

Resolving Braess's Paradox in Random Networks

Dimitris Fotakis¹ · Alexis C. Kaporis² ·
Thanasis Lianas¹ · Paul G. Spirakis^{3,4}

Received: 12 September 2014 / Accepted: 8 June 2016 / Published online: 28 June 2016
© Springer Science+Business Media New York 2016

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 Valiant and Roughgarden *RSA '10* (Random Struct Algorithms 37(4):495–515, 2010), Chung and Young *WINE '10* (LNCS 6484:194–208, 2010) and Chung et al. *RSA '12* (Random Struct Algorithms 41(4):451–468, 2012). 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 source s and destination t are directly connected by 0 latency edges.

This work 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", under the research funding program *Thales*, by the EU ERC project *RIMACO*, by EU ERC project *ALGAME* Grant Agreement no. 321171, and by the EU FP7/2007-13 (DG INFSO G4-ICT for Transport) project *eCompass* Grant Agreement no. 288094.

✉ Alexis C. Kaporis
kaporisa@gmail.com

Dimitris Fotakis
fotakis@cs.ntua.gr

Thanasis Lianas
tlianas@mail.ntua.gr

Paul G. Spirakis
P.Spirakis@liverpool.ac.uk

- ¹ Electrical and Computer Engineering, National Technical University of Athens, Athens, Greece
- ² Information and Communication Systems Department, University of the Aegean, Samos, Greece
- ³ Computer Science Department, University of Liverpool, Liverpool, UK
- ⁴ Computer Technology Institute and Press "Diophantus", Patras, Greece

Building on this, we consider two cases, either when the total rate r is sufficiently *low*, or, when r is sufficiently *high*. In the first case of *low* $r = O(n_+)$, here n_+ is the maximum degree of $\{s, t\}$, 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 $O(\text{poly}(\ln n))$. Finally, in the second case of *high* $r = \Omega(n_+)$, we compute in strongly polynomial time a subnetwork and an ε -Nash flow with maximum latency at most $(1 + 2\varepsilon + o(1))L^*$.

Keywords Algorithmic game theory · Braess's paradox · Selfish routing · Wardrop equilibrium · Random graphs

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 infinitesimally small players. A picturesque way to see a large network of links shared by many infinitesimally small selfish users is as a large pipeline infrastructure with users as liquid molecules flowing into it. 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, while the remaining unused paths have higher (unappealing) latency. Under some general assumptions on the latency functions, a Nash equilibrium flow (or simply a *Nash flow*) exists, it is efficiently computable and the common players' latency in a Nash flow is essentially unique (see e.g., [32]).

When the owner of such a selfishly congested network tries to improve its flow speed, the common sense suggests to focus and fix links that seem older and slower. Contrary to this belief, *Braess's paradox* illustrates that *destroying* a part of a network, even of the most expensive infrastructure, can *improve* its performance. So a wise owner should take steps cautiously and benefit by exploiting the nature of this paradox. There are a few natural approaches for improving network performance. A simple approach, not requiring any network modifications, is *Stackelberg* routing. The network owner dictatorially controls a small fraction of flow, aiming to improve the induced routing performance of the remaining selfish flow. Unfortunately, there are examples of unboundedly bad performance under any possible control attempt made by the owner. Another side effect is that the dictatorially controlled flow is usually sacrificed through slower paths, compared to the latency faced by the remaining free flow. An alternative approach is to introduce economic incentives, usually modeled as flow-dependent per-unit-of-flow tolls, that influence the users selfish choices towards improving performance. However, the idea of tolls is not appealing to the users, since large tolls increase the users *disutility*: routing time plus tolls paid, see details in [7].

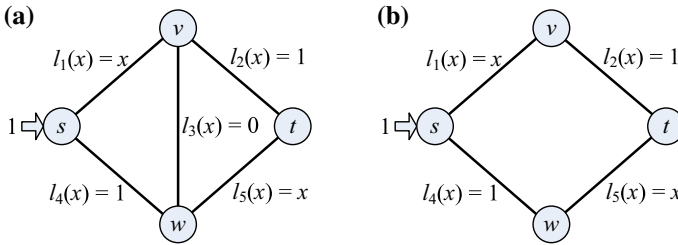


Fig. 1 **a** The optimal total latency is $3/2$, achieved by routing half of the flow on each of the paths (s, v, t) and (s, w, t) . In the (unique) Nash flow, all traffic goes through the path (s, v, w, t) and has a latency of 2. **b** If we remove the edge (v, w) , the Nash flow coincides with the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

A simple and easy to implement way out from the above side effects is to exploit the essence of Braess's paradox towards improving network performance.

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 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) [7] for the use of congested edges, or exploiting the presence of centrally coordinated players (Stackelberg routing) [31], see also [32] for more references. A simple way to improve the network performance at equilibrium is to exploit Braess's paradox [3, 26], namely the fact that removing some edges may improve the latency of the Nash flow (see e.g., Fig. 1 for an example and [27, 33] for more bibliography). 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., [10] for a discussion).

Despite the intense research [30, Sect. 5.1.2] for algorithmically detecting the paradox, little positive results have been shown rigorously. Unfortunately, Roughgarden [33] 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, 21], Fotakis et al. [10] 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. [19] 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. [19] 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 the impact of the network topology, Milchtaich [24] 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. 1a 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., [17, 28, 32, 35]). In this direction, Valiant and Roughgarden [36] 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 [5] extended the result of [36] 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 [6]). Nevertheless, the proof of [5, 6, 36] is merely existential; it provides no clue on how one can actually find (or even approximate) the best subnetwork and its equilibrium latency.

In all the work above, the graph G and the latencies ℓ are random. But, the traffic r is adversarial and selected for the paradox to occur whp. Roughgarden raised the question of *random* traffic $r > 0$, or, investigating the range of r that causes the paradox, citing the works [12, 28] with evidence of r ranges that the paradox is unlikely. A related question is to identify the *vulnerable* network topologies [30, pp. 125–126] that, given a graph G , there is a choice of traffic value r_G and latency functions ℓ_G that cause the Braess paradox to occur. As a sharp contrast, vulnerable graphs are easy [8, 24, 26].

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 [5, 6, 36] 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.

It is well known that a NP-completeness reduction may use complex structures that may rarely occur in generic/realistic instances. NP-completeness focus to the worst-case analysis of a given class of instances, while it provides limited or no information about the algorithmic hardness of the typical (overwhelming majority of) instances. There is a need to get a bigger picture of the complexity landscape. Therefore, a way to widen this limited view of an NP-hard proof, is to suggest the probabilistic analysis of algorithms [14, 16]. Where, a meaningful target is to exhibit that algorithmic hard instances come up often, or show that hard cases are rare, given a distribution that resembles most of the problem's rich landscape. Towards to achieve more insight in the underlying algorithmic complexity for the majority of the instances, random instances are used for evaluating algorithms for NP-hard problems. Random instances are cheap and usually (but not always) lack structures that expose information and facilitate the running time of algorithms, often unavoidably hidden in deterministic instances. Random instances often provide *control parameters* for important char-

acteristics, such as expected hardness and/or (in)solubility, that help to validate and improve sophisticated heuristics [15, 25, 34]. Of course, instances obtained from real-world applications are the best source, albeit of limited supply and sometimes suffer being structured/oriented towards specific applications. On the positive side, there is a wide experience, constantly updated from ongoing competitions [15, 38, 39], illustrating the strong correlation (wrt algorithm performance) between real-world and random instances. Hence, in the last 20 years an area of intense research in *Artificial Intelligence* (AI) [4, 18, 23], *Computer Science* (CS) [11] and *Statistical Mechanics* (SM) [22] has been the typical algorithmic complexity of hard problems wrt the Erdős-Rényi $G_{n,p}$ (or $G(n, m)$) model [2] of random instances.

Departing from [5, 36], 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 [5, 36] 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 [5, 36] 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 [5, 36] 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. In particular, our work exploits algorithmically the paradox down to the *connectivity threshold* $p = \frac{\ln n}{n}$ wrt *control parameter* p of a random $\mathcal{G}_{n,p}$ graph [9]. Our argument relies strongly to the existence of many "short & fast paths" that connect the neighbors of s to the neighbors of t . Since the existence of such paths is critically related to the connectivity threshold, we believe it is also interesting to explore for parameter p ranging below this threshold, whether the paradox can still be efficiently exploited or not. Another source of randomness is the random coefficient model wrt edge latencies. But, our main focus is to assume the same assumptions for the random edge coefficients as in [5, 6, 36]. Of course, if we change the coefficient's distribution it is possible to ruin the existence of such fast paths, despite p ranging above the connectivity threshold.

Technically, we present essentially an approximation scheme. In the first case of *low* $r = O(n_+)$, with n_+ the maximum degree of $\{s, t\}$, 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 $O(\text{poly}(\ln \ln n))$, where n is the number of vertices. Specifically, the running time is polynomial if the good network has average degree $O(\text{poly}(\ln n))$, i.e., if $pn = O(\text{poly}(\ln n))$, 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., [32] 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 [10, Theorem 3] gives an exponential-time algorithm. Finally, in the second case of *high* $r = \Omega(n_+)$, we compute in strongly polynomial time a subnetwork with maximum latency at most $(1 + 2\varepsilon + o(1))L^*$.

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. 1a 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 [20,21] and [10, 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 [5,36], showing that Braess's paradox occurs in good networks with high probability (see [5, 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 5 and 6.

We highlight that the reduction itself runs in polynomial time. The time consuming step is the application of [10, 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 [33]. 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 Erdős-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 , $\ell_e : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ is a non-decreasing latency function associated with each edge e , and $r > 0$ is the traffic rate. A picturesque way to see the total traffic rate r in a large network of links is that there are many infinitesimally small selfish users of total volume r that flow into a large pipeline infrastructure. That is, users are considered as liquid molecules that start to flow from node s through the available pipelines of the smallest latency, trying to reach destination node t . 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 $a_e, b_e \geq 0$. These encapsulate that the time delay (latency), on any edge a particular commuter decides to walk, *increases* when x other commuters also decide to walk along it, with a rate that depends on the road specific characteristics a_e, b_e . 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 [5, 6, 36], we define:

Definition 1 We say that \mathcal{A} and \mathcal{B} are *reasonable* if:

1. \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 $|V|$.
2. There is a closed interval $I_{\mathcal{A}}$ of positive length, such that for every non-trivial subinterval $I' \subseteq I_{\mathcal{A}}$, $\mathbb{P}_{a \sim \mathcal{A}}[a \in I'] > 0$.
3. 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 $(G(V, E), r)$.

Given a network G and a traffic rate r , there are exponentially many subnetworks, each incurring its own common path latency. Therefore the problem of detecting the particular subnetwork that achieves the minimum common path latency is a combinatorial one with exponential worst case complexity.

Flows Given an instance (G, r) , a (feasible) *flow* f is a non-negative vector $\langle f_q : q \in \mathcal{P} \rangle$ indexed by \mathcal{P} such that $\sum_{q \in \mathcal{P}} f_q = r$. That is, f_q is the amount of flow routed from s to t through the links of path $q \in \mathcal{P}$. 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 through all the paths that traverse e . That is, path flow f induces the non-negative vector $\langle f_e : e \in E \rangle$ indexed by 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 .

Our notation is based on the fact that each *path flow* $\langle f_q : q \in \mathcal{P} \rangle$ induces a unique *edge flow* $\langle f_e : e \in E \rangle$, see [29, Th. 2.2]. In general, the converse is not true since in an edge flow it is possible to induce cycles with positive flow, see [29, Sect. 2.2.2]. But, in our case, all edges have strictly increasing latencies, therefore, in (or in a social optimum flow) a Nash equilibrium it is not possible for a positive amount of flow to be trapped in cycles. This nice observation allows us to conveniently interchange between path and edge flows. This nice fact that Nash flows are acyclic and independent of the particular flow decomposition is extensively and implicitly being used in recent works, see for example in [38] Proposition 2.4 and the paragraph above it.

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., [32]). 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., [32]).

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

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 BestSubEL in short. In 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 $V(G), n(G), V_s(G), n_s(G), \dots$, 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 .

Our layered graph construction above allows us to think that each path latency is only contributed by the latency of the edge exiting s plus the edge latency entering to t , while the remaining edges (those not touching s, t) of the path contribute 0 latency. The main concern here is that a path, while exiting V_s and visiting vertices in V_m , is possible to come back and visit again some vertex in $u \in V_s$. This bad scenario can hurt our argument only if this path also sends positive flow back from u to s . In this scenario however, a cycle appears, but, as mentioned above, it is known that an arbitrary Nash equilibrium can be made acyclic with no increase of the common latency. The idea is that a Nash equilibrium is the solution of a convex program and hence, we can remove the flow trapped around a cycle (it important that it traverses edges with strictly increasing latency functions, otherwise the removing of circulated flow would not turn beneficial) without increasing any path latency. See for example the recent work [33] below Proposition 2.3, or, for a detailed exhibition of this argument the nice book of Patriksson [29, Sect. 2.2.2].

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 Erdős-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 $\ell_e(x) = 0$.

Definition 2 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 $|n_s - n_t| \leq 2n_-$.

Definition 3 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 $v \in V$, $|F(v)| \leq 3np/2$.
2. G is an *expander graph*, namely, for any set $S \subseteq V$, $|F(S)| \geq \min\{np|S|, n\}/2$.
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, \dots, 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 .

In our text whenever we wish to give emphasis to these particular 4 properties above that *good networks* possess, we explicitly use the term (n, p, k) -good networks. Our assumption 3 above: $\mathbb{P}[\mathcal{B} \leq \frac{\eta}{\log n}] = \omega(\frac{1}{np})$, for constant $\eta > 0$, is equivalent to the assumption in [5, Corollary 6 and Lemmata 7, 8] requiring that for any small constant $\delta > 0$, there are constants $c > 1$ and $n_0 > 0$ such that for $n > n_0$ to hold $\mathbb{P}[\mathcal{B} \leq \frac{\delta}{\log n}] \geq \frac{c \log n}{np}$. Our assumption 3 also is in comparison to [5, Lemma 5], that requires that for any small constant $\delta > 0$, there are constants $c > 1$ and $n_0 > 0$ such that for $n > n_0$ to hold $\mathbb{P}[\mathcal{B} \leq \frac{\delta}{\log n}] \geq \frac{4}{np}$. It is also helpful for the reader to see our assumption 3 in comparison to [6, Sect. 1.2-1st paragraph] stating that if $pn \geq c \log n$ then the G_{np} graph is an $(\alpha = \frac{3}{2}np, \beta = \frac{1}{4})$ -expander. Therefore in the subsequent paragraph in [6, Sect. 2.2-pp. 457] the 2nd bullet becomes $\mathbb{P}[\mathcal{B} \leq \frac{\delta}{\log n}] > \frac{20}{3np}$. Our assumption 2 above: $\forall S \subseteq V$ it holds $|\Gamma(S)| \geq \frac{1}{2} \min\{np|S|, n\}$ is more relaxed than [5, Lemma 4] stating that for G_{np} graphs with $p \geq \frac{c}{\log n}$ whp $\forall U \subseteq V$ it holds that $|\Gamma(U)| \geq \frac{e-1}{e} \min\{np|U|, n\}$, since $\frac{e-1}{e} > \frac{1}{2}$. Our assumption 1 above: $\forall u \in V$ whp it holds $\Gamma(u) \leq \frac{3}{2}np$ follows from a standard Chernoff bound. Our assumption 4 above: it is easy to see that it holds for G_{np} graphs due to the fact that each edge appears independently and hence each subset $V' \subseteq V$ with $|V'| = n'$ of a G_{np} graph behaves as $G_{n'p}$.

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 [5] 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, see step 1 of Algorithm 1.

Algorithm 1 is based on an approximation-preserving reduction of BestSubEL for a good network G to BestSubEL for the 0-latency simplification G_0 of G . The first step of our approximation-preserving reduction is to show in Lemma 1 in Sect. 4 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. 1a) may trigger Braess’s paradox, we need Lemma 1, in Sect. 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.

Algorithm 1: Approximation Scheme for 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 let $H \equiv G_g \subseteq G$ the g -used subnetwork with g computed by Lemma 5 given H_0 and f computed above ;
- 5 else, let H_0 be the subnetwork and f the $\varepsilon/6$ -Nash flow of Thm. 2 applied with error $\varepsilon/6$;
- 6 let $H \equiv G_g \subseteq G$ the g -used subnetwork with g computed by Lemma 6 given H_0 and f computed above ;
- 7 return the subnetwork H and the ε -Nash flow g ;

Next, we focus on the 0-latency simplification G_0 of G (Definition 2), see step 1 in Algorithm 1. We show that if the traffic rate is large enough, i.e., if $r > (B_{\max}n_+)/(\varepsilon A_{\min})$, the paradox has a marginal influence on the equilibrium latency and can be approximated in step 1 and 1 of Algorithm 1. In particular, in Sect. 6.3, we first consider the Nash flow f on (G_0, r) computed in strongly polynomial time. Then, using Lemma 5 we “extend” f in $poly(|V|)$ time to a corresponding flow g on the g -used subgraph G_g of the random instance G . Flow g satisfies the s, t -link capacity constraints imposed by f , while being the minimizer of a potential over all G and, hence, g is a Nash flow on the remaining g -used subgraph $G_g \subseteq G$ (after discarding all the empty paths of G). Furthermore, if $L(H^*) = \omega(1)$ then g approximates within $(1 + \varepsilon + o(1))$ the BestSubEL, and if $L(H^*) = O(1)$ then g approximates within $(1 + 2\varepsilon + o(1))$ the BestSubEL (Remark 2). On the other hand, if $r \leq (B_{\max}n_+)/(\varepsilon A_{\min})$ we work as step 5 and 6 of Algorithm 1. In Sect. 6.2 we use [10, Theorem 3] restated as Theorem 2 here, and we obtain (within the time bounds of this theorem) a subnetwork H_0 and an $\varepsilon/6$ -Nash flow f that comprise a good approximate solution to 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 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 [5, 36] 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 5, the proof draws on ideas from [5, 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 6). 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 6).

Theorem 1 *Let $G(V, E)$ be an (n, p, k) -good network (Definition 3), with $k \geq 1$ is a large enough constant.*

- Let $r \leq \frac{B_{\max} n_+}{A_{\min} \varepsilon}$ be any traffic rate. Let H^* be the best subnetwork of (G, r) . Then, for any $\varepsilon > 0$, Algorithm 1 computes in time $n_+^{O(r^2 A_{\max}^2 \ln(n_+)/\varepsilon^2)}$ $\text{poly}(|V|)$, 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 common path latency $L(g) \leq (1 + \varepsilon)L(H^*) + \varepsilon$.
- Let $r > \frac{B_{\max} n_+}{A_{\min} \varepsilon}$ be any traffic rate. Then, for any $\varepsilon > 0$, Algorithm 1 computes in strongly polynomial time 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 common path latency $L(g) \leq (1 + 2\varepsilon + o(1))L(H^*)$.

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., [32]. Thus, if $r = O(1)$ (or more generally, if $r = O(\text{poly}(\ln \ln n))$) and $pn = O(\text{poly}(\ln n))$, in which case $n_+ = O(\text{poly}(\ln n))$, Theorem 1 gives a randomized polynomial-time approximation scheme for BestSubEL in good networks. Moreover, the running time is quasipolynomial for traffic rates up to $O(\text{poly}(\ln n))$ and average degrees up to $o(n)$, i.e., for the entire range of p in [5, 36]. The next sections are devoted to several lemmas and theorems that are useful and combined together in Sects. 6.2 and 6.3 for achieving the corresponding approximation for low and high values of r

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 Sect. 2. Lemma 1 will be important in the proof of Lemmata 2-4, since we can work directly in the 0-latency simplification G_0 of G .

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 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 parts.

1st Part 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 also show how to further simplify H_1 so that its intermediate bipartite subnetwork becomes acyclic.

To construct the simplification H_1 of H , we let f be the equilibrium flow of H , and let $L \equiv L(H, r)$. For each $u_i \in \Gamma(s)$ and $v_j \in \Gamma(t)$, we let $f_{ij} = \sum_{p=(s, u_i, \dots, v_j, t)} f_p$ be the flow routed by f from u_i to v_j . The network H_1 is obtained from H by replacing the intermediate subnetwork of H with a bipartite subnetwork connecting $\Gamma(s)$

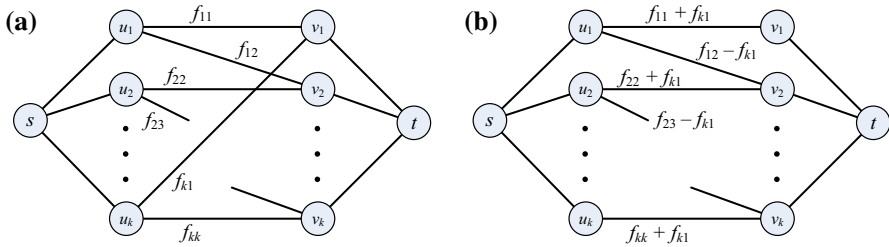


Fig. 2 In **a** we have a cycle $C = (u_1, v_2, u_2, \dots, v_k, u_k, v_1, u_1)$ in the intermediate subnetwork $H_1[\Gamma(s) \cup \Gamma(t)]$. We assume that f_{k1} is the minimum amount flow through an edge of C in the equilibrium flow f . In **b** we have removed the edge e_{k1} , and show the corresponding change in the amount of flow on the remaining edges of C . Since the latency functions of the edges in C are constant, the change in the flow does not affect equilibrium

and $\Gamma(t)$ with constant latency edges. More specifically, instead of the intermediate subnetwork of H , for each $u_i \in \Gamma(s)$ and $v_j \in \Gamma(t)$ with $f_{ij} > 0$, we have an edge $\{u_i, v_j\}$ of constant latency $b_{ij} = L - (a_{\{s,u_i\}} f_{\{s,u_i\}} + b_{\{s,u_i\}}) - (a_{\{v_j,t\}} f_{\{v_j,t\}} + b_{\{v_j,t\}})$ (the corresponding a_{ij} is set to 0). If $f_{ij} = 0$, u_i and v_j are not connected in H_1 . We note that by construction, H_1 admits f as an equilibrium flow, and thus $L(H_1, r) = L$.

Furthermore, we modify H_1 by deleting some edges from its intermediate subnetwork so that the induced bipartite subgraph $H_1[\Gamma(s) \cup \Gamma(t)]$ becomes acyclic. Therefore, in the resulting network, for each $u_i \in \Gamma(s)$ and each $v_j \in \Gamma(t)$, there is at most one (s, u_i, v_j, t) path in H_1 . Hence, the resulting network admits a unique equilibrium flow with a unique path decomposition.

To this end, let us assume that there is a cycle $C = (u_1, v_2, u_2, \dots, v_k, u_k, v_1, u_1)$ in the intermediate subnetwork $H_1[\Gamma(s) \cup \Gamma(t)]$. We let $e_{k1} = \{u_k, v_1\}$ be the edge of C with the minimum amount of flow in f , and let f_{k1} be the flow through e_{k1} (see also Fig. 2). Then, removing e_{k1} , and updating the flows along the remaining edges of C so that $f'_{ii} = f_{ii} + f_{k1}$, $1 \leq i \leq k$, and $f'_{i(i+1)} = f_{i(i+1)} - f_{k1}$, $1 \leq i \leq k - 1$, we “break” the cycle C , by eliminating the flow in e_{k1} , and obtain a new equilibrium flow f' of the same rate r and with the same latency L as that of f . Applying this procedure repeatedly to all cycles, we end up with an internally bipartite network H_1 with an acyclic intermediate subnetwork that includes constant latency edges only. Moreover, H_1 admits an equilibrium flow f of latency L . This concludes the first part of the proof.

2nd Part 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 \mathbf{F} be the $|\mathcal{P}| \times |\mathcal{P}|$ matrix, indexed by the paths $q \in \mathcal{P}$, where $\mathbf{F}[q_1, q_2] = \sum_{e \in q_1 \cap q_2} a_e - \sum_{e \in q_1 \cap q_{kl}} a_e$, and let \mathbf{r} be the vector of r_q 's. Then, the q -th component of $\mathbf{F}\mathbf{r}$ is equal to $\ell_q(f) - \ell_q(f')$. In the following, we consider two cases depending on whether \mathbf{F} is singular or not.

If matrix \mathbf{F} is non-singular, the linear system $\mathbf{F}\mathbf{r} = \varepsilon \mathbf{I}$ has a unique solution \mathbf{r}_ε , for any $\varepsilon > 0$. Moreover, due to linearity, for any $\alpha \geq 0$, the unique solution of the system $\mathbf{F}\mathbf{r} = \alpha \varepsilon \mathbf{I}$ is $\alpha \mathbf{r}_\varepsilon$. Therefore, for an appropriately small $\varepsilon > 0$, the linear system $\mathcal{Q}_\varepsilon = \{\mathbf{F}\mathbf{r} = \varepsilon \mathbf{I}, f_q - r_q \geq 0 \ \forall q \in \mathcal{P}, f_{kl} + \sum_q r_q \geq 0, \ell_{q_{kl}}(f') \leq L + b_{kl} - \varepsilon\}$ admits a unique solution \mathbf{r} . We keep increasing ε until one of the inequalities of \mathcal{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 .

If \mathbf{F} is singular, proceeding similarly, we compute r_p '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 . When \mathbf{F} is singular, the homogeneous linear system $\mathbf{F}\mathbf{r} = \mathbf{0}$ admits a nontrivial solution $\mathbf{r} \neq \mathbf{0}$. Moreover, due to linearity, for any $\alpha \in \mathbb{R}$, $\alpha \mathbf{r}$ is also a solution to $\mathbf{F}\mathbf{r} = \mathbf{0}$. Therefore, the linear system $\mathcal{Q}_0 = \{\mathbf{F}\mathbf{r} = \mathbf{0}, f_p - r_p \geq 0 \ \forall p \in \mathcal{P}, f_{kl} + \sum_p r_p \geq 0\}$ admits a solution $\mathbf{r} \neq \mathbf{0}$ that makes at least one of the inequalities tight and has $\ell_{q_{kl}}(f') \leq L + b_{kl}$.¹ We recall that the p -th component of $\mathbf{F}\mathbf{r}$ is equal to $\ell_p(f) - \ell_p(f')$. Therefore, for the flow f' obtained from the particular solution \mathbf{r} of \mathcal{Q}_0 , the latency of any path $p \in \mathcal{P}$ is equal to L . If \mathbf{r} is such that $r_p = f_p$ for some path $p = (s, u_i, v_j, t) \in \mathcal{P}$, 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 \mathbf{r} is such that $\sum_p r_p = -f_{kl}$, 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

¹ If the direction of a solution r_0 gives increase to $\ell_{q_{kl}}(f')$ then, because of linearity, the direction of $-r_0$ decreases it. So we can assume that we can hit one of the constraints and have $\ell_{q_{kl}}(f') \leq L + b_{kl}$ (e.g. by choosing the direction that decreases $\ell_{q_{kl}}(f')$). Note that in both directions of r_0 and $-r_0$ we are bounded by one of the constraints because, in any direction, either some r_q would be increasing and $f_q - r_q \geq 0$ will be bounding us, or all r_q 's would be decreasing and $f_{kl} + \sum_q r_q \geq 0$ will be bounding us.

than H_1 . Moreover, we can show (see Property 1 below) 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 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 $b'_{kl} = 0$. So, by repeatedly applying these steps, we end up with a subnetwork H' of the 0-latency simplification of H with $L(H', r) \leq L(H, r)$. \square

Property 1 *If the path p_{kl} is disjoint to the paths $p \in \mathcal{P}$, the matrix F is positive definite, and thus non-singular.*

Proof We first note that if p_{kl} is disjoint to all $p \in \mathcal{P}$, then for all $p_1, p_2 \in \mathcal{P}$, $F[p_1, p_2] = \sum_{e \in p_1 \cap p_2} a_e$. Hence, for all $x \in \mathbb{R}^{|\mathcal{P}|}$, $x^T F x = \sum_{e \in E(\mathcal{P})} a_e x_e^2 \geq 0$, where $E(\mathcal{P})$ denotes the set of edges included in the paths of \mathcal{P} and $x_e = \sum_{p: e \in p} x_p$. Since the intermediate network of H_1 is acyclic and any flow in H_1 has a unique path decomposition, if x has one or more non-zero components, there is at least one edge e adjacent to either s or t such that $x_e > 0$, and thus $x^T F x > 0$. Otherwise, the difference of the flow defined by x with the trivial flow defined by $\mathbf{0}$ would indicate the existence of a cycle in the intermediate subnetwork of H_1 . This is a contradiction, since by the first part of the proof, the intermediate part of H_1 is acyclic. \square

5 Approximating the Best Subnetwork of Simplified Networks

We proceed to show in Theorem 2 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 . Before proving this theorem, we first prove two useful lemmas, that is, Lemma 2 about the maximum traffic rate r up to which BestSubEL remains interesting, and Lemma 3 about the maximum amount of flow routed on any edge/path in the best subnetwork. The combination of these two lemmas readily yields the proof of Lemma 4, concluding that the best subnetwork of any simplified instance (G_0, r) routes $O(1)$ units of flow on any used edge and on any used path. This is the “missing tile” for finally proving Theorem 2.

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 we assume arbitrarily small constant $0 < \varepsilon < 1$, then $r > \frac{B_{\max} n_+}{A_{\min}}$, $L_1, L_2 > B_{\max}$ and all edges in $E_s \cup E_t$ are used by f .

A useful property is that $\exists e \in E_s$ with $a_e f_e \leq r/A_s$ and similarly, $\exists e \in E_t$ with $a_e f_e \leq r/A_t$. To reach a contradiction, assume this is not true: $\forall e \in E_s$ it holds

$f_e > \frac{r}{a_e A_s}$ and: $\forall e \in E_t$ it holds $f_e > \frac{r}{a_e A_t}$. But, this contradicts the total r of the s -links flow, since $r = \sum_{e \in E_s} f_e > \sum_{e \in E_s} \frac{r}{a_e A_s} = \frac{r}{A_s} \sum_{e \in E_s} \frac{1}{a_e} = \frac{r}{A_s} A_s = r$ (similar is the omitted contradiction for r on the t -links). Since all s -links with positive flow must have equal edge latency L_1 , due to this particular s -link e , it follows that $L_1 = a_e f_e + b_e \leq r/A_s + b_e \leq r/A_s + B_{\max}$. Similarly, since all t -links with positive flow must have equal edge latency L_2 , due to this particular t -link e , it follows that $L_2 = a_e f_e + b_e \leq r/A_t + b_e \leq r/A_t + B_{\max}$. Thus,

$$L = L_1 + L_2 \leq \left(\frac{r}{A_s} + \frac{r}{A_t} \right) + 2B_{\max} \tag{1}$$

On the other hand, consider the best subnetwork H_0^* with Nash common path latency $L(H_0^*, r)$ and hence, with cost $r \times L(H_0^*, r)$. Let OPT the cost of the *optimum*² flow on H_0^* , which by definition it holds

$$rL(H_0^*, r) \geq OPT$$

Note that OPT is at least the cost OPT' of separately assigning optimally the flow r on s -links (considered as parallel links), plus, the cost of separately assigning the flow r on t -links (considered as parallel links). Because we optimize the *same objective* $\sum_{e \in E_s \cup E_t} x_e(a_e \cdot x_e + b_e)$ consisting only of the s, t -links (since the intermediary links have 0-latency) *without* imposing the flow constraints for the intermediary paths of H_0^* that connect $\Gamma(s)$ to $\Gamma(t)$. In symbols:

$$OPT \geq OPT' \equiv \min \left\{ \sum_{e \in E_s \cup E_t} x_e(a_e \cdot x_e + b_e) \right\}$$

$$\text{s.t. } \sum_{e \in E_s} x_e = r, \sum_{e \in E_t} x_e = r \tag{2}$$

In turn, OPT' is at least the cost OPT'' of the optimum flow assignment as above, but, assuming now that each s, t -link e has $b_e = 0$. This holds because we change the objective $C'(x) = \sum_{e \in E_s \cup E_t} x_e(a_e \cdot x_e + b_e)$ of OPT' into the objective $C''(x) = \sum_{e \in E_s \cup E_t} x_e(a_e \cdot x_e)$ of OPT'' , which trivially gives better solution, since $\forall x$ it holds $C'(x) \geq C''(x)$. In symbols:

$$OPT' \geq OPT'' \equiv \min \left\{ \sum_{e \in E_s \cup E_t} x_e(a_e \cdot x_e) \right\}$$

$$= \min \left\{ \sum_{e \in E_s} x_e(a_e \cdot x_e) + \sum_{e \in E_t} x_e(a_e \cdot x_e) \right\}$$

$$\text{s.t. } \sum_{e \in E_s} x_e = r, \sum_{e \in E_t} x_e = r \tag{3}$$

² Recall Sect. 1, paragraph *Previous Work*, that for the optimum cost, we compute the flow X that minimizes the total latency cost $C(X) = \sum_{e \in E(H_0^*)} x_e(a_e \cdot x_e + b_e)$ incurred by all users on network H_0^* wrt flow X .

But, this OPT'' is an instance of affine parallel s -links (similarly, t -links) and the optimal load $\forall e \in E_s$ is $o_e = \frac{r}{a_e A_s}$ (similarly, for the t -links, $o_e = \frac{r}{a_e A_t}$), inducing optimum cost per s -link $o_e \ell_e(o_e) = \frac{r}{a_e A_s} a_e \frac{r}{a_e A_s} = \frac{r}{a_e A_s} \frac{r}{A_s}$ and cost over all s -links $\sum_{e \in E_s} \frac{r}{a_e A_s} \frac{r}{A_s} = r \frac{r}{A_s}$. Similarly, we get for the t -links the optimum cost is $r \frac{r}{A_t}$. Therefore $OPT'' = r \left(\frac{r}{A_s} + \frac{r}{A_t} \right)$. The above series of inequalities establish that:

$$rL(H_0^*, r) \geq OPT \geq OPT' \geq OPT'' = r \left(\frac{r}{A_s} + \frac{r}{A_t} \right)$$

which readily implies that

$$L(H_0^*, r) \geq \left(\frac{r}{A_s} + \frac{r}{A_t} \right) \tag{4}$$

otherwise the Nash flow on H_0^* would improve the cost of the optimum flow on H_0^* , a contradiction. Now, recall our assumption that $r > \frac{B_{\max} n_+}{A_{\min} \varepsilon}$ and note also that $A_s = \sum_{e \in E_s} \frac{1}{a_e} \leq \sum_{e \in E_s} \frac{1}{A_{\min}} = n_s \frac{1}{A_{\min}}$ and similarly that $A_t \leq n_t \frac{1}{A_{\min}}$. Plugging these into the above inequality, we get:

$$L(H_0^*, r) \geq \frac{r}{A_s} + \frac{r}{A_t} \geq \frac{B_{\max} n_s}{A_{\min} \varepsilon} \frac{A_{\min}}{n_s} + \frac{B_{\max} n_t}{A_{\min} \varepsilon} \frac{A_{\min}}{n_t} \geq 2B_{\max} / \varepsilon, \tag{5}$$

therefore $2B_{\max} \leq \varepsilon L(H_0^*, r)$. Plugging this and (17) into (1) we conclude:

$$L \leq (1 + \varepsilon)L(H_0^*, r)$$

□

We proceed to show that in a 0-latency simplified instance (G_0, r) , the best subnetwork Nash flow routes $O(r/n_+)$ units of flow on any edge and on any $s - t$ path with high probability (where the probability is with respect to the random choice of the latency function coefficients). Intuitively, we show that in the best subnetwork Nash flow, with high probability, all used edges and all used $s - t$ paths route a volume of flow not significantly larger than their fair share. We first prove the following technical lemma. Recall that we assume below that $L(G_0, r) \geq \varepsilon$, because otherwise it becomes trivial to ε -approximate the BestSubEL problem.

Lemma 3 *Given a random instance G and total flow r , let the balanced 0-latency simplified network G_0 and the Nash flow on (G_0, r) with common path latency $L(G_0, r) \geq \varepsilon > 0$. Then the Nash flow of the best subnetwork H_0^* of G_0 whp³ induces edge load $\leq \frac{24 A_{\max} r}{\delta_\varepsilon A_{\min} n_+}$ on each edge, with $\delta_\varepsilon > 0$ a constant that depends on $\varepsilon > 0$ and the reasonable input distribution of Definition 1(3).*

³ With probability at least $1 - e^{-\delta_\varepsilon n - 8}$.

Proof Given a random instance G and flow $r > 0$ construct the the 0-latency simplified network G_0 of G . That is, G_0 consists of the random snapshot of s -links and t -links that are realized in random instance G . Also G_0 contains the construction of the complete bipartite subnetwork that connects with 0-latency links each neighbor of s to all the neighbors of t . We let $L \equiv L(G_0, r) \geq \varepsilon > 0$ denote the unknown equilibrium latency and g denote a Nash flow of the original instance (G_0, r) . We wish to bound L whp as a function of the given total flow r . Since G_0 is a 0-latency simplified network and g is a Nash flow of (G_0, r) , there are $L_1, L_2 > 0$, with $L_1 + L_2 = L$, such that:

(i) for any edge e incident to s , if $b_e < L_1, g_e > 0$ and $a_e g_e + b_e = L_1$, while $g_e = 0$, otherwise,

(ii) for any edge e incident to t , if $b_e < L_2, g_e > 0$ and $a_e g_e + b_e = L_2$, while $g_e = 0$, otherwise.

Namely, all used edges incident to s (resp. to t) have latency L_1 (resp. L_2) in the Nash flow g . Wlog., we assume that $L_1 \geq L_2$, and thus, $L_1 \geq L/2 \geq \varepsilon/2$.

Inequality (6) below gives the lower bound $\frac{L}{4A_{\max}}$ on the load of each random s -link e that has the nice random property $b_e \leq \varepsilon/4$. Let e be any edge incident to s with $b_e \leq \varepsilon/4$. By the discussion above, in the Nash flow g of (G_0, r) , $g_e > 0$ and $a_e g_e + b_e = L_1$. Using that $L_1 \geq L/2 \geq \varepsilon/2$, we obtain that:

$$L_1 = a_e g_e + b_e \leq a_e g_e + \varepsilon/4 \Rightarrow g_e \geq \frac{L_1 - \varepsilon/4}{a_e} \geq \frac{L_1}{2a_e} \geq \frac{L}{4A_{\max}} \tag{6}$$

In the sequel, a Chernoff bound (e.g., [13, (7)]) establishes that whp there are at least $\delta_\varepsilon n_s/2$ such s -links with the property of $\geq \frac{L}{4A_{\max}}$ load per link, for an appropriate constant $\delta_\varepsilon > 0$ that depends on $\varepsilon > 0$ and Definition 1(3). In particular, from Definition 1(3), there exists a constant $\delta_\varepsilon > 0$ such that $\mathbb{P}[\mathcal{B} \leq \varepsilon/4] \geq \delta_\varepsilon$ and Chernoff bounds yield:

$$\mathbb{P}[|\{e \in E_s(G_0) \text{ with } b_e \leq \varepsilon/4\}| \geq \delta_\varepsilon n_s/2] \geq 1 - e^{-\delta_\varepsilon n_s/8} \tag{7}$$

Therefore the total of accumulated load on these s -links whp is at least $\delta_\varepsilon n_s/2 \times \frac{L}{4A_{\max}}$, which, of course is upper bounded by our fixed total flow r , that is, $\frac{\delta_\varepsilon n_s L}{8A_{\max}} \leq r$. Solving this wrt L , gives:

$$L \leq \frac{8A_{\max} r}{\delta_\varepsilon n_s} \leq \frac{24A_{\max} r}{\delta_\varepsilon n_+} \tag{8}$$

The last inequality holds because G_0 is balanced, and $|n_s - n_t| \leq 2n_-$.

To conclude the proof, observe that in the equilibrium flow f of the best subnetwork H_0^* of G_0 , no used s, t -link e has edge latency greater than the common path latency L of g on G_0 , which is bounded as in (8). Therefore, for any used edge e incident to either s or t , it holds:

$$a_e f_e + b_e \leq L \Rightarrow f_e \leq \frac{L}{a_e} \leq \frac{L}{A_{\min}} \leq \frac{24A_{\max}}{\delta_\varepsilon A_{\min}} \times \frac{r}{n_+} \tag{9}$$

where the last inequality follows from (8). Moreover, any edge e in the intermediate subnetwork of G has $f_e \leq L/A_{\min}$ due to the flow conservation constraints. \square

Remark 1 Lemma 2 shows that the interesting case for the paradox is for $r \leq \frac{B_{\max}n_+}{A_{\min}\varepsilon}$, in this case we combine Lemma 4 and Theorem 2 below and finally proceed to approximate the best subnetwork in Sect. 6.2. On the other hand, if $r > \frac{B_{\max}n_+}{A_{\min}\varepsilon}$ we combine the ideas of Lemma 2 and Lemma 3 and proceed to approximate the best subnetwork in Sect. 6.3.

Lemma 4 *Given a random instance G and total flow $0 < r \leq \frac{B_{\max}n_+}{A_{\min}\varepsilon}$, let the balanced 0-latency simplified network G_0 and the Nash flow on (G_0, r) with common path latency $L(G_0, r) \geq \varepsilon > 0$. Then the Nash flow of the best subnetwork H_0^* of G_0 whp⁷ induces edge load $\varrho \leq \frac{24A_{\max}B_{\max}}{\delta_\varepsilon A_{\min}^2\varepsilon}$ on each edge, with $\delta_\varepsilon > 0$ a constant that depends on $\varepsilon > 0$ and the reasonable distribution of Definition 1(3).*

Proof Recall that we assume above that $L(G_0, r) \geq \varepsilon$, because otherwise it becomes trivial to ε -approximate the BestSubEL problem. Moreover, by Definition 1(3) of reasonable latency functions, we have that for any constant $\varepsilon > 0$, there is a constant $\delta_\varepsilon > 0$, such that $\mathbb{P}[\mathcal{B} \leq \varepsilon/4] \geq \delta_\varepsilon$. Combining these with Lemmas 2 and 3, we obtain Lemma 4. \square

So from now on, we can assume, with high probability and wlog., that the Nash flow in the best subnetwork of any simplified instance (G_0, r) routes $O(1)$ units of flow on any used edge and on any used path.

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 of a random instance G with reasonable latencies and let the total rate $0 < r \leq \frac{B_{\max}n_+}{A_{\min}\varepsilon}$. 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 $H_0 \subseteq G_0$ 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 $0 < \varrho \leq \frac{24A_{\max}B_{\max}}{\delta_\varepsilon A_{\min}^2\varepsilon}$, such that $f_e \leq \varrho + \varepsilon$, for all e .*

Theorem 2 is a corollary of [10, Theorem 3] (depicted as Theorem 3 below), 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 [10, Theorem 3], we have $d_1 = 2, d_2 = 0, \alpha = A_{\max}$, and the error is ε/r . Moreover, we know from Lemma 4 above that any Nash flow g of (H_0^*, r) routes $g_e \leq \varrho$ units of flow on any edge e , and that in the exhaustive search step, in the proof of [10, Theorem 3], one of the acceptable flows f has $|g_e - f_e| \leq \varepsilon$, for all edges e (see also [10, Lemma 3]). Thus, there is an acceptable flow f with $f_e \leq \varrho + \varepsilon$, for all edges e . In fact, if among all acceptable flows enumerated in the proof of [10, Theorem 3], we keep the acceptable flow f that minimizes the maximum amount flow routed on any edge, we have that $f_e \leq \varrho + \varepsilon$, for all edges e .

Theorem 3 *Let $\mathcal{G} = (G(V, E), (a_e x + b_e)_{e \in E}, 1)$ be an instance with linear latency functions, let $\alpha = \max_{e \in E} \{a_e\}$, and let H^B be the best subnetwork of G . For some*

constants d_1, d_2 , let $|\mathcal{P}| \leq m^{d_1}$ and $|p| \leq \log^{d_2} m$, for all $p \in \mathcal{P}$. Then, for any $\varepsilon > 0$, we can compute in time

$$m^{O(d_1 \alpha^2 \log^{2d_2+1}(2m)/\varepsilon^2)}$$

a flow \tilde{f} that is an ε -Nash flow on $G_{\tilde{f}}$ and satisfies $\ell_p(\tilde{f}) \leq L(H^B) + \varepsilon/2$, for all paths p in $G_{\tilde{f}}$.

6 Extending the Solution to the Good Network

First, in Sect. 6.2, we consider sufficiently low values of r and 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 of this extension 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 u - v paths of G_m , for any $u \in V_s$ and $v \in V_t$. In Sect. 6.1 and particularly in Lemma 5, we 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 5 below as some small parts of the original network G and of the actual subnetwork H_0 of G_0 . Thus, we obtain Lemma 5, the building block for proving the more general Lemma 6 in Sect. 6.2, that helps to match the neighbors of s of almost equal s -link latency to the corresponding neighbors of t of almost equal t -link latency.

Finally, in Sect. 6.3, we show how to compute in strongly polynomial time a subnetwork $G' \subseteq G$ that approximates the best subnetwork.

6.1 Bounding the Latency of the Intermediary Paths for Any $r > 0$

Lemma 5 *Let instance (G, r) with G an $(n, p, 1)$ -good network (Definition 3) with the relaxed degree bound $n_+ \leq 3knp/2$, for some constant $k > 0$ and let $r > 0$ be any traffic rate. Assume that $\exists B_s, B_t \geq 0 : \forall e \in E_s \Rightarrow \ell_e(x) = B_s$ and $\forall e \in E_t \Rightarrow \ell_e(x) = B_t$. Consider the 0-latency simplification G_0 of G . Let any $H_0 \subseteq G_0$ endowed with a flow f on (H_0, r) that satisfies an edge load bound $0 < \rho' \leq r$, that is, $\forall e \in E(H_0) : f_e > 0 \Rightarrow f_e \leq \rho'$.*

Then, for any constant $\epsilon_1 > 0$, whp we can compute in time $\text{poly}(|V|)$ a subnetwork $G' \subseteq G$ and a flow g on (G', r) with the properties:

1. $E_s(G') = \{e \in E_s(H_0) : f_e > 0\}$ and $E_t(G') = \{e \in E_t(H_0) : f_e > 0\}$.
2. $\forall e \in E_s(G') \cup E_t(G') \Rightarrow g_e = f_e > 0$.
3. $E_m(G') = \{e \in E_m(G) : g_e > 0\}$.
4. Flow g can be regarded as a Nash flow on G' for any pair $u \in V_s(G')$ and $v \in V_t(G')$ connected by g -used paths.

5. Each g -used path $q = (s, u, \dots, v, t)$ in G' has s - t path latency:

$$\begin{aligned} \ell_q(g) \leq B_s + B_t + 6\epsilon_1 + \rho' \frac{8A_{\max}}{P_b(\epsilon_1)} \left[\frac{2}{np} \left(1 + \frac{6k}{P_b\left(\frac{\epsilon_1}{\ln n}\right)} \right) \right. \\ \left. + \frac{9kp}{2P_b(\epsilon_1)} \right] = B_s + B_t + 6\epsilon_1 + \rho' \times o(1) \end{aligned}$$

Proof We will construct G' as the subnetwork G_g containing only the edges that receive positive load by the flow g computed below. Therefore property (3) above will be satisfied by our construction. For each $e \in E_s \cup E_t$, we set the capacity constraint $g_e = f_e$. Therefore the flow g satisfies property (1) above by construction, that is, $E_s(G')$, $E_t(G')$ contain only the s , t -links that have positive load wrt f .

We compute the extension of g through G_m as an “almost” Nash flow in the modified version G' of G , where each edge $e \in E_s \cup E_t$ has the capacity constraint $g_e = f_e$, therefore property (2) above is satisfied as well. Also, we set the 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 no capacity constraint and have (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., [37]). The subnetwork G' of G is simply G_g , namely, the subnetwork that includes only the edges that receive positive flow 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 $u \in V_s(G')$ and $v \in V_t(G')$, and any pair of $s - t$ paths q, q' going through u and v , if $g_q > 0$, then ⁴ $\ell_q(g) \leq \ell_{q'}(g)$. Thus, g can be regarded as a Nash flow for any pair $u \in V_s(G')$ and $v \in V_t(G')$ connected by g -used paths, which proves property (4) above.

It remains to prove property (5) above that upper bounds the path latency of any g -used path. Towards this, we adjust the proof of [5, Lemma 5] in Proposition 1 below. To prove this, we let $p = (s, u, \dots, v, t)$ be the $s - t$ path used by g that maximizes $\ell_p(g)$. We show the existence of a path $p' = (s, u, \dots, v, t)$ in G of latency $\ell_{p'}(g) \leq B_s + B_t + 6\epsilon_1 + \rho' \times o(1)$. Therefore, since g is a minimizer of $\text{Pot}(g)$, the latency of the maximum latency g -used path p , and thus the latency of any other g -used $s - t$ path, is at most $B_s + B_t + 6\epsilon_1 + \rho' \times o(1)$.

Proposition 1 For any s - t path q used by g it holds $\ell_q(g) \leq B_s + B_t + 6\epsilon_1 + \rho' \times o(1)$.

Proof Let $p = (s, u, \dots, v, t)$ be the $s - t$ path used by g that maximizes $\ell_p(g)$. To show the existence of a path $p' = (s, u, \dots, v, t)$ in G of latency $\ell_{p'}(g) \leq B_s + B_t + 6\epsilon_1 + \rho' \times o(1)$, we start from $S_0 = \{u\}$ and grow a sequence of vertex sets $S_0 \subseteq S_1 \subseteq \dots \subseteq S_{i^*}$, stopping when $|\Gamma(S_{i^*})| \geq 3n/5$ for the first time. We use the

⁴ This holds because the flow travelling from u to v inside G faces no capacities. For this flow and the edges used by it in G , a new minimization problem similar to the one above could be defined. The objective would be the same and the flows other than the one going from u to v would be handled as constants. A solution to this problem could be used to minimize the initial objective and that's why the initial problem returns a solution that solves also the new problem, which in turn has an equilibrium flow as a solution.

expansion properties of G , and condition (3), on the distribution of \mathcal{B} , in the definition of good networks, and show that these sets grow exponentially fast, and thus, $i^* \leq \ln n$, with high probability. Moreover, we show that there are edges of latency $\epsilon_1 + o(1)$ from $S_0 = \{u\}$ to each vertex of S_1 , and edges of latency $\epsilon_1 / \ln n + o(1 / \ln n)$ from S_i to each vertex of S_{i+1} , for all $i = 1, \dots, i^* - 1$. To see this, the intuition is that if among the edges e incident to $V_s \cup V_t$, we keep only those with $b_e \leq \epsilon_1$, and among all the remaining edges e , we keep only those with $b_e \leq \epsilon_1 / \ln n$, then due to condition (3) on the distribution of \mathcal{B} , a good network G remains an expander. Thus, there is a path of latency at most $2\epsilon_1 + o(1)$ from u to each vertex of S_{i^*} . Similarly, we start from $T_0 = \{v\}$ and grow a sequence of vertex sets $T_0 \subseteq T_1 \subseteq \dots \subseteq T_{j^*}$, stopping when $|\Gamma(T_{j^*})| \geq 3n/5$ for the first time. By exactly the same reasoning, we establish the existence of a path of latency at most $2\epsilon_1 + o(1)$ from each vertex of T_{j^*} to v . Finally, since $|\Gamma(S_{i^*})| \geq 3n/5$ and $|\Gamma(T_{j^*})| \geq 3n/5$, the neighborhoods of S_{i^*} and T_{j^*} contain at least $n/10$ vertices in common. With high probability, most of these vertices can be reached from S_{i^*} and from T_{j^*} using edges of latency $\epsilon_1 + o(1)$. Putting everything together, we find a $u - v$ path (in fact, many of them) of length $O(\ln n)$ and latency at most $6\epsilon_1 + o(1) \leq 7\epsilon_1$.

For completeness, we next give a detailed proof, by adjusting the arguments in the proof of [5, Lemma 5]. For convenience, for each vertex x , we let $d_s(x)$ (resp. $d_t(x)$) be the latency wrt g of the shortest latency path from s to x (resp. from x to t). Also, for any $\delta > 0$, we let $P_b(\delta) \equiv \mathbb{P}[\mathcal{B} \leq \delta]$ denote the probability that the additive term of a reasonable latency is at most δ . Recall also that by hypothesis, there exists a constant $\rho' > 0$, such that for all $e \in E(H_0)$, $f_e \leq \rho'$. Hence, the total flow through G (and through H_0) is $r \leq \rho' n_+$.

At the conceptual level, the proof proceeds as explained above. We start with $S_0 = \{u\}$. By hypothesis, the flow entering u is at most ρ' . By the expansion property of good networks and by Chernoff bounds,⁵ with high probability, there are at least $P_b(\epsilon_1)np/4$ edges e adjacent to u with $b_e \leq \epsilon_1$. At most half of these edges have flow greater than $\frac{8\rho'}{P_b(\epsilon_1)np}$, thus there are at least $P_b(\epsilon_1)np/8$ edges adjacent to u with latency, wrt g , less than $\frac{8A_{\max}\rho'}{P_b(\epsilon_1)np} + \epsilon_1$. We now let $d_1 = B_s + \frac{8A_{\max}\rho'}{P_b(\epsilon_1)np} + \epsilon_1$ and $S_1 = \{x \in V : d_s(x) \leq d_1\}$. By the discussion above, $|S_1| \geq P_b(\epsilon_1)np/8$.

We now inductively define a sequence of vertex sets S_i and upper bounds d_i on the latency of the vertices in S_i from s , such that $S_i \subseteq S_{i+1}$ and $d_i < d_{i+1}$. This sequence stops the first time that $|\Gamma(S_i)| \geq 3n/5$. We inductively assume that the vertex set S_i and the upper bound d_i on the latency of the vertices in S_i are defined, and that $|\Gamma(S_i)| < 3n/5$. By the expansion property of good networks $|\Gamma(S_i) \setminus S_i| \geq np|S_i|/3$, for sufficiently large n . Thus, with probability at least $1 - e^{-P_b(\epsilon_1/\ln n)np|S_i|/24}$, there are at least $P_b(\frac{\epsilon_1}{\ln n})np|S_i|/6$ vertices outside S_i that are connected to a vertex in S_i by an edge e with $b_e \leq \epsilon_1 / \ln n$. Let S'_i be the set of such vertices, and let E_i be the set of edges that for each vertex $v \in S'_i$, includes a unique edge $e \in E_i$ with $b_e \leq \epsilon_1 / \ln n$ connecting v to a vertex in S_i . Since the flow g may be assumed to be acyclic, a volume

⁵ We repeatedly use the following form of the Chernoff bound (see e.g., [13]): Let X_1, \dots, X_k be random variables independently distributed in $\{0, 1\}$, and let $X = \sum_{i=1}^k X_i$. Then, for all $\epsilon \in (0, 1)$, $\mathbb{P}[X < (1 - \epsilon)\mathbb{E}[X]] \leq e^{-\epsilon^2 \mathbb{E}[X]/2}$, where e is the basis of natural logarithms.

$r \leq \rho'n_+$ of flow is routed through the cut $(S_i, V \setminus S_i)$. Then, at most half of the edges in E_i have flow greater than $2\rho'n_+/|S'_i|$. Consequently, at least half of the vertices $v \in S'_i$ have latency from s :

$$\begin{aligned} d_s(x) &\leq d_i + \frac{\epsilon_1}{\ln n} + A_{\max} \frac{2\rho'n_+}{|S'_i|} \\ &\leq d_i + \frac{\epsilon_1}{\ln n} + \frac{12A_{\max}\rho'n_+}{P_b(\frac{\epsilon_1}{\ln n})np|S_i|} \end{aligned}$$

Thus, we define the next latency upper bound d_{i+1} in the sequence as:

$$d_{i+1} = d_i + \frac{\epsilon_1}{\ln n} + \frac{12A_{\max}\rho'n_+}{P_b(\frac{\epsilon_1}{\ln n})np|S_i|},$$

and we let $S_{i+1} = \{x \in V(G) | d_s(x) \leq d_{i+1}\}$. By the discussion above, and using the inductive definition of S_i 's, we obtain that:

$$\begin{aligned} |S_{i+1}| &\geq \left(\frac{1}{12} P_b(\epsilon_1 / \ln n) np + 1\right) |S_i| \\ &\geq \left(\frac{1}{12} P_b(\epsilon_1 / \ln n) np + 1\right)^i |S_1| \end{aligned}$$

We recall that i^* is the first index i such that $|\Gamma(S_i)| \geq 3n/5$. Then, the inequality above implies that:

$$i^* \leq \frac{\ln(3n/(5|S_1|))}{\ln\left(\frac{1}{12} P_b(\epsilon_1 / \ln n) np + 1\right)} \leq \frac{\ln(24n/(5P_b(\epsilon_1)np))}{\ln\left(\frac{1}{12} P_b(\epsilon_1 / \ln n) np + 1\right)}$$

Using that $pn \geq \ln n$ and that $P_b(\epsilon_1 / \ln n) np = \omega(1)$, the inequality above implies that $i^* \leq \ln n$, for sufficiently large n .

Therefore, we obtain an upper bound on the latency from s of any vertex in S_{i^*} :

$$\begin{aligned} d_{i^*} &\leq d_0 + i^* \frac{\epsilon_1}{\ln n} + \sum_{i=1}^{i^*} \frac{12A_{\max}\rho'n_+}{P_b(\frac{\epsilon_1}{\ln n})np|S_i|} \\ &\leq d_1 + \frac{\epsilon_1}{\ln n} \ln n + \sum_{i=1}^{\ln n} \frac{12A_{\max}\rho'n_+}{P_b(\frac{\epsilon_1}{\ln n})np \left(\frac{1}{12} P_b(\frac{\epsilon_1}{\ln n}) np + 1\right)^i |S_1|} \\ &= d_1 + \epsilon_1 + \frac{12A_{\max}\rho'n_+}{P_b(\frac{\epsilon_1}{\ln n})np|S_1|} \sum_{i=1}^{\ln n} \left(\frac{1}{12} P_b(\frac{\epsilon_1}{\ln n}) np + 1\right)^{-i} \\ &\leq \left(B_s + \frac{8A_{\max}\rho'}{P_b(\epsilon_1)np} + \epsilon_1\right) + \epsilon_1 + \frac{96A_{\max}\rho'n_+}{P_b\left(\frac{\epsilon_1}{\ln n}\right) P_b(\epsilon_1)(np)^2} \sum_{i=1}^{\infty} 2^{-i} \\ &\leq B_s + 2\epsilon_1 + \frac{8A_{\max}\rho'}{P_b(\epsilon_1)np} + \frac{144A_{\max}\rho'k}{P_b\left(\frac{\epsilon_1}{\ln n}\right) P_b(\epsilon_1)np} \end{aligned}$$

For the penultimate inequality, we use that $P_b(\epsilon_1/\ln n)np = \omega(1)$, which implies that $1 + P_b(\epsilon_1/\ln n)np/12 \geq 2$, for n sufficiently large. For the last inequality, we use that $n_+ \leq 3knp/2$, for some constant $k > 0$, by hypothesis.

Moreover, we observe that probability that the above construction fails is at most:

$$\begin{aligned} \sum_{i=1}^{i^*} e^{-P_b(\epsilon_1/\ln n)np|S_i|/24} &\leq \sum_{i=1}^{i^*} e^{-\left(\frac{1}{12}P_b(\epsilon_1/\ln n)np+1\right)^i |S_i|/24} \\ &\leq \ln n e^{-\left(\frac{1}{12}P_b(\epsilon_1/\ln n)np+1\right)P_b(\epsilon_1)np/192} \end{aligned}$$

Therefore, the construction above succeeds with high probability.

Similarly, we start from $T_0 = \{v\}$, and inductively define a sequence of vertex sets $T_0 \subseteq T_1 \subseteq \dots \subseteq T_{j^*}$, and a sequence of upper bounds $d'_0 < d'_1 < \dots < d'_{j^*}$ on the latency from t of the vertices in each T_j . We let $T_j = \{x \in V(G) | d_t(x) \leq d'_j\}$. The sequence stops as soon as $|\Gamma(T_j)| \geq 3n/5$ for the first time. Namely, j^* is the first index with $|\Gamma(T_{j^*})| \geq 3n/5$. Using exactly the same arguments, we can show that with high probability, we have that $j^* \leq \ln n$, and that:

$$d'_{j^*} \leq B_t + 2\epsilon_1 + \frac{8A_{\max}\rho'}{P_b(\epsilon_1)np} + \frac{144A_{\max}\rho'k}{P_b\left(\frac{\epsilon_1}{\ln n}\right)P_b(\epsilon_1)np}$$

Wlog., we assume that $S_{i^*} \cap T_{j^*} = \emptyset$. Since $|\Gamma(S_{i^*})| + |\Gamma(T_{j^*})| \geq 6n/5$, there are at least $n/10$ edge disjoint paths of length at most 2 between S_{i^*} and T_{j^*} . Furthermore, by Chernoff bounds, with high probability, there are at least $P_b(\epsilon_1)^2n/12$ such paths with both edges e on the path having $b_e \leq \epsilon_1$. At most half of these paths have flow more than $2\frac{12\rho'n_+}{P_b(\epsilon_1)^2n}$ and thus there is a path from a vertex of S_{i^*} to a vertex of T_{j^*} that costs at most $2\epsilon_1 + 2A_{\max}\frac{24\rho'n_+}{P_b(\epsilon_1)^2n}$.

Putting everything together, we have that there is a path p' that starts from s , moves to u , goes through vertices of the sequence S_1, \dots, S_{i^*} , proceeds to a vertex of $\Gamma(S_{i^*}) \cap \Gamma(T_{j^*})$, and from there, continues through vertices of the sequence T_{j^*}, \dots, T_1 , until finally reaches v , and then t . The latency of this path is:

$$\ell_{p'}(g) \leq B_s + B_t + 6\epsilon_1 + 2\left(\frac{8A_{\max}\rho'}{P_b(\epsilon_1)np} + \frac{48A_{\max}\rho'k}{P_b\left(\frac{\epsilon_1}{\ln n}\right)P_b(\epsilon_1)np}\right) + \frac{48A_{\max}\rho'n_+}{P_b(\epsilon_1)^2n}$$

We recall that since the flow g is a minimizer of $\text{Pot}(g)$, for any g -used path $q = (s, u, \dots, v, t)$, $\ell_q(g) \leq \ell_{p'}(g)$. Thus we obtain that any g -used path $q = (s, u, \dots, v, t)$ has latency

$$\begin{aligned} \ell_q(g) &\leq B_s + B_t + 6\epsilon_1 + 2\left(\frac{8A_{\max}\rho'}{P_b(\epsilon_1)np} + \frac{48A_{\max}\rho'k}{P_b\left(\frac{\epsilon_1}{\ln n}\right)P_b(\epsilon_1)np}\right) + \frac{48A_{\max}\rho'n_+}{P_b(\epsilon_1)^2n} \\ &= B_s + B_t + 6\epsilon_1 + \rho' \frac{8A_{\max}}{P_b(\epsilon_1)} \left[\frac{2}{np} \left(1 + \frac{6k}{P_b\left(\frac{\epsilon_1}{\ln n}\right)}\right) + \frac{9kp}{2P_b(\epsilon_1)} \right] \end{aligned}$$

$$\begin{aligned}
 &= B_s + B_t + 6\epsilon_1 + \rho' \times \Theta(1) \left[O\left(\frac{1}{np}\right) + O\left(\frac{1}{npP_b\left(\frac{\epsilon_1}{\ln n}\right)}\right) + O(1)p \right] \\
 &= B_s + B_t + 6\epsilon_1 + \rho' \times o(1) \tag{10}
 \end{aligned}$$

In the above asymptotic expressions we used the assumption of Lemma 5 that $n_+/n = \max\{n_s, n_t\}/n \leq 3kp/2$, for constant $k > 0$. We also used in the Definition 3 of good networks the condition (3) that $P_b(\epsilon_1/\ln n)np = \omega(1)$. We also used in the Definition 1 of reasonable latency coefficients the condition (3) that for any constant $\epsilon_1 > 0$ there exists a constant $\delta_{\epsilon_1} > 0$ such that $P_b(\epsilon_1) \geq \delta_{\epsilon_1}$. Finally, we used in the Definition 3 of good networks that $pn = o(n) \Rightarrow p = o(1)$ and $pn = \Omega(\log n)$. \square

We conclude that the flow g satisfies property (5) of Lemma 5 by Proposition 1 proved above in (10). \square

Corollary 1 *If $\rho' = O(1)$ in (10) then $\forall \epsilon_1 > 0 \Rightarrow \ell_q(g) \leq B_s + B_t + 7\epsilon_1$, for each g -used path in (10). Thus, g is an $7\epsilon_1$ -NE, $\forall \epsilon_1 > 0$.*

6.2 Approximating the Best Subnetwork When $r \leq \frac{B_{\max} n_+}{A_{\min} \epsilon}$

Grouping the Neighbors of s and t Let us now consider the entire random instance G and the best subnetwork H_0^* of G_0 (recall, G_0 is the 0-latency simplification of G , see Definition 2) and f^* the corresponding NE flow on H_0^* . A subtle issue of Lemma 5 is that it applies only to subsets of s -links in $E_s(H_0^*)$ and t -links in $E_t(H_0^*)$ that have (almost) the same s, t -link latency (recall the role of B_s, B_t in Lemma 5) under the given NE flow f^* in H_0^* . But, unlike G_0 , now H_0^* need *not* be internally complete bipartite, therefore may exist neighbors of s (resp. t) connected to disjoint subsets of V_t (resp. of V_s) in H_0^* , and thus have quite different s, t -link latency. Hence, to apply Lemma 5, we partition the neighbors of s and the neighbors of t into an at most a *constant* number of classes V_s^i and V_t^j according to their latency. For convenience, we let $\epsilon_2 = \epsilon/6$ and apply Theorem 2. By Theorem 2 there is an acceptable flow f , determined by exhaustive search within the time bound of this theorem, and a network H_0 containing only the f -used paths, which is an ϵ_2 -Nash flow on H_0 and induces a path latency $L \equiv L_{H_0}(f)$. By Theorem 2, applied with error $\epsilon_2 = \epsilon/6$, there exists a constant ϱ such that $0 < \varrho \leq \frac{24A_{\max}B_{\max}}{\delta_\epsilon A_{\min}^2 \epsilon}$ (where this bound on ϱ holds by Lemma 4), where for all $e \in E(H_0)$, $0 < f_e \leq \varrho + \epsilon_2$. The highest latency wrt f of an s -link, plus the highest latency of an t -link upper bounds the latency of any s - t path used by f in H_0 . Thus, using the error $\epsilon_2 = \epsilon/6$ above, we can define in (11) the following constants κ, \mathcal{L}, k that will be useful in the sequel:

$$\begin{aligned}
 L \equiv L_{H_0}(f) &\leq 2A_{\max}(\varrho + \epsilon_2) + 2B_{\max} \leq 2A_{\max} \left(\frac{24A_{\max}B_{\max}}{\delta_\epsilon A_{\min}^2 \epsilon} + \epsilon/6 \right) \\
 &+ 2B_{\max} = \mathcal{L}
 \end{aligned}$$

$$\begin{aligned} \kappa &= \lceil \mathcal{L}/\epsilon_2 \rceil = \lceil 6\mathcal{L}/\epsilon \rceil \\ 0 \leq k \leq \kappa^2 &= \left\lceil 12 \frac{A_{\max} \left(\frac{24A_{\max}B_{\max}}{\delta_\epsilon A_{\min}^2 \epsilon} + \epsilon/6 \right) + B_{\max}}{\epsilon} \right\rceil^2 \end{aligned} \tag{11}$$

Note that *constant* \mathcal{L} in (11) above is independent of the particular value that r takes and independent of f^* , f . Recall (11) and partition the interval $[0, \mathcal{L}]$ of the possible s, t -link latency values into the *constant* number $\kappa = \lceil \mathcal{L}/\epsilon_2 \rceil = \lceil 6\mathcal{L}/\epsilon \rceil$ of subintervals, where the i -th subinterval is $I^i = (i\epsilon_2, (i + 1)\epsilon_2], i = 0, \dots, \kappa - 1$. Once more κ is independent of the particular value that r takes and independent of f^* , f . We partition the vertices of V_s (resp. of V_t) that receive positive flow by f into the same constant number κ of 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 the same constant number κ of 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 5, 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 $k \leq \kappa^2$ in (11) to 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 bounded above by *constant* κ^2 that does not depend on $|V|, r, f, f^*$. In particular, the flow f at hand determines k which, by virtue of Theorem 2, the f at hand is the best outcome flow via *deterministic* exhaustive search over all acceptable flows. 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 $V_t^{(i,j)}$. We conclude that, when f is handed, the above construction of all the corresponding subsets takes time bounded by κ^2 in (11).

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) of Definition 3 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 $G^{(i,j)} \equiv G[\{s, t\} \cup V_s^{(i,j)} \cup V_t^{(i,j)} \cup V_t^{(i,j)}]$, which is an $(n/k, p, 1)$ -good network, since G is an (n, p, k) -good network. Therefore, we can apply Lemma 5 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 since $\rho' = \varrho + \epsilon_2 = O(1)$ we can apply, in particular, Corollary 1. 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 5. 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 5, 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 $\epsilon_1 = \epsilon/42$ and $\epsilon_2 = \epsilon/6$, the flow g is an ϵ -Nash flow of (H, r) , and satisfies $L_H(g) \leq L_{H_0}(f) + \epsilon/2$. Thus, we obtain:

Lemma 6 Fix any $\epsilon > 0$ and set constant $k = \kappa^2$ as in (11). Let (G, r) with G a (n, p, k) -good network (Definition 3) and arbitrary $r : 0 < r \leq \frac{B_{\max} n_{\pm}}{A_{\min} \epsilon}$. Consider by Theorem 2 an acceptable $(\epsilon/6)$ -Nash flow f and the subnetwork $H_0 \subseteq G_0$ used by f which satisfies: $\forall e \in E(H_0) \Rightarrow 0 < f_e \leq \varrho + \epsilon/6$, with constant $0 < \varrho \leq \frac{24 A_{\max} B_{\max}}{\delta_{\epsilon} A_{\min}^2 \epsilon}$.

Then whp⁶ we can compute in $\text{poly}(|V|)$ time a subnetwork $H \subseteq G$ and an ϵ -Nash flow g of (H, r) with $L_H(g) \leq L_{H_0}(f) + \epsilon/2$.

Proof We consider the subnetwork H (with the original latency functions of G), computed as the union of subnetworks $H^{(i,j)}$, and the flow g , computed as the union of the flows $g^{(i,j)}$, 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. We recall that by construction, all edges of H are used by g . We show that if $\epsilon_1 = \epsilon/42$ and $\epsilon_2 = \epsilon/6$, the flow g is an ϵ -Nash flow of (H, r) , and satisfies $L_H(g) \leq L_{H_0}(f) + \epsilon/2$. We stress that the edge and path latencies here are calculated with respect to the original latency functions of G and under the edge congestion induced by the flow g (or the flow f).

For convenience, we let $B^{(i,j)} = B_s^{(i,j)} + B_t^{(i,j)}$ for any pair of non-empty classes $V_s^{(i,j)}$ and $V_t^{(i,j)}$. Since the difference in the latency of any edges in the same group is at most ϵ_2 , we obtain that for any edge $e \in E_s^{(i,j)}$, $B_s^{(i,j)} - \epsilon_2 \leq \ell_e(f_e) \leq B_s^{(i,j)}$, and similarly, that for any edge $e \in E_t^{(i,j)}$, $B_t^{(i,j)} - \epsilon_2 \leq \ell_e(f_e) \leq B_t^{(i,j)}$. Therefore, since H_0 is a 0-latency simplified network, and since by hypothesis, all the edges of H_0 are used by f , for any pair of non-empty classes $V_s^{(i,j)}$ and $V_t^{(i,j)}$, and for any $s - t$ path p going through a vertex of $V_s^{(i,j)}$ and a vertex of $V_t^{(i,j)}$,

$$B^{(i,j)} - 2\epsilon_2 \leq \ell_p(f) \leq B^{(i,j)}$$

⁶ wrt. the random choice of the latency functions of G .

Moreover, since f is an ϵ_2 -Nash flow of (H_0, r) , for any $s - t$ path $p \in \mathcal{P}_{H_0}$,

$$L_{H_0}(f) - \epsilon_2 \leq \ell_p(f) \leq L_{H_0}(f)$$

Combining the two inequalities above, we obtain that for any pair of non-empty classes $V_s^{(i,j)}$ and $V_t^{(i,j)}$,

$$B^{(i,j)} - 2\epsilon_2 \leq L_{H_0}(f) \leq B^{(i,j)} + \epsilon_2 \tag{12}$$

As for the flow g , by construction, we have that $g_e = f_e$ for all edges $e \in E_s \cup E_t$. Therefore, for any edge $e \in E_s^{(i,j)}$, $B_s^{(i,j)} - \epsilon_2 \leq \ell_e(g_e) \leq B_s^{(i,j)}$, and similarly, for any edge $e \in E_t^{(i,j)}$, $B_t^{(i,j)} - \epsilon_2 \leq \ell_e(g_e) \leq B_t^{(i,j)}$. Thus, by Lemma 5 and its Corollary 1, and since all the edges of any subnetwork $H^{(i,j)}$ are used by g , for any $s - t$ path p in the subnetwork $H^{(i,j)}$, $B^{(i,j)} - 2\epsilon_2 \leq \ell_p(g) \leq B^{(i,j)} + 7\epsilon_1$. Using (12), we obtain that for any subnetwork $H^{(i,j)}$ and any $s - t$ path p of $H^{(i,j)}$,

$$L_{H_0}(f) - 3\epsilon_2 \leq \ell_p(g) \leq L_{H_0}(f) + 2\epsilon_2 + 7\epsilon_1 \tag{13}$$

Furthermore, we recall that the subnetworks $H^{(i,j)}$ only have in common the vertices s and t , and possibly some vertices of $V_s \cup V_t$ and some edges of $E_s \cup E_t$. They have neither any other vertices in common, nor any edges connecting vertices in the intermediate parts of different subnetworks $H^{(i,j)}$ and $H^{(i',j')}$. Hence, any $s - t$ path p of H passes through a single subnetwork $H^{(i,j)}$. Therefore, and since by construction, all the edges and the paths of H are used by g , (13) holds for any $s - t$ path p of H .

Thus, we have shown that g is a $(5\epsilon_2 + 7\epsilon_1)$ -Nash flow of (H, r) , and that $L_H(g) \leq L_{H_0}(f) + 2\epsilon_2 + 7\epsilon_1$. Using $\epsilon_2 = \epsilon/6$ and $\epsilon_1 = \epsilon/42$, we obtain the performance guarantees of g as stated in Lemma 6. \square

6.3 Approximating the Best Subnetwork When $r > \frac{B_{\max}n+}{A_{\min}\epsilon}$

In Lemma 2 we have proved that: for an arbitrary net G at hand, if G_0 denotes the 0-latency simplified network (Definition 2) of G the Nash flow f can be computed in strongly polynomial time (see e.g., [37]) on (G_0, r) and 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 $r > \frac{B_{\max}n+}{A_{\min}}$ then $L_1, L_2 > B_{\max}$ and all edges in $E_s \cup E_t$ are used by f . In particular, recall here (1) that the common path latency of the NE of G_0 is

$$L(G_0, r) = L_1 + L_2 \leq \left(\frac{r}{A_s} + \frac{r}{A_t} \right) + 2B_{\max} \tag{14}$$

Also, in Lemma 2 we have proved that the common edge latency L_1 of loaded s -links is $L_1 \leq \frac{r}{A_s} + B_{\max}$. From this, we derive that the upper bound ρ' used in the property (5) of Lemma 5 on the load of any s -link e , satisfies:

$$A_{\min}\rho' = L_1 \leq \frac{r}{A_s} + B_{\max} \Rightarrow \rho' \leq \left(\frac{r}{A_s} + B_{\max} \right) \frac{1}{A_{\min}} \quad (15)$$

which bounds the flow traversed by any loaded path departing from a $u \in \Gamma(G_0)$ and arriving at a $u \in \Gamma(G_0)$. By the NE flow f on (G_0, r) all the loaded s -links have equal latency L_1 and all the loaded t -links have equal latency L_2 , therefore we can plug f into Lemma 5 and compute in $poly|V|$ time a flow g and a subnetwork $G' \subseteq G$ that satisfies properties (1-5) of Lemma 5. In particular, g is a NE on the subnetwork $G' = G_g$ that contains all g -used paths, where each g -used path has latency as in property (5) of Lemma 5, that is, for any constant $\epsilon_1 > 0$:

$$\begin{aligned} L(G', r) &\leq L_1 + L_2 + 6\epsilon_1 + \rho' \times o(1) \leq \left(\frac{r}{A_s} + \frac{r}{A_t} \right) + 2B_{\max} + 6\epsilon_1 + \rho' \times o(1) \\ &\leq \left(\frac{r}{A_s} + \frac{r}{A_t} \right) + 2B_{\max} + 6\epsilon_1 + \left(\frac{r}{A_s} + B_{\max} \right) \frac{1}{A_{\min}} \times o(1) \end{aligned} \quad (16)$$

where above we have plugged the ρ' in (15). The expression in (16) will be useful below, when divided by the common path latency $L(H^*, r)$ of the best subnetwork $H^* \subseteq G$ and derive the approximation ratio of our algorithm 1. Towards this, in Lemma 2 we have proved that the best subnetwork $H_0^* \subseteq G$ of the 0-latency simplification of our initial graph G has a common path latency $L(H_0^*, r)$ at least:

$$L(H^*, r) \geq L(H_0^*, r) \geq \left(\frac{r}{A_s} + \frac{r}{A_t} \right) \gg 1 \quad (17)$$

In particular, by virtue of our Lemma 1, the expression in (17) is a lower bound of the common path latency $L(H^*, r)$ of the best subnetwork of the original G . We should stress here that we assume that this expression is $\gg 1$, since we refer to very high values of total rate r that loads all s, t -links thus inducing high common path latency. Otherwise, if the common path latency is $O(1)$, our previous analysis in Sect. 6.2 applies as is. We conclude that, if we consider the ratio of (16) and (17), then for any positive constant $\epsilon_1 > 0$, our approximation ratio for computing an ϵ -NE is at most:

$$\begin{aligned} \frac{L(G', r)}{L(H^*, r)} &\leq \frac{\left(\frac{r}{A_s} + \frac{r}{A_t} \right) + 2B_{\max} + 6\epsilon_1 + \left(\frac{r}{A_s} + B_{\max} \right) \frac{1}{A_{\min}} \times o(1)}{\left(\frac{r}{A_s} + \frac{r}{A_t} \right)} \\ &\leq 1 + \epsilon + \frac{6\epsilon_1}{\left(\frac{r}{A_s} + \frac{r}{A_t} \right)} + \frac{\left(\frac{r}{A_s} + B_{\max} \right)}{\left(\frac{r}{A_s} + \frac{r}{A_t} \right) A_{\min}} \times o(1) \\ &\leq 1 + \epsilon + o(\epsilon_1) + \frac{\left(\frac{r}{A_s} + \frac{r}{A_t} \right)}{\left(\frac{r}{A_s} + \frac{r}{A_t} \right) A_{\min}} \times o(1) + \frac{B_{\max}}{\left(\frac{r}{A_s} + \frac{r}{A_t} \right) A_{\min}} \times o(1) \end{aligned}$$

$$\begin{aligned}
 &= 1 + \varepsilon + o(\varepsilon_1) + \frac{o(1)}{A_{\min}} + \frac{\varepsilon}{2A_{\min}} \times o(1) \\
 &= 1 + \varepsilon + o(\varepsilon_1) + o(1) \Rightarrow \\
 L(G', r) &\leq (1 + \varepsilon + o(\varepsilon_1) + o(1))L(H^*, r) = (1 + \varepsilon + o(1))L(H^*, r) \tag{18}
 \end{aligned}$$

To derive (18), we have used the following bounds. From the 1st line to 2nd line we used that: $\frac{2B_{\max}}{\left(\frac{r}{A_s} + \frac{r}{A_t}\right)} \leq \varepsilon$, proved in (5). From the 2nd line to 3rd line we used that: $\frac{r}{A_s} < \left(\frac{r}{A_s} + \frac{r}{A_t}\right)$ and that $\left(\frac{r}{A_s} + \frac{r}{A_t}\right) \gg 1$ that we assumed in (17) because we study the case of high values of r . From the 3rd line to 4th line, we recall the bound used in 1st to 2nd line, to obtain: $\frac{B_{\max}}{\left(\frac{r}{A_s} + \frac{r}{A_t}\right)} \leq \varepsilon/2$. From the 4th line to 5th line we used that: $\frac{1}{A_{\min}} = \Omega(1)$, see assumption (1) of the reasonable coefficients in Definition 1.

Remark 2 We can also relax the assumption that $\left(\frac{r}{A_s} + \frac{r}{A_t}\right) \gg 1$ and obtain a slightly weaker approximation ratio than in (18). In particular, recall $\frac{2B_{\max}}{\left(\frac{r}{A_s} + \frac{r}{A_t}\right)} \leq \varepsilon$, proved in (5) that yields in the 2nd line of (18) that term $\frac{6\varepsilon_1}{\left(\frac{r}{A_s} + \frac{r}{A_t}\right)} \leq \frac{3\varepsilon_1\varepsilon}{B_{\max}} < \varepsilon, \forall \varepsilon_1 < \frac{B_{\max}}{3}$. Thus, in a scenario that $\left(\frac{r}{A_s} + \frac{r}{A_t}\right) = \Omega(1)$ then $\forall \varepsilon_1 : 0 < \varepsilon_1 < \frac{B_{\max}}{3}$ we get $L(G', r) \leq (1 + 2\varepsilon + o(1))L(H^*, r)$.

7 Open Problem

Once more we highlight that the reduction itself runs in polynomial time. The time consuming step is the application of [10, 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 [33]. The (improved) approximability of the best subnetwork for 0-latency simplified networks is an intriguing open problem arising from our work.

References

1. Althöfer, I.: On sparse approximations to randomized strategies and convex combinations. *Linear Algebra Appl.* **99**, 339–355 (1994)
2. Bollobás, B.: *Random Graphs*, 2nd Edition Cambridge Studies in Advanced Mathematics, vol. 3. Cambridge University Press, Cambridge (2001)
3. Braess, D.: Über ein paradox aus der Verkehrsplanung. *Unternehmensforschung* **12**, 258–268 (1968)
4. Cheeseman, P., Kanefsky, B., Taylor, W.: Where the really hard problems are. *IJCAI* **1**, 331–337 (1991)
5. Chung, F., Young, S.J.: Braess’s paradox in large sparse graphs. In: *Proceedings of the 6th Workshop on Internet and Network Economics (WINE ’10)*, LNCS, vol. 6484, pp. 194–208 (2010)
6. Chung, F., Young, S.J., Zhao, W.: Braess’s paradox in expanders. *Random Struct. Algorithms* **41**(4), 451–468 (2012)
7. Cole, R., Dodis, Y., Roughgarden, T.: How much can taxes help selfish routing? *J. Comput. Syst. Sci.* **72**(3), 444–467 (2006)
8. Duffin, R.J.: Topology of series-parallel networks. *J. Math. Anal. Appl.* **10**, 303–318 (1965)

9. Erdos, P., Renyi, A.: On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci.* **5**, 17–61 (1960)
10. Fotakis, D., Kaporis, A.C., Spirakis, P.G.: Efficient methods for selfish network design. *Theor. Comput. Sci.* **448**, 9–20 (2012)
11. Franco, J.: Results related to threshold phenomena research in satisfiability: lower bounds. *Theor. Comput. Sci.* **265**, 147–157 (2001)
12. Friedman, E.: Genericity and congestion control in selfish routing. In: 43rd IEEE Conference on Decision and Control, pp. 4667–4672 (2004)
13. Hagerup, T., Rüb, C.: A guided tour of Chernoff bounds. *Inf. Process. Lett.* **33**(6), 305–308 (1990)
14. Johnson, D.: The NP-completeness column: an ongoing guide. *J. Algorithms* **5**(2), 284–299 (1984)
15. Johnson, D., Aragon, C., McGeoch, L., Shevon, C.: Optimization by simulated annealing: an experimental evaluation. *Oper. Res.* **37**(6), 865–892 (1989)
16. Karp, R.: Combinatorics, complexity, and randomness (turing award lecture). *Commun. ACM* **29**(2), 98–109 (1989)
17. Kelly, F.: The Princeton companion to mathematics. In: Gowers, T., Green, J., Leader, I. (eds.) *The Mathematics of Traffic in Networks*. Princeton University Press, Princeton (2008)
18. Kirkpatrick, S., Selman, B.: Critical behavior in the satisfiability of random Boolean expressions. *Science* **264**, 1297–1301 (1994)
19. Lin, H.C., Roughgarden, T., Tardos, É., Walkover, A.: Stronger bounds on Braess’s paradox and the maximum latency of selfish routing. *SIAM J. Discrete Math.* **25**(4), 1667–1686 (2011)
20. Lipton, R.J., Markakis, E., Mehta, A.: Playing large games using simple strategies. In: Proceedings of the 4th ACM Conference on Electronic Commerce (EC ’03), pp. 36–41 (2003)
21. Lipton, R.J., Young, N.E.: Simple strategies for large zero-sum games with applications to complexity theory. In: Proceedings of the 26th ACM symposium on theory of computing (STOC ’94), pp. 734–740 (1994)
22. Mézard, M., Parisi, G., Zecchina, R.: Analytic and algorithmic solution of random satisfiability problems. *Science* **297**, 812–815 (2002)
23. Mitchell, D., Selman, B., Levesque, H.: Hard and easy distribution of SAT problems. In: Proceedings of 10th National Conference on Artificial Intelligence (AAAI ’92), pp. 459–465
24. Milchtaich, I.: Network topology and the efficiency of equilibrium. *Games Econ. Behav.* **57**, 321–346 (2006)
25. Morris, P.: The breakthrough method for escaping local minima. *AAAI* **6**, 40–45 (1993)
26. Murchland, J.D.: Braess’s paradox of traffic flow. *Transp. Res.* **4**, 391–394 (1970)
27. Nagurney, A., Boyce, D.: Preface to “on a paradox of traffic planning”. *Transp. Sci.* **39**(4), 443–445 (2005)
28. Pas, E.I., Principio, S.L.: Braess’s paradox: some new insights. *Transp. Res. Part B* **31**(3), 265–276 (1997)
29. Patriksson, M.: *The Traffic Assignment Problem—Models and Methods*. Linköping Institute of Technology, Linköping (1991)
30. Roughgarden, T.: *Selfish Routing*. Ph.D dissertation, Cornell Univ., USA, May (2002). <http://theory.stanford.edu/~tim/>
31. Roughgarden, T.: Stackelberg scheduling strategies. *SIAM J. Comput.* **33**(2), 332–350 (2004)
32. Roughgarden, T.: *Selfish Routing and the Price of Anarchy*. MIT press, Cambridge, MA (2005)
33. Roughgarden, T.: On the severity of braess’s paradox: designing networks for selfish users is hard. *J. Comput. Syst. Sci.* **72**(5), 922–953 (2006)
34. Selman, B., Kautz, H., Cohen, B.: Local search strategies for satisfiability testing. *DIMACS* **26**, 521–532 (1993)
35. Steinberg, R., Zangwill, W.I.: The prevalence of Braess’ paradox. *Transp. Sci.* **17**(3), 301–318 (1983)
36. Valiant, G., Roughgarden, T.: Braess’s paradox in large random graphs. *Random Struct. Algorithms* **37**(4), 495–515 (2010)
37. Végh, L.A.: Strongly polynomial algorithm for a class of minimum-cost flow problems with separable convex objectives. In: Proceedings of the 44th ACM Symposium on Theory of Computing (STOC ’12), pp. 27–40 (2012)
38. International Competition and Symposium on Satisfiability Testing, Beijing ’96, Beijing, China, March 15–17 (1996)
39. 2nd Dimacs Implementation Challenge. In: Johnson, D., Trick, M. (eds.) *Dimacs Series in Discrete Mathematics and TCS*, **26**(4), AMS (1996). <http://dimacs.rutgers.edu/Volumes/Vol26.html>