polynomial time a partitioning of V_m into k sets $V_m^1, \ldots, V_m^k$, each of cardinality $|V_m|/k$, such that all the induced subnetworks $G[\{s, t\} \cup V_s \cup V_m^i \cup V_t]$ are $(n/k, p, 1)$-good, with a possible violation of the maximum degree bound by s and t.

If G is a *random* $\mathcal{G}_{n,p}$ network, with n sufficiently large and $p \geq ck \ln n/n$, for some large enough constant $c > 1$, then G is an (n, p, k)-good network with high probability (see e.g., [2]), provided that the latency functions satisfy condition (3) above. As for condition (4), a random partitioning of V_m into k sets of cardinality $|V_m|/k$ satisfies (4) with high probability. Similarly, the random instances considered in [4] are good with high probability. Also note that the 0-latency simplification of a good network is balanced, due to (1) and (2).

3 The Approximation Scheme and Outline of the Analysis

In this section, we describe the main steps of the approximation scheme (see also Algorithm 1), and give an outline of its analysis. We let $\varepsilon > 0$ be the approximation guarantee, and assume that $L(G, r) \geq \varepsilon$. Otherwise, any Nash flow of (G, r) suffices.

Algorithm 1 is based on an approximation-preserving reduction of $\mathrm{BestSubEL}$ for a good network G to $\mathrm{BestSubEL}$ for the 0-latency simplification G_0 of G. The first step of our approximation-preserving reduction is to show that the equilibrium latency of the best subnetwork does not increase when we consider the 0-latency simplification G_0 of a network G instead of G itself. Since decreasing the edge latencies (e.g., decreasing $l'_3(x) = 1$ to $l_3(x) = 0$ in Fig. 1.a) may trigger Braess's paradox, we need Lemma 1, in Section 4, and its careful proof to make sure that zeroing out the latency of the intermediate subnetwork does not cause an abrupt increase in the equilibrium latency.

Next, we focus on the 0-latency simplification G_0 of G (step 2 in Alg. 1). We show that if the traffic rate is large enough, i.e., if $r = \Omega(n_+/\varepsilon)$, the paradox has a marginal influence on the equilibrium latency. Thus, any Nash flow of (G_0, r) is a $(1 + \varepsilon)$-approximation of $\mathrm{BestSubEL}$ (see Lemma 2 and step 4). If $r = O(n_+/\varepsilon)$, we use [6, Theorem 3] and obtain an $\varepsilon/6$-approximation of $\mathrm{BestSubEL}$ for (G_0, r) (see Theorem 2 and step 4).

We now have a subnetwork H_0 and an $\varepsilon/6$-Nash flow f that comprise a good approximate solution to $\mathrm{BestSubEL}$ for the simplified instance (G_0, r). The next step of our approximation-preserving reduction is to extend f to an approximate solution to $\mathrm{BestSubEL}$ for the original instance (G, r). The intuition is that due to the expansion and the reasonable latencies of G, any collection of 0-latency edges of H_0 used by f to route flow from V_s to V_t can be "simulated" by an appropriate collection of low-latency paths of the intermediate subnetwork G_m of G. In fact, this observation was the key step in the approach of [4,17] showing that Braess's paradox occurs in good networks with high probability. We first prove this claim for a small part of H_0 consisting only of neighbors of s and neighbors of t with approximately the same latency under f (see Lemma 4, the proof draws on ideas from [4, Lemma 5]). Then, using a careful latency-based grouping of the neighbors of s and of the neighbors of t in H_0, we extend this claim to the entire H_0 (see Lemma 5). Thus, we obtain a subnetwork H of G and an ε-Nash flow g in H such that $L(g) \leq (1+\varepsilon)L(H^*, r) + \varepsilon$ (step 5).

We summarize our main result. The proof follows by combining Lemma 1, Theorem 2, and Lemma 5 in the way indicated by Algorithm 1 and the discussion above.

Theorem 1. *Let $G(V, E)$ be an (n, p, k)-good network, where $k \geq 1$ is a large enough constant, let $r > 0$ be any traffic rate, and let H^* be the best subnetwork of (G, r). Then, for any $\varepsilon > 0$, Algorithm 1 computes in time $n_+^{O(r^2 A_{\max}^2 \ln(n_+)/\varepsilon^2)} \mathrm{poly}(|V|)$, a flow g and a subnetwork H of G such that with high probability, wrt. the random choice of the latency functions, g is an ε-Nash flow of (H, r) and has $L(g) \leq (1+\varepsilon)L(H^*)+\varepsilon$.*

By the definition of reasonable latencies, $A_{\max}$ is a constant. Also, by Lemma 2, r affects the running time only if $r = O(n_+/\varepsilon)$. In fact, previous work on selfish network design assumes that $r = O(1)$, see e.g., [14]. Thus, if $r = O(1)$ (or more generally, if $r = O(\mathrm{poly}(\ln \ln n)))$ and $pn = O(\mathrm{poly}(\ln n))$, in which case $n_+ = O(\mathrm{poly}(\ln n))$, Theorem 1 gives a randomized polynomial-time approximation scheme for $\mathrm{BestSubEL}$ in good networks. Moreover, the running time is quasipolynomial for traffic rates up to $O(\mathrm{poly}(\ln n))$ and average degrees up to $o(n)$, i.e., for the entire range of p in [4,17]. The next sections are devoted to the proofs of Lemmas 1 and 5, and of Theorem 2.

4 Network Simplification

We first show that the equilibrium latency of the best subnetwork does not increase when we consider the 0-latency simplification G_0 of a network G instead of G itself. We highlight that the following lemma holds not only for good networks, but also for any network with linear latencies and with the layered structure described in Section 2.

Lemma 1. *Let G be any network, let $r > 0$ be any traffic rate, and let H be the best subnetwork of (G, r). Then, there is a subnetwork H' of the 0-latency simplification of H (and thus, a subnetwork of G_0) with $L(H', r) \leq L(H, r)$.*

Proof sketch. We assume that all the edges of H are used by the equilibrium flow f of (H, r) (otherwise, we can remove all unused edges from H). The proof is constructive, and at the conceptual level, proceeds in two steps. For the first step, given the equilibrium flow f of the best subnetwork H of G, we construct a simplification H_1 of H that is internally bipartite and has constant latency edges connecting $\Gamma(s)$ to $\Gamma(t)$. H_1 also admits f as an equilibrium flow, and thus $L(H_1, r) = L(H, r)$. We can also show how to further simplify H_1 so that its intermediate bipartite subnetwork becomes acyclic.

The second part of the proof is to show that we can either remove some of the intermediate edges of H_1 or zero their latencies, and obtain a subnetwork H' of the 0-latency simplification of H with $L(H', r) \leq L(H, r)$. To this end, we describe a procedure where in each step, we either remove some intermediate edge of H_1 or zero its latency, without increasing the latency of the equilibrium flow.

Let us focus on an edge $e_{kl} = \{u_k, v_l\}$ connecting a neighbor u_k of s to a neighbor v_l of t. By the first part of the proof, the latency function of e_{kl} is a constant $b_{kl} > 0$. Next, we attempt to set the latency of e_{kl} to $b'_{kl} = 0$. We have also to change the equilibrium flow f to a new flow f' that is an equilibrium flow of latency at most L in the modified network with $b'_{kl} = 0$. We should be careful when changing f to f', since increasing the flow through $\{s, u_k\}$ and $\{v_l, t\}$ affects the latency of all $s - t$ paths going through u_k and v_l and may destroy the equilibrium property (or even increase the equilibrium latency). In what follows, we let r_q be the amount of flow moving from an $s - t$ path $q = (s, u_i, v_j, t)$ to the path $q_{kl} = (s, u_k, v_l, t)$ when we change f to f'. We note that r_q may be negative, in which case, $|r_q|$ units of flow actually move from q_{kl} to q. Thus, r_q's define a rerouting of f to a new flow f', with $f'_q = f_q - r_q$, for any $s - t$ path q other than q_{kl}, and $f'_{kl} = f_{kl} + \sum_q r_q$.

We next show how to compute r_q's so that f' is an equilibrium flow of cost at most L in the modified network (where we attempt to set $b'_{kl} = 0$). We let $\mathcal{P} = \mathcal{P}_{H_1} \setminus \{q_{kl}\}$ denote the set of all $s - t$ paths in H_1 other than q_{kl}. We let $\boldsymbol{F}$ be the $|\mathcal{P}| \times |\mathcal{P}|$ matrix, indexed by the paths $q \in \mathcal{P}$, where $\boldsymbol{F}[q_1, q_2] = \sum_{e \in q_1 \cap q_2} a_e - \sum_{e \in q_1 \cap q_{kl}} a_e$, and let $\boldsymbol{r}$ be the vector of r_q's. Then, the q-th component of $\boldsymbol{F}\boldsymbol{r}$ is equal to $\ell_q(f) - \ell_q(f')$. In the following, we consider two cases depending on whether $\boldsymbol{F}$ is singular or not.

If matrix $\boldsymbol{F}$ is non-singular, the linear system $\boldsymbol{F}\boldsymbol{r} = \varepsilon\,\mathbf{1}$ has a unique solution $\boldsymbol{r}_\varepsilon$, for any $\varepsilon > 0$. Moreover, due to linearity, for any $\alpha \geq 0$, the unique solution of the system $\boldsymbol{F}\boldsymbol{r} = \alpha\,\varepsilon\,\mathbf{1}$ is $\alpha\,\boldsymbol{r}_\varepsilon$. Therefore, for an appropriately small $\varepsilon > 0$, the linear system $Q_\varepsilon = \{\boldsymbol{F}\boldsymbol{r} = \varepsilon\,\mathbf{1}, f_q - r_q \geq 0\ \forall q \in \mathcal{P}, f_{kl} + \sum_q r_q \geq 0, \ell_{q_{kl}}(f') \leq L + b_{kl} - \varepsilon\}$ admits a unique solution $\boldsymbol{r}$. We keep increasing ε until one of the inequalities of Q_ε becomes tight. If it first becomes $r_q = f_q$ for some path $q = (s, u_i, v_j, t) \in \mathcal{P}$, we remove the edge $\{u_i, v_j\}$ from H_1 and adjust the constant latency of e_{kl} so that $\ell_{q_{kl}}(f') = L - \varepsilon$. Then, the flow f' is an equilibrium flow of cost $L - \varepsilon$ for the resulting network, which has one edge less than the original network H_1. If $\sum_q r_q < 0$ and it first becomes $\sum_q r_q = -f_{kl}$, we remove the edge e_{kl} from H_1. Then, f' is an equilibrium flow of cost $L - \varepsilon$ for the resulting network, which again has one edge less than H_1. If $\sum_q r_q > 0$ and it first becomes $\ell_{q_{kl}}(f') = L + b_{kl} - \varepsilon$, we set the constant latency of the edge e_{kl} to $b'_{kl} = 0$. In this case, f' is an equilibrium flow of cost $L - \varepsilon$ for the resulting network that has one edge of 0 latency more than the initial network H_1. Moreover, we can show that if q_{kl} is disjoint from the paths $q \in \mathcal{P}$, the fact that the intermediate network H_1 is acyclic implies that the matrix $\boldsymbol{F}$ is positive definite, and thus non-singular. Therefore, if q_{kl} is disjoint from the paths in $\mathcal{P}$,the procedure above leads to a decrease in the equilibrium latency, and eventually to setting $b'_{kl} = 0$.

If $\boldsymbol{F}$ is singular, we can compute r_q's so that f' is an equilibrium flow of cost L in a modified network that includes one edge less than the original network H_1. If $\boldsymbol{F}$ is singular, the homogeneous linear system $\boldsymbol{F}\boldsymbol{r} = \boldsymbol{0}$ admits a nontrivial solution $\boldsymbol{r} \neq \boldsymbol{0}$. Moreover, due to linearity, for any $\alpha \in \mathbb{R}$, $\alpha\,\boldsymbol{r}$ is also a solution to $\boldsymbol{F}\boldsymbol{r} = \boldsymbol{0}$. Therefore, the linear system $Q_0 = \{\boldsymbol{F}\boldsymbol{r} = \boldsymbol{0}, f_q - r_q \geq 0 \ \forall q \in \mathcal{P}, f_{kl} + \sum_q r_q \geq 0\}$ admits a solution $\boldsymbol{r} \neq \boldsymbol{0}$ that makes at least one of the inequalities tight. We recall that the q-th component of $\boldsymbol{F}\boldsymbol{r}$ is equal to $\ell_q(f) - \ell_q(f')$. Therefore, for the flow f' obtained from the particular solution $\boldsymbol{r}$ of Q_0, the latency of any path $q \in \mathcal{P}$ is equal to L. If $\boldsymbol{r}$ is such that $r_q = f_q$ for some path $q = (s, u_i, v_j, t) \in \mathcal{P}$, we remove the edge $\{u_i, v_j\}$ from H_1 and adjust the constant latency of e_{kl} so that $\ell_{q_{kl}}(f') = L$. Then, the flow f' is an equilibrium flow of cost L for the resulting network, which has one edge less than the original network H_1. If $\boldsymbol{r}$ is such that $\sum_q r_q = -f_{kl}$, we remove the edge e_{kl} from H_1. Then, f' is an equilibrium flow of cost L for the resulting network, which again has one edge less than H_1.

Each time we apply the procedure above either we decrease the number of edges of the intermediate network by one or we increase the number of 0-latency edges of the intermediate network by one, without increasing the latency of the equilibrium flow. So, by repeatedly applying these steps, we end up with a subnetwork H' of the 0-latency simplification of H with $L(H', r) \leq L(H, r)$. □

5 Approximating the Best Subnetwork of Simplified Networks

We proceed to show how to approximate the BestSubEL problem in a balanced 0-latency simplified network G_0 with reasonable latencies. We may always regard G_0 as the 0-latency simplification of a good network G. We first state two useful lemmas about the maximum traffic rate r up to which BestSubEL remains interesting, and about the maximum amount of flow routed on any edge / path in the best subnetwork.

Lemma 2. *Let G_0 be any 0-latency simplified network, let $r > 0$, and let H_0^* be the best subnetwork of (G_0, r). For any $\varepsilon > 0$, if $r > \frac{B_{\max} n_+}{A_{\min} \varepsilon}$, then $L(G_0, r) \leq (1+\varepsilon) L(H_0^*, r)$.*

Proof. We assume that $r > \frac{B_{\max} n_+}{A_{\min} \varepsilon}$, let f be a Nash flow of (G_0, r), and consider how f allocates r units of flow to the edges of $E_s \equiv E_s(G_0)$ and to the edges $E_t \equiv E_t(G_0)$. For simplicity, we let $L \equiv L(G_0, r)$ denote the equilibrium latency of G_0, and let $A_s = \sum_{e \in E_s} 1/a_e$ and $A_t = \sum_{e \in E_t} 1/a_e$.

Since G_0 is a 0-latency simplified network and f is a Nash flow of (G_0, r), there are $L_1, L_2 > 0$, with $L_1 + L_2 = L$, such that all used edges incident to s (resp. to t) have latency L_1 (resp. L_2) in the Nash flow f. Since $r > \frac{B_{\max} n_+}{A_{\min}}$, $L_1, L_2 > B_{\max}$ and all edges in $E_s \cup E_t$ are used by f. Moreover, by an averaging argument, we have that there is an edge $e \in E_s$ with $a_e f_e \leq r/A_s$, and that there is an edge $e \in E_t$ with $a_e f_e \leq r/A_t$. Therefore, $L_1 \leq (r/A_s) + B_{\max}$ and $L_2 \leq (r/A_t) + B_{\max}$, and thus, $L \leq \frac{r}{A_s} + \frac{r}{A_t} + 2B_{\max}$.

On the other hand, if we ignore the additive terms b_e of the latency functions, the optimal average latency of the players is $r/A_s + r/A_t$, which implies that $L(H_0^*, r) \geq r/A_s + r/A_t$. Therefore, $L \leq L(H_0^*, r) + 2B_{\max}$. Moreover, since $r > \frac{B_{\max} n_+}{A_{\min}\varepsilon}$, $A_s \leq n_s/A_{\min}$, and $A_t \leq n_t/A_{\min}$, we have that:

$$L(H_0^*, r) \geq \frac{r}{A_s} + \frac{r}{A_t} \geq \frac{B_{\max} n_s}{A_{\min}\varepsilon} \frac{A_{\min}}{n_s} + \frac{B_{\max} n_t}{A_{\min}\varepsilon} \frac{A_{\min}}{n_t} \geq 2B_{\max}/\varepsilon$$

Therefore, $2B_{\max} \leq \varepsilon L(H_0^*, r)$, and $L \leq (1+\varepsilon) L(H_0^*, r)$. □

Lemma 3. *Let G_0 be a balanced 0-latency simplified network with reasonable latencies, let $r > 0$, and let f be a Nash flow of the best subnetwork of (G_0, r). For any $\varepsilon > 0$, if $\mathbb{P}_{b \sim \mathcal{B}}[b \leq \varepsilon/4] \geq \delta$, for some constant $\delta > 0$, there exists a constant $\rho = \frac{24 A_{\max} B_{\max}}{\delta \varepsilon A_{\min}^2}$ such that with probability at least $1 - \mathrm{e}^{-\delta n_-/8}$, $f_e \leq \rho$, for all edges e.*

Approximating the Best Subnetwork of Simplified Networks. We proceed to derive an approximation scheme for the best subnetwork of any simplified instance (G_0, r).

Theorem 2. *Let G_0 be a balanced 0-latency simplified network with reasonable latencies, let $r > 0$, and let H_0^* be the best subnetwork of (G_0, r). Then, for any $\varepsilon > 0$, we can compute, in time $n_+^{O(A_{\max}^2 r^2 \ln(n_+)/\varepsilon^2)}$, a flow f and a subnetwork H_0 consisting of the edges used by f, such that (i) f is an ε-Nash flow of (H_0, r), (ii) $L(f) \leq L(H_0^*, r) + \varepsilon/2$, and (iii) there exists a constant $\rho > 0$, such that $f_e \leq \rho + \varepsilon$, for all e.*

Theorem 2 is a corollary of [6, Theorem 3], since in our case the number of different $s-t$ paths is at most n_+^2 and each path consists of 3 edges. So, in [6, Theorem 3], we have $d_1 = 2$, $d_2 = 0$, $\alpha = A_{\max}$, and the error is ε/r. Moreover, we know that any Nash flow g of (H_0^*, r) routes $g_e \leq \rho$ units of flow on any edge e, and that in the exhaustive search step, in the proof of [6, Theorem 3], one of the acceptable flows f has $|g_e - f_e| \leq \varepsilon$, for all edges e (see also [6, Lemma 3]). Thus, there is an acceptable flow f with $f_e \leq \rho + \varepsilon$, for all edges e. In fact, if among all acceptable flows enumerated in the proof of [6, Theorem 3], we keep the acceptable flow f that minimizes the maximum amount flow routed on any edge, we have that $f_e \leq \rho + \varepsilon$, for all edges e.

6 Extending the Solution to the Good Network

Given a good instance (G, r), we create the 0-latency simplification G_0 of G, and using Theorem 2, we compute a subnetwork H_0 and an $\varepsilon/6$-Nash flow f that comprise an approximate solution to BestSubEL for (G_0, r). Next, we show how to extend f to an approximate solution to BestSubEL for the original instance (G, r). The intuition is that the 0-latency edges of H_0 used by f to route flow from V_s to V_t can be "simulated" by low-latency paths of G_m. We first formalize this intuition for the subnetwork of G induced by the neighbors of s with (almost) the same latency B_s and the neighbors of t with (almost) the same latency B_t, for some B_s, B_t with $B_s + B_t \approx L(f)$. We may think of the networks G and H_0 in the lemma below as some small parts of the original network G and of the actual subnetwork H_0 of G_0. Thus, we obtain the following lemma, which serves as a building block in the proof of Lemma 5.

Lemma 4. *We assume that $G(V, E)$ is an $(n, p, 1)$-good network, with a possible violation of the maximum degree bound by s and t, but with $|V_s|, |V_t| \leq 3knp/2$, for some constant $k > 0$. Also the latencies of the edges in $E_s \cup E_t$ are not random, but there exist constants $B_s, B_t \geq 0$, such that for all $e \in E_s$, $\ell_e(x) = B_s$, and for all $e \in E_t$, $\ell_e(x) = B_t$. We let $r > 0$ be any traffic rate, let H_0 be any subnetwork of the 0-latency simplification G_0 of G, and let f be any flow of (H_0, r). We assume that there exists a constant $\rho' > 0$, such that for all $e \in E(H_0)$, $0 < f_e \leq \rho'$. Then, for any $\epsilon_1 > 0$, with high probability, wrt. the random choice of the latency functions of G, we can compute in $\text{poly}(|V|)$ time a subnetwork G' of G, with $E_s(G') = E_s(H_0)$ and $E_t(G') = E_t(H_0)$, and a flow g of (G', r) such that (i) $g_e = f_e$ for all $e \in E_s(G') \cup E_t(G')$, (ii) g is a $7\epsilon_1$-Nash flow in G', and (iii) $L_{G'}(g) \leq B_s + B_t + 7\epsilon_1$.*

Proof sketch. For convenience and wlog., we assume that $E_s(G) = E_s(H_0)$ and that $E_t(G) = E_t(H_0)$, so that we simply write V_s, V_t, E_s, and E_t from now on. For each $e \in E_s \cup E_t$, we let $g_e = f_e$. So, the flow g satisfies (i), by construction.

We compute the extension of g through G_m as an "almost" Nash flow in a modified version of G, where each edge $e \in E_s \cup E_t$ has a capacity $g_e = f_e$ and a constant latency $\ell_e(x) = B_s$, if $e \in E_s$, and $\ell_e(x) = B_t$, if $e \in E_t$. All other edges e of G have an infinite capacity and a (randomly chosen) reasonable latency function $\ell_e(x)$.

We let g be the flow of rate r that respects the capacities of the edges in $E_s \cup E_t$, and minimizes $\text{Pot}(g) = \sum_{e \in E} \int_0^{g_e} \ell_e(x)dx$. Such a flow g can be computed in strongly polynomial time (see e.g., [18]). The subnetwork G' of G is simply G_g, namely, the subnetwork that includes only the edges used by g. It could have been that g is not a Nash flow of (G, r), due to the capacity constraints on the edges of $E_s \cup E_t$. However, since g is a minimizer of $\text{Pot}(g)$, for any $u \in V_s$ and $v \in V_t$, and any pair of $s-t$ paths q, q' going through u and v, if $g_q > 0$, then $\ell_q(g) \leq \ell_{q'}(g)$. Thus, g can be regarded as a Nash flow for any pair $u \in V_s$ and $v \in V_t$ connected by g-used paths.

To conclude the proof, we adjust the proof of [4, Lemma 5], and show that for any $s-t$ path q used by g, $\ell_q(g) \leq B_s + B_t + 7\epsilon_1$. To prove this, we let $q = (s, u, \ldots, v, t)$ be the $s - t$ path used by g that maximizes $\ell_q(g)$. We show the existence of a path $q' = (s, u, \ldots, v, t)$ in G of latency $\ell_{q'}(g) \leq B_s + B_t + 7\epsilon_1$. Therefore, since g is a minimizer of $\text{Pot}(g)$, the latency of the maximum latency g-used path q, and thus the latency of any other g-used $s - t$ path, is at most $B_s + B_t + 7\epsilon_1$, i.e., g satisfies (iii). Moreover, since for any $s-t$ path q, $\ell_q(g) \geq B_s + B_t$, g is an $7\epsilon_1$-Nash flow in G'. □

Grouping the Neighbors of s and t. Let us now consider the entire network G and the entire subnetwork H_0 of G_0. Lemma 4 can be applied only to subsets of edges in $E_s(H_0)$ and in $E_t(H_0)$ that have (almost) the same latency under f. Since H_0 does not need to be internally complete bipartite, there may be neighbors of s (resp. t) connected to disjoint subsets of V_t (resp. of V_s) in H_0, and thus have quite different latency. Hence, to apply Lemma 4, we partition the neighbors of s and the neighbors of t into classes V_s^i and V_t^j according to their latency. For convenience, we let $\epsilon_2 = \varepsilon/6$, i.e., f is an ϵ_2-Nash flow, and $L \equiv L_{H_0}(f)$. By Theorem 2, applied with error $\epsilon_2 = \varepsilon/6$, there exists a ρ such that for all $e \in E(H_0)$, $0 < f_e \leq \rho + \epsilon_2$. Therefore, $L \leq 2A_{\max}(\rho + \epsilon_2) + 2B_{\max}$ is bounded by a constant.

We partition the interval $[0, L]$ into $\kappa = \lceil L/\epsilon_2 \rceil$ subintervals, where the i-th subinterval is $I^i = (i\epsilon_2, (i+1)\epsilon_2]$, $i = 0, \ldots, \kappa - 1$. We partition the vertices of V_s (resp. of V_t)

that receive positive flow by f into κ classes V_s^i (resp. V_t^i), $i = 0, \dots, \kappa - 1$. Precisely, a vertex $x \in V_s$ (resp. $x \in V_t$), connected to s (resp. to t) by the edge $e_x = \{s, x\}$ (resp. $e_x = \{x, t\}$), is in the class V_s^i (resp. in the class V_t^i), if $\ell_{e_x}(f_{e_x}) \in I_i$. If a vertex $x \in V_s$ (resp. $x \in V_t$) does not receive any flow from f, x is removed from G and does not belong to any class. Hence, from now on, we assume that all neighbors of s and t receive positive flow from f, and that $V_s^0, \dots V_s^{\kappa-1}$ (resp. $V_t^0, \dots, V_t^{\kappa-1}$) is a partitioning of V_s (resp. V_t). In exactly the same way, we partition the edges of E_s (resp. of E_t) used by f into k classes E_s^i (resp. E_t^i), $i = 0, \dots, \kappa - 1$.

To find out which parts of H_0 will be connected through the intermediate subnetwork of G, using the construction of Lemma 4, we further classify the vertices of V_s^i and V_t^j based on the neighbors of t and on the neighbors of s, respectively, to which they are connected by f-used edges in the subnetwork H_0. In particular, a vertex $u \in V_s^i$ belongs to the classes $V_s^{(i,j)}$, for all j, $0 \leq j \leq \kappa - 1$, such that there is a vertex $v \in V_t^j$ with $f_{\{u,v\}} > 0$. Similarly, a vertex $v \in V_t^j$ belongs to the classes $V_t^{(i,j)}$, for all i, $0 \leq i \leq \kappa - 1$, such that there is a vertex $u \in V_s^i$ with $f_{\{u,v\}} > 0$. A vertex $u \in V_s^i$ (resp. $v \in V_t^j$) may belong to many different classes $V_s^{(i,j)}$ (resp. to $V_t^{(i,j)}$), and that the class $V_s^{(i,j)}$ is non-empty iff the class $V_t^{(i,j)}$ is non-empty. We let $k \leq \kappa^2$ be the number of pairs (i, j) for which $V_s^{(i,j)}$ and $V_t^{(i,j)}$ are non-empty. We note that k is a constant, i.e., does not depend on $|V|$ and r. We let $E_s^{(i,j)}$ be the set of edges connecting s to the vertices in $V_s^{(i,j)}$ and $E_t^{(i,j)}$ be the set of edges connecting t to the vertices in $V_t^{(i,j)}$.

Building the Intermediate Subnetworks of G. The last step is to replace the 0-latency simplified parts connecting the vertices of each pair of classes $V_s^{(i,j)}$ and $V_t^{(i,j)}$ in H_0 with a subnetwork of G_m. We partition, as in condition (4) in the definition of good networks, the set V_m of intermediate vertices of G into k subsets, each of cardinality $|V_m|/k$, and associate a different such subset $V_m^{(i,j)}$ with any pair of non-empty classes $V_s^{(i,j)}$ and $V_t^{(i,j)}$. For each pair (i, j) for which the classes $V_s^{(i,j)}$ and $V_t^{(i,j)}$ are non-empty, we consider the induced subnetwork $G^{(i,j)} \equiv G[\{s, t\} \cup V_s^{(i,j)} \cup V_m^{(i,j)} \cup V_t^{(i,j)}]$, which is an $(n/k, p, 1)$-good network, since G is an (n, p, k)-good network. Therefore, we can apply Lemma 4 to $G^{(i,j)}$, with $H_0^{(i,j)} \equiv H_0[\{s, t\} \cup V_s^{(i,j)} \cup V_t^{(i,j)}]$ in the role of H_0, the restriction $f^{(i,j)}$ of f to $H_0^{(i,j)}$ in the role of the flow f, and $\rho' = \rho + \epsilon_2$. Moreover, we let $B_s^{(i,j)} = \max_{e \in E_s^{(i,j)}} \ell_e(f_e)$ and $B_t^{(i,j)} = \max_{e \in E_t^{(i,j)}} \ell_e(f_e)$ correspond to B_s and B_t, and introduce constant latencies $\ell'_e(x) = B_s^{(i,j)}$ for all $e \in E_s^{(i,j)}$ and $\ell'_e(x) = B_t^{(i,j)}$ for all $e \in E_t^{(i,j)}$, as required by Lemma 4. Thus, we obtain, with high probability, a subnetwork $H^{(i,j)}$ of $G^{(i,j)}$ and a flow $g^{(i,j)}$ that routes as much flow as $f^{(i,j)}$ on all edges of $E_s^{(i,j)} \cup E_t^{(i,j)}$, and satisfies the conclusion of Lemma 4, if we keep in $H^{(i,j)}$ the constant latencies $\ell'_e(x)$ for all $e \in E_s^{(i,j)} \cup E_t^{(i,j)}$.

The final outcome is the union of the subnetworks $H^{(i,j)}$, denoted H (H has the latency functions of the original instance G), and the union of the flows $g^{(i,j)}$, denoted g, where the union is taken over all k pairs (i, j) for which the classes $V_s^{(i,j)}$ and $V_t^{(i,j)}$ are non-empty. By construction, all edges of H are used by g. Using the properties of the construction above, we can show that if $\epsilon_1 = \varepsilon/42$ and $\epsilon_2 = \varepsilon/6$, the flow g is an ε-Nash flow of (H, r), and satisfies $L_H(g) \leq L_{H_0}(f) + \varepsilon/2$. Thus, we obtain:

Lemma 5. *Let any $\varepsilon > 0$, let $k = \lceil 12(A_{\max}(\rho + \varepsilon) + B_{\max})/\varepsilon \rceil^2$, let $G(V, E)$ be an (n, p, k)-good network, let $r > 0$, let H_0 be any subnetwork of the 0-latency simplification of G, and let f be an $(\varepsilon/6)$-Nash flow of (H_0, r) for which there exists a constant $\rho' > 0$, such that for all $e \in E(H_0)$, $0 < f_e \leq \rho'$. Then, with high probability, wrt. the random choice of the latency functions of G, we can compute in $\mathrm{poly}(|V|)$ time a subnetwork H of G and an ε-Nash flow g of (H, r) with $L_H(g) \leq L_{H_0}(f) + \varepsilon/2$.*

References

1. Althöfer, I.: On Sparse Approximations to Randomized Strategies and Convex Combinations. Linear Algebra and Applications 99, 339–355 (1994)
2. Bollobás, B.: Random Graphs, 2nd edn. Cambridge Studies in Advanced Mathematics, vol. 73. Cambridge University Press (2001)
3. Braess, D.: Über ein paradox aus der Verkehrsplanung. Unternehmensforschung 12, 258–268 (1968)
4. Chung, F., Young, S.J.: Braess's paradox in large sparse graphs. In: Saberi, A. (ed.) WINE 2010. LNCS, vol. 6484, pp. 194–208. Springer, Heidelberg (2010)
5. Chung, F., Young, S.J., Zhao, W.: Braess's paradox in expanders. Random Structures and Algorithms 41(4), 451–468 (2012)
6. Fotakis, D., Kaporis, A.C., Spirakis, P.G.: Efficient methods for selfish network design. Theoretical Computer Science 448, 9–20 (2012)
7. Kelly, F.: The mathematics of traffic in networks. In: Gowers, T., Green, J., Leader, I. (eds.) The Princeton Companion to Mathematics. Princeton University Press (2008)
8. Lin, H.C., Roughgarden, T., Tardos, É., Walkover, A.: Stronger bounds on Braess's paradox and the maximum latency of selfish routing. SIAM Journal on Discrete Mathematics 25(4), 1667–1686 (2011)
9. Lipton, R.J., Markakis, E., Mehta, A.: Playing Large Games Using Simple Strategies. In: Proc. of the 4th ACM Conference on Electronic Commerce (EC 2003), pp. 36–41 (2003)
10. Lipton, R.J., Young, N.E.: Simple Strategies for Large Zero-Sum Games with Applications to Complexity Theory. In: Proc. of the 26th ACM Symposium on Theory of Computing (STOC 1994), pp. 734–740 (1994)
11. Milchtaich, I.: Network Topology and the Efficiency of Equilibrium. Games and Economic Behavior 57, 321–346 (2006)
12. Nagurney, A., Boyce, D.: Preface to "On a Paradox of Traffic Planning". Transportation Science 39(4), 443–445 (2005)
13. Pas, E.I., Principio, S.L.: Braess's paradox: Some new insights. Transportation Research Part B 31(3), 265–276 (1997)
14. Roughgarden, T.: Selfish Routing and the Price of Anarchy. MIT Press (2005)
15. Roughgarden, T.: On the Severity of Braess's Paradox: Designing Networks for Selfish Users is Hard. Journal of Computer and System Sciences 72(5), 922–953 (2006)
16. Steinberg, R., Zangwill, W.I.: The prevalence of Braess' paradox. Transportation Science 17(3), 301–318 (1983)
17. Valiant, G., Roughgarden, T.: Braess's paradox in large random graphs. Random Structures and Algorithms 37(4), 495–515 (2010)
18. Végh, L.A.: Strongly polynomial algorithm for a class of minimum-cost flow problems with separable convex objectives. In: Proc. of the 44th ACM Symposium on Theory of Computing (STOC 2012), pp. 27–40 (2012)