A Refined Performance Characterization of Longest-Queue-First Policy in Wireless Networks

Bo Li, Cem Boyaci and Ye Xia
Computer and Information Science and Engineering Department
University of Florida
Email: {boli, cboyaci, yx1}@cise.ufl.edu

***Abstract*—One of the major challenges in wireless networking is how to optimize the link scheduling decisions under interference constraints. Recently, a few algorithms have been introduced to address the problem. However, solving the problem to optimality for general wireless interference models is known to be NP-hard. The research community is currently focusing on finding simpler, sub-optimal scheduling algorithms and on characterizing the algorithm performance. In this paper, we address the performance of a specific scheduling policy called Longest Queue First (LQF), which has gained significant recognition lately due to its simplicity and high efficiency in empirical studies. There has been a sequence of studies characterizing the guaranteed performance of the LQF schedule, culminating at the construction of the σ-local pooling concept by Joo et al. [1]. In this paper, we refine the notion of σ-local pooling and use the refinement to capture a larger region of guaranteed performance.**

***Index Terms*—Wireless Networks Scheduling, Longest Queue First Policy, Stability, Local Pooling, Interference**

I. Introduction

Recent years have seen a great development in wireless networking technologies and their usage. One of the major challenges in wireless networking is how to utilize the communication medium efficiently under interference constraints. For different wireless technologies, different interference models have been established. The 1-hop interference model (also known as the node-exclusive or primary interference model), where two links interfere only if they share a common node, is suitable to characterize the interference in FH-CDMA and Bluetooth networks [2]. The hop count is measured in the interference graph in which a node represents a physical link in the network and an edge between two nodes means the two corresponding physical links interfere with each other. With the same interference-graph-based terminology, the 2-hop interference model can successfully capture the interference relationship in the IEEE 802.11 network [3]. In [4] and [5], the authors considered the more general k-hop interference model.

The research on joint link scheduling and routing strives to find the most efficient way to forward traffic from sources to destinations. In [6], the authors provided an algorithm that achieves the full capacity region of the wireless network. However, their algorithm requires solving a global maximization subproblem at each iteration. For the 1-hop interference model, this subproblem reduces to finding the maximum weighted matching in the backlog weighted network graph. While maximum matching can be solved in $O(|V|^3)$ time with a centralized algorithm [1], where V is the set of nodes in the network, the running time is considered inefficient for a network algorithm and a faster approach is desired. The same algorithm given in [6] can also be applied to more generic interference models; but the subproblem becomes intractable in those cases. For instance, under general interference models characterized by some interference graph, the subproblem is to find the maximum weighted independent set of the interference graph, which is NP-hard and makes the algorithm in [6] inapplicable. In [5], the authors showed that for the k-hop interference model where $k > 1$, the subproblem is also NP-Hard.

Given the above difficulty, one of the main efforts by the research community is to find simpler sub-optimal scheduling algorithms that are also friendly to distributed implementation, and to characterize their performance guarantee. Among the proposed solutions, the *Longest Queue First* (LQF) algorithm has distinguished itself due to its simplicity and high performance in empirical studies [7]. In an effort to understand the surprising efficiency of this simple algorithm, Dimakis and Walrand have identified a sufficient condition for the algorithm to achieve the entire capacity region of the network for single-hop traffic [8]. In particular, they have shown that if the network topology and interference structure satisfy a condition known as *local pooling*, then the LQF algorithm achieves the entire capacity region. Brzezinski et al. have extended the definition of local pooling to the multi-hop traffic situation [9]. The same authors have also investigated classes of networks that satisfy single-hop local pooling [10].

In a different direction of generalizing local pooling, Joo et al. have investigated a fractional version called σ-local pooling [1]. Specifically, suppose the network is denoted by G, and the capacity region is denoted by Λ, i.e., the largest rate region that can possibly be supported by the network using some scheduling policy. They defined and studied the properties of the largest number σ, $0 < \sigma \leq 1$, for which the rate region $\sigma\Lambda$ is achievable (stabilizable) by the LQF policy. This largest number is denoted by $\sigma^*(G)$. It provides a way to measure the performance of LQF on an arbitrary network.

In this paper, we extend the definition of σ-local pooling further to better characterize the performance of LQF. We show that the one-parameter characterization by σ-local pooling, although attractive for its parsimony, tends to underestimate the stability region that can be achieved by LQF. This leads to the investigation of multiple-parameter characterizations. We start by defining σ-local pooling *for a link*, denoted by σ_l^* for link l. We then construct a diagonal matrix $\Sigma^*(G) = \text{diag}(\sigma_l^*)_{l \in E}$, where E is the set of links. We then show that the linearly transformed region $\Sigma^*(G)\Lambda$ is achievable by LQF.

The relationship between the newly-defined link σ-local pooling and the original σ-local pooling (for the whole network) in [1] is intriguing. We show that $\sigma^*(G) = \min_{l \in E} \sigma_l^*$. In other words, the guaranteed stability region in [1] is derived by linearly transforming the capacity region (with the $\sigma^*(G)I$ matrix, where I is the identity matrix) using the smallest diagonal entry in $\Sigma^*(G)$. As a result, using $\sigma^*(G)$ can lead to severe underestimate of the stability region of LQF. Hence, our new local pooling concept leads to a more accurate performance characterization for LQF.

Throughout the paper, we show that our multiple-parameter refinement of σ-local pooling is at an appropriate level of generality and structural richness, the study of which can provide tools for deeper understanding and operationalization of the local pooling concept. We argue that link σ-local pooling and the associated *limiting set* are fundamental concepts. We also define *set* σ-local pooling (for a set of links), which can be computed by linear programming, and show how it is related to *link* σ-local pooling. The duality theory of linear programming provides means for bounding or estimating the values of various σ-local pooling concepts. We provide an algorithm for estimating the local-pooling factors of links.

The rest of the paper is organized as follows. In Section II, we provide our network model, basic definitions and notations, and describe the link scheduling problem. In Section III, we describe the main conclusion for performance characterization of LQF using the new notion of link σ-local pooling. In Section IV, we develop a fuller theory of link and set σ-local pooling that helps to apply these new concepts. In Section V, we provide methods to estimate or bound the link and set σ-local pooling factors. In Section VI, we provide additional theoretical results about σ-local pooling. In Section VII, we give additional related work. Section VIII concludes the paper.

II. Preliminaries

In our model, a wireless network is represented by a directed graph $G = (V, E, \mathcal{I})$, where V is the set of nodes, E is the set of links and $\mathcal{I}$ represents the interference relation among the links. In this work, we inherit the *protocol model* for interference [11] in which two links cannot be activated simultaneously if they interfere with each other.[1] The interference relation for the protocol model can be represented by a symmetric 0-1 matrix of size $|E| \times |E|$ where a value 1 in an entry indicates the existence of pairwise interference between the two corresponding links and 0 indicates the absence of such interference. Equivalently, the interference relation can be represented by the *interference graph* (also known as *conflict graph*) in which a node represents a physical link and an edge represents the existence of interference between two physical links (which are two nodes in the interference graph). The interference graph is denoted by G^I, unless specified otherwise.

We represent a schedule by a $|E|$-dimensional 0-1 vector, where a value 1 in an entry indicates the link is active and 0 indicates otherwise. A feasible schedule corresponds to a set of active links that is free from interfering pairs. A feasible schedule is said to be *maximal* if no more links can be activated without violating the interference constraint. Note that a feasible schedule corresponds to an *independent set* in the interference graph and a maximal schedule corresponds to a maximal independent set.[2]

Let M_E be the matrix whose columns are all the maximal schedules. Occasionally, we also view M_E as the set of all maximal schedules. Let $Co(M_E)$ denote the convex hull of the maximal schedules for the whole network.

For a subset of the links $L \subseteq E$, we can consider the interference relation among the links in L: The interference matrix is a submatrix of the original one with only those rows and columns corresponding to the links in L; the interference graph is the node-induced subgraph of the original interference graph. With this, we can talk about feasibility and maximality of schedules restricted to L, which are those defined with respect to the interference submatrix or subgraph. Similarly, we define M_L to be the matrix (set) of maximal schedules restricted to L, where each column of M_L is $|L|$-dimensional 0-1 vector. Let $Co(M_L)$ denote the convex hull of these maximal schedules.

We assume a time-slotted system, where each slot is of a unit length. We assume the traffic arrival processes to different links are independent on each other. For simplicity, we assume that each arrival process to a link is IID over time. This assumption can be relaxed provided the resulting queueing process is Markovian. (See [8] [12] for the reasons.) For each link $l \in E$, the average arrival rate is denoted by λ_l. We assume single-hop traffic: The traffic is transmitted over only one link and leaves the network after the transmission. Extension to the multi-hop traffic situation needs ideas from [9], but will not be further considered in this paper.

The capacity region Λ of a network is defined to be the set of all arrival rate vectors $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_{|E|})'$ that can be supported by a time sharing of the feasible schedules. It is easy to see that

$$\Lambda = \{\lambda \mid 0 \leq \lambda \leq \mu \text{ for some } \mu \in Co(M_E)\}. \quad (1)$$

In the above, $\lambda \leq \mu$ means λ is component-wise less than or equal to μ. We also need the notion of the capacity region for a set of links $L \subseteq E$. This region is defined analogously by replacing $Co(M_E)$ with $Co(M_L)$ in (1) and is denoted by Λ_L. We define the *interior* of Λ as follows, which is denoted by Λ^o.[3]

$$\Lambda^o = \{\lambda \mid 0 \leq \ \lambda < \mu \text{ for some } \mu \in Co(M_E)\}. \quad (2)$$

[1]This is in contrast to the *physical* interference model, where the link rate depends on the power levels of the interfering links in the neighborhood.

[2]An implicity assumption is that all active links transmit at the same constant rate. As discussed in [8], the main local pooling-related results are extensible to the case where an active link may transmit at different rates. However, we do not explore this extension in the paper.

[3]This definition of interior shouldn't be confused with the usual definition of interior of a set in mathematical analysis, which is an open set.

The interior of Λ_L is similarly defined and is denoted by Λ_L^o.

Tassiulas and Ephremides have shown that the interior of the capacity region can be stabilized by using the *maximum weighted schedule* (MWS) in each time slot where the weights are the queue sizes at the links [6]. In other words, for any arrival process whose average rate vector λ is in Λ^o, the resulting queueing process is stable if the MWS is used. Here, stability means the queueing process is a positive recurrent Markov process.

However, finding the MWS is a difficult problem. For the queue length vector at time t, $Q(t)$, the problem is to find the schedule $m^*(t)$ such that

$$m^*(t) \in \text{argmax}_{m \in M_E} Q(t)' m. \tag{3}$$

Hence, the problem is to find the maximum weighted independent set in the interference graph, which is NP-hard for the family of all graphs. Even under the more restricted k-hop interference model, the problem is still NP-hard for $k \geq 2$ [4] [5]. Under the 1-hop interference model, the problem becomes finding the maximum weighted matching, which can be solved in $O(|V|^3)$. However, the complexity is still very high.

The LQF schedule can be viewed as an approximation to the MWS. LQF operates as follows at each time slot t. The link with the largest backlog is activated and all links interfering with it are discarded. Next, the same procedure is applied to the remaining links: The link with the largest backlog among the remaining links is found and activated and the interfering links are discarded. The procedure is applied recursively until all links are either activated or discarded. The LQF algorithm requires sorting the backlog values; however, this can be done reasonably efficiently in a distributed manner. Further improvement has been made by Joo [13], whose local greedy scheduling does not require global sorting of the queues. His algorithm is to have the links with the local maximum queue sizes in their neighborhoods to transmit data, subject to the interference constraints. He showed that the local scheduling algorithm achieves the same stability region as LQF. Hence, from the MWS to LQF, a globally optimized decision is replaced by distributed, local decisions.

Notations: The following notations are used. Given a vector u, let u' denote the transpose of u. For two vectors u and v, $u'v$ means the inner product of the two; $u \leq v$ means u is component-wise less than or equal to v; the meanings of $u \geq v$, $u < v$ or $u > v$ are also component-wise. The symbols $\ngtr$ and $\ngeq$ are the negations of $>$ and $\geq$, respectively. For instance, by $u \ngtr v$, we mean that u is not component-wise greater than v; that is, there exists a component k such that $u_k \leq v_k$. For a vector $\lambda \in \mathbb{R}^{|E|}$ and for a set of links $L \subseteq E$, we denote the $|L|$-dimensional vector $[\lambda]_L$ as the restriction of λ to those dimensions associated to L.

III. σ-Local Pooling and the Performance of LQF

In this section, we introduce our main notion of *link* σ-local pooling and relate it to the original σ-local pooling (for the network graph) by Joo et al. [1]. We will call the latter *network* σ-local pooling. We show how to capture a larger stability region of LQF using this new notion of link σ-local pooling. By stability region, we mean a rate region in Λ for which LQF leads to a stable queueing process. We start by reviewing the σ-local pooling related results in [1].

A. Review of Network σ-Local Pooling by Joo et al.

Joo et al. investigated a single-parameter performance characterization of a scheduling policy [1]. In this case, the scheduling policy is LQF. They defined the efficiency ratio $\gamma^*(G)$ of a scheduling policy as follows

Definition 1: The efficiency ratio $\gamma^*(G)$ of a scheduling policy for a given network graph G is:

$$\gamma^*(G) := \sup\{\gamma | \text{The network is stable for all arrival rate vectors } \lambda \in \gamma\Lambda^o\}.$$

Joo et al. showed that $\gamma^*(G)$ is equal to the σ-local pooling factor for the network G, i.e.,

$$\gamma^*(G) = \sigma^*(G). \tag{4}$$

$\sigma^*(G)$ depends only on the topological and interference structures of the network and is defined as follows.

Definition 2: The *local pooling factor* $\sigma^*(G)$ of a network graph $G = (V, E, \mathcal{I})$ is:

$$\sigma^*(G) := \sup\{\sigma | \sigma\mu \ngeq \nu \text{ for all } L \subseteq E \text{ and all } \mu, \nu \in Co(M_L)\} \tag{5}$$

$$:= \inf\{\sigma | \sigma\mu \geq \nu \text{ for some } L \subseteq E \text{ and some } \mu, \nu \in Co(M_L)\}. \tag{6}$$

B. Motivation of Defining Link σ-Local Pooling

Note that, in the case of $\gamma^*(G) < 1$, $\gamma^*(G)\Lambda$ shrinks all dimensions of the capacity region by the same scaling factor to ensure stability. This can be overly conservative in that some links may be required to reduce the arrival rates more than necessary. This means that using a single factor to characterize the performance will underestimate the achievable rates of these links. Since links are all different in terms of the interference constraints they face, the reduction factors should be non-uniform across the links. Consider the network examples in Fig. 1 under the 1-hop interference model. The interference graph for the 6-cycle network is still a 6-cycle, which has been well studied in [8], [1], [10]. It has been shown that for the 6-cycle, $\gamma^*(G) = \sigma^*(G) = 2/3$. This is also the case for the 8-link network, although the interference graph is slightly more complicated.

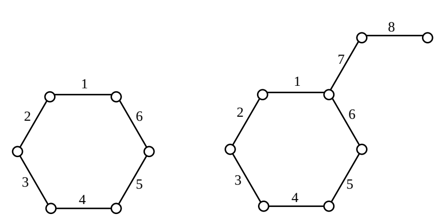


Fig. 1. Two networks with different performance behavior from the link perspective. Left: uniform; Right: heterogeneous.

However, for link 8 of the 8-link network on the right, there is no need to reduce its arrival rate by $2/3$ to achieve stability.

To see this, for any arrival rate vector inside the capacity region, we have $\lambda_7 + \lambda_8 < 1$. Under 1-hop interference, exactly one link from link 7 and link 8 must be activated in any maximal schedule, provided queue 8 (i.e., the queue of link 8) is not empty. Hence, the total queue length of the two must decrease when queue 8 is non-empty. The decrease will continue until queue 8 becomes empty. The time taken by this is bounded by the time taken for queue 7 to drift down to the size of queue 8, if queue 7 starts to be larger than queue 8, plus the time taken for both queues to drift down to zero from that point. Hence, if queue 7 is stable, then queue 8 is also stable.

The reason why links have different performance efficiency is that the efficiency ratio is related with the structure of certain "bottleneck" subgraph containing the link in question. The characterization of efficiency is determined by this subgraph but not by the whole graph in general. For different links, such subgraphs are different. For instance, for link 2 in the 8-link network in Fig. 1, the subgraph is the six cycle, which leads to its efficiency ratio to be $2/3$. For link 8, the subgraph is link 8 itself, which leads its efficiency ratio to be 1. We will later identify this kind of subgraphs more explicitly.

C. Definitions of Link σ-Local Pooling

The above discussion motivates us to come up with a performance characterization for each link individually. We next extend the concept of *network* σ-local pooling in Definition 2 to *link* σ-local pooling. This characterization is obtained from the topology and interference structure of a sub-network graph containing link l in question.

Definition 3: Given a network graph $G = (V, E, \mathcal{I})$, the *local pooling factor of a link $l \in E$*, denoted by σ_l^*, is

$$\sigma_l^* := \sup\{\sigma | \sigma\mu \not> \nu \text{ for all } L \subseteq E \text{ such that } l \in L, \text{ and all } \mu, \nu \in Co(M_L)\} \tag{7}$$

$$:= \inf\{\sigma | \sigma\mu > \nu \text{ for some } L \subseteq E \text{ such that } l \in L, \text{ and some } \mu, \nu \in Co(M_L)\}. \tag{8}$$

Compared with Definition 2, the subset L in Definition 3 is required to contain link l. Moreover, in (8), the inequality is strict as opposed to the non-strict inequality in (6). This in turn leads to the difference ($\not>$ as opposed to $\not\geq$) between the two supremum-based definitions. We will show later that the strict inequality is a tighter condition for the stability proof. However, the σ_l^* value is unchanged whether the strict or non-strict inequality is used.

The following lemma relates the original network σ-local pooling with the new link σ-local pooling. The proof is obvious after we replace the strict inequality with a non-strict inequality in (8).

Lemma 1: For a network graph $G = (V, E, \mathcal{I})$, the following holds.

$$\sigma^*(G) = \min_{l \in E} \sigma_l^*. \tag{9}$$

D. Link-Based Performance Guarantee of LQF

Lemma 1 makes it clear that the original network σ-local pooling factor in [1] is equal to the smallest of the σ-local pooling factors for the links. In [1], the LQF performance is bounded by the lowest link σ-local pooling factor. In contrast, we can show the following improved performance bound for LQF, which takes into account all σ_l^*. Let $\Sigma^*(G)$ be the $|E| \times |E|$ diagonal matrix whose diagonal entries are σ_l^* for $l \in E$. That is, $\Sigma^*(G) = \text{diag}(\sigma_l^*)_{l \in E}$. The next theorem is one of the main results in the paper.

Therorem 2: Given a network graph $G = (V, E, \mathcal{I})$, if an arrival rate vector λ satisfies $\lambda \in \Sigma^*(G)\Lambda^o$, then, the network is stable under the LQF policy.

Proof: We will consider the fluid limit of the queue process, denoted by $\{q_l(t)\}_{t \geq 0}$, for all $l \in E$. (See [12] [8] [1] for more details on this approach.) Consider a fixed time instance t. Let L be the set of those longest queues (with equal length) whose time derivatives at t, $\dot{q}_l(t)$, are the largest (also identical) under the given LQF policy (that is, an instance of the LQF policy that is being used.). The queues in L will remain the longest with identical length in the next infinitesimally small time interval.

Let $l \in \text{argmax}_{k \in L} \sigma_k^*$. Since $\lambda \in \Sigma^*(G)\Lambda^o$, there exists $\mu \in Co(M_E)$ such that $\lambda < \Sigma^*(G)\mu$. This implies $[\lambda]_L < \Sigma_L^*[\mu]_L$, where Σ_L^* denotes the restriction of $\Sigma^*(G)$ to L, i.e., the diagonal submatrix of $\Sigma^*(G)$ with only the rows and columns corresponding to the set L. Hence, $[\lambda]_L < \sigma_l^*[\mu]_L$.

It is easy to see that $[\mu]_L \in \Lambda_L$. Hence, there exists $\mu_L \in Co(M_L)$ such that $[\mu]_L \leq \mu_L$. Then, $[\lambda]_L < \sigma_l^* \mu_L$. Given λ, let us suppose the way of picking such a μ_L is well-defined. Let $\epsilon_L = \min_{k \in L}(\sigma_l^* \mu_L(k) - \lambda(k))$.[4] We have $\epsilon_L > 0$.

For such fixed λ and μ_L, consider any arbitrary $\nu_L \in Co(M_L)$. We must have $\sigma_l^* \mu_L \not> \nu_L$ by the definition of link σ-local pooling. Hence, there exists a link $k \in L$ such that $\sigma_l^* \mu_L(k) \leq \nu_L(k)$. For such a k, since $\lambda(k) < \sigma_l^* \mu_L(k) \leq \nu_L(k)$, we have $\nu_L(k) - \lambda(k) \geq \epsilon_L$. Hence, $\max_{k \in L}(\nu_L(k) - \lambda(k)) \geq \epsilon_L$. Note that ϵ_L is independent of ν_L.

Note that the service rate vector, when restricted to L, must belong to the set $Co(M_L)$. Roughly, this is because L contains all the queues that are among the longest and remain longest in the near future, and hence, every LQF schedule being used must be a maximal schedule when restricted to L. (See [12] [8] for a more rigorous argument for this.)

Now imagine ν_L is the service rate vector at the current time t. We have just shown that, for some $k \in L$, $\nu_L(k) - \lambda(k)$ is at least ϵ_L. Hence, the queue at link k decreases at a rate no less than ϵ_L. Since the queues in the set L change at the same rate, they all decrease at a rate no less than ϵ_L. Hence, each of the longest queues decreases its size at a rate no less than ϵ_L. Let $\epsilon = \min\{\epsilon_L | L \subseteq E\}$. Since the number of possible subsets of E is finite, we have $\epsilon > 0$. Hence, at any time instance, each of the longest queues decreases at a positive rate no less than ϵ. By [12], this is sufficient to conclude that the original queueing process is a positive recurrent Markov process, which means the queues are stable by definition. ■

To summarize, Joo et al. showed in [1] that the region $\sigma^*(G)\Lambda^o$ is stable under LQF. Our Theorem 2 shows

[4] Given a vector ν, we write its component corresponding to link k by ν_k or $\nu(k)$ interchangeably.

that the region $\Sigma^*(G)\Lambda^o$, which contains $\sigma^*(G)\Lambda^o$, is stable under LQF. For the example of the 8-link network in Fig. 1, $\sigma^*(G) = 2/3$ whereas $\Sigma^*(G) = \text{diag}(2/3, 2/3, 2/3, 2/3, 2/3, 2/3, 1, 1)$.[5]

IV. Theory of Link σ-Local Pooling

In this section, we show some important properties of the newly defined concept of link σ-local pooling and the related concept of *limiting set*. In general, these concepts are difficult to work with since their definitions involve combinatorial enumerations. Our objective is to provide tools for using or applying these concepts. As will be shown, some of the theories developed in this section can help to estimate the link σ-local pooling factors. We will also argue that link σ-local pooling and limiting set are fundamental concepts. The understanding of them may help to reveal deeper structures and key insights about wireless link scheduling.

A. σ-Local Pooling of a Set

In order to develop the intended theory, it is convenient to first define *set* σ-local pooling. This concept can be used as a building block for performance characterization of the LQF policy for a set of links in the network graph. Since links in a wireless network may exert influence on each other, it is natural to study a set of links as a whole.

Suppose $L \subseteq E$ and L is non-empty. For convenience, let

$$\Theta_L = \{\sigma \mid \sigma\mu_L \not> \nu_L, \text{for all } \mu_L, \nu_L \in Co(M_L)\}. \quad (10)$$

The compliment of Θ_L is

$$\Theta_L^c = \{\sigma \mid \sigma\mu_L > \nu_L, \text{for some } \mu_L, \nu_L \in Co(M_L)\}. \quad (11)$$

Definition 4: Given a non-empty set $L \subseteq E$, we say L has a *set σ-local pooling factor σ_L^* if the following holds.*

$$\sigma_L^* := \sup\{\sigma \mid \sigma \in \Theta_L\} \quad (12)$$
$$:= \inf\{\sigma \mid \sigma \in \Theta_L^c\}. \quad (13)$$

Note that, unlike the definition of network σ-local pooling in Definition 2, the definition of set σ-local pooling for a set L does not involve subsets of L.

The following are some elementary facts. Since $0 \notin Co(M_L)$, by (11), $\sigma_L^* > 0$. By considering $\mu_L = \nu_L$ in (10), we see that $\sigma_L^* \leq 1$. If $\sigma \in \Theta_L^c$, then $(\sigma - \epsilon)\mu_L > \nu_L$ for small enough $\epsilon > 0$, where L, μ_L and ν_L are as in the definition of Θ_L^c. Hence, Θ_L^c is an open set on $\mathbb{R}$. In fact, $\Theta_L^c = (\sigma_L^*, \infty)$; $\Theta_L = [0, \sigma_L^*]$.

The following lemma says that σ_L^* can be found by a well-defined optimization problem where the constraint region is a closed set. The constraint region can be thought as being compact when taking into account the fact that σ_L^* is bounded from above by 1. The fact in the lemma needs to be explicitly stated since the infimum-based definition in (13) is not over a closed set in variables (σ, μ_L, ν_L).

[5]This may not be obvious now, but can be shown easily by applying the theory to be developed subsequently.

Lemma 3: For any non-empty $L \subseteq E$, σ_L^* is the optimal value of the following optimization problem.

$$\textbf{(I)} \min_{\sigma, \mu_L, \nu_L} \sigma \quad (14)$$
$$\text{subject to } \sigma\mu_L \geq \nu_L \quad (15)$$
$$\mu_L, \nu_L \in Co(M_L). \quad (16)$$

Proof: By (11) and (13), σ_L^* is the optimal value of the following problem.

$$\textbf{(II)} \inf_{\sigma, \mu_L, \nu_L} \sigma \quad (17)$$
$$\text{subject to } \sigma\mu_L > \nu_L \quad (18)$$
$$\mu_L, \nu_L \in Co(M_L). \quad (19)$$

Let the constraint sets of the optimization problem (I) and (II) be denoted by S_1 and S_2 ($S_2 = \Theta_L^c$), respectively, both of which lie in $\mathbb{R} \times \mathbb{R}^{|L|} \times \mathbb{R}^{|L|}$. We will show that S_1 is the closure of S_2. For this, we need to show that every point in $S_1 \backslash S_2$ is a limit point of S_2. Let a point $(\sigma, \mu_L, \nu_L) \in S_1 \backslash S_2$, which is characterized by $\sigma\mu_L \geq \nu_L$ with $\sigma\mu_k = \nu_k$ for some $k \in L$. If $\mu_L > 0$, we only need to increase σ by a little bit to find a point in S_2. To handle the general case where $\mu_k = 0$ for some $k \in L$, note that $\nu_k = 0$. Since M_L includes all maximal vectors, there exists $\omega_L \in Co(M_L)$ such that $\omega_L > 0$. Then, a vector $\hat{\mu}_L = (1 - \epsilon_1)\mu_L + \epsilon_1\omega_L$, where $0 < \epsilon_1 \leq 1$, has the property that $\hat{\mu}_L > 0$. Note that $\hat{\mu}_L \in Co(M_L)$. We can choose ϵ_1 small enough and choose $\epsilon_2 > 0$ accordingly such that $(\sigma + \epsilon_2)\hat{\mu}_L > \nu_L$ and also $(\sigma + \epsilon_2, \hat{\mu}_L, \nu_L)$ is in the ϵ-open ball around (σ, μ_L, ν_L). Hence, S_1 is the closure of S_2.

Next, $(\sigma, \mu_L, \nu_L) \mapsto \sigma$ is a continuous function on $\mathbb{R} \times \mathbb{R}^{|L|} \times \mathbb{R}^{|L|}$. Hence, the two problems have the same optimal value and the optimum is attained for problem (I). ■

Since the optimization problem (I) has a continuous objective function and the constraint set is closed, the optimum is attained in its constraint set. The optimization problem (II) is not attained. The following lemma makes this more precise.

Lemma 4: For any non-empty set $L \in E$, the optimum solution to the optimization problem (I) satisfies $\sigma_L^*\mu_L^* \geq \nu_L^*$ with $\sigma_L^*\mu_k = \nu_k$ for some $k \in L$, where $\mu_L^*, \nu_L^* \in Co(M_L)$ and σ_L^* is the σ-local pooling factor for set L. Furthermore, such k is not unique.

Proof: Suppose we have $\sigma_L^*\mu_L^* > \nu_L^*$. Then, $(\sigma_L^* - \epsilon)\mu_L^* \geq \nu_L^*$ for small enough $\epsilon > 0$, and σ_L^* cannot be optimal.

We next show such k is not unique; in other words, there are at least two components achieving equality in an optimal solution. The proof is by contradiction. Suppose there is only one k such that $\sigma_L^*\mu_k^* = \nu_k^*$, and $\sigma_L^*\mu_i^* > \nu_i^*$ for all $i \neq k, i \in L$. Consider two cases. The first case is $\sigma_L^*\mu_k^* = \nu_k^* > 0$. Note that there is a vector $\hat{\nu} \in Co(M_L)$ such that $\hat{\nu}_k = 0$. By choosing a small enough $\epsilon > 0$, we can obtain a new vector $\tilde{\nu} = (1 - \epsilon)\nu^* + \epsilon\hat{\nu}$ such that $\sigma_L^*\mu_i^* > \tilde{\nu}_i$ for all $i \in L$. Consider the second case where $\sigma_L^*\mu_k^* = \nu_k^* = 0$. Note that there must be a vector $\hat{\mu} \in Co(M_L)$ such that $\hat{\mu}_k > 0$. By choosing a small enough $\epsilon > 0$, we can obtain a new vector $\tilde{\mu} = (1 - \epsilon)\mu^* + \epsilon\hat{\mu}$ such that $\sigma_L^*\tilde{\mu}_i > \nu_i^*$ for all $i \in L$. In either case, the conclusion contradicts the definition of σ_L^* (or equivalently, the first part of this lemma). ■

The optimization characterization of σ_L^* is useful since one can apply the duality theory to derive important results and insights. The problem (I) has an alternative form, which is a linear program. From the linear program, we can obtain the following dual problem. Suppose M_L has $c(L)$ columns. Let e_n be the vector $(1, 1, \ldots, 1)'$ with n 1's.

Lemma 5: σ_L^* is the optimal value of the following optimization problem.

$$\textbf{(Dual)} \quad \max_{x \geq 0, w} \; w \quad \text{subject to} \quad x' M_L \leq e'_{c(L)}, \quad x' M_L \geq w e'_{c(L)}.$$

Proof: The problem (Dual) is the dual problem of a linear version of the problem (I). First, rewrite the problem (I).

$$\begin{aligned} &\min_{\alpha,\beta,\sigma} \; \sigma \\ &\text{subject to} \quad \sigma M_L \alpha \geq M_L \beta \\ &\qquad \alpha' e_{c(L)} = 1 \\ &\qquad \beta' e_{c(L)} = 1 \\ &\qquad \alpha, \beta \geq 0. \end{aligned}$$

Let $\gamma = \sigma \alpha$. The problem can be written as,

$$\begin{aligned} &\min_{\gamma,\beta,\sigma} \; \sigma && (20) \\ &\text{subject to} \quad M_L \gamma \geq M_L \beta && (21) \\ &\qquad \gamma' e_{c(L)} = \sigma && (22) \\ &\qquad \beta' e_{c(L)} = 1 && (23) \\ &\qquad \gamma, \beta \geq 0. && (24) \end{aligned}$$

Let x, y, z to be dual variables associated with (21), (22) and (23), respectively. Then, the dual problem is

$$\begin{aligned} &\max_{x \geq 0, z} \; -z \\ &\text{subject to} \quad x' M_L + z e'_{c(L)} \geq 0 \\ &\qquad y e'_{c(L)} - x' M_L \geq 0 \\ &\qquad y = 1. \end{aligned}$$

Let $w = -z$, we get the optimization problem in the lemma. ■

Remark 1: From Lemma 5, a set $L \subseteq E$ is σ_L^*-local pooling if and only if σ_L^* is the largest number for which $\sigma_L^* e'_{c(L)} \leq x' M_L \leq e'_{c(L)}$ holds for some $x \geq 0$. In the special case of $\sigma_L^* = 1$, we see that L is local pooling if and only if there exists some $x \geq 0$ such that $x' M_L = e'_{c(L)}$, or equivalently, there exists some nonzero $x \geq 0$ such that the components of the vector $x' M_L$ are all identical. The latter statement coincides with the original definition of local pooling in [8].

Remark 2: The optimization problem (20)-(24) can be rewritten as follows.

$$\begin{aligned} &\min_{\gamma,\beta} \; \sum_{i=1}^{c(L)} \gamma_i \\ &\text{subject to} \quad M_L \gamma \geq M_L \beta \\ &\qquad \beta' e_{c(L)} = 1 \\ &\qquad \gamma, \beta \geq 0. \end{aligned}$$

Consider an optimal solution (γ^*, β^*) and $\sigma^* = \sum_{i=1}^{c(L)} \gamma_i^*$. We can interpret $\nu_L^* = M_L \beta^*$ as achieving the service rates ν_L^* by time sharing of the maximal schedules with the time shares β_i^*. Then, $M_L \gamma^*$ is an alternative way of time sharing the maximal schedules that achieves at least link rates ν_L^*, but with the least amount of time, σ^*. That is, γ^* is the most compact schedule in terms of time. Therefore, σ^* is the largest degree (smallest number) at which any time-sharing schedule (i.e., one in $Co(M_L)$) can be packed.

B. Relation between Set and Link σ-Local Pooling

The development of set σ-local pooling serves as a basis for better understanding of link σ-local pooling. The performance limitation of a link is related to all subsets of links containing the link itself. Therefore, some of the results about set σ-local pooling can be applied here. The performance of the links is generally not uniform due to the fact that each link is associated with a different collection of subsets.

Lemma 6: For a link $l \in E$, σ_l^* is the smallest σ_L^* for all $L \subseteq E$ that contains l, i.e.,

$$\sigma_l^* = \min_{\{L \subseteq E \mid l \in L\}} \sigma_L^* \tag{25}$$

Proof: The proof is by definition of σ_l^* and σ_L^*. ■

We have the following lemma indicating the relationship between the local pooling factor for sets and the local pooling factor for links.

Corollary 7: Let $L \subseteq E$ be an arbitrary non-empty set. For all $l \in L$, $\sigma_L^* \geq \sigma_l^*$.

C. Limiting Set

We next study the situation where the set σ-local pooling factor is equal to the link σ-local pooling factor for a link in the set. When equality holds, we see that the efficiency ratio, i.e., the link σ-local pooling factor, is limited by the set.

Loosely speaking, a limiting set for a link l is a subset of the links, $L \subseteq E$ with $l \in L$, that "achieves" σ_l^* (for instance, see the infimum definition of σ_l^* in (8)). The significance of a limiting set is that it is the set of links whose interference with l prevents σ_l^* from becoming larger, hence, the term *limiting*. Therefore, it is the limiting set for a link, instead of the complete network, that represents structural constraints for the link. While the network can be large, the limiting set for a link may contain a much smaller number of links. Hence, finding the limiting set and understanding its properties have both theoretical and practical significance.

Definition 5: For any link $l \in E$, a set $L \subseteq E$ is called a *limiting set for link* l if $l \in L$ and there exist $\mu_L, \nu_L \in Co(M_L)$ such that $\sigma_l^* \mu_L \geq \nu_L$.

Lemma 8: For any link l, a limiting set for l exists.

Proof: The proof is omitted for brevity. ■

Note that the limiting set for a link is not necessarily unique.

Lemma 9: A set $L \subseteq E$ containing link l is a limiting set for l if and only if $\sigma_L^* = \sigma_l^*$.

Proof: Suppose L is a limiting set for l. Then, there exist $\mu_L, \nu_L \in Co(M_L)$ such that $\sigma_l^* \mu_L \geq \nu_L$. By Lemma 3, $\sigma_L^* \leq \sigma_l^*$. Combining this with Corollary 7, we have $\sigma_L^* = \sigma_l^*$.

Conversely, suppose $\sigma_L^* = \sigma_l^*$. Then, by Lemma 3, there exist $\mu_L, \nu_L \in Co(M_L)$ such that $\sigma_l^* \mu_L \geq \nu_L$. By the definition, L is a limiting set for l. ■

Corollary 10: Given a non-empty set of links $L \subseteq E$, if $\sigma_L^* = \max_{l \in L} \sigma_l^*$, then L is a limiting set for each link in the set $\mathrm{argmax}_{l \in L} \sigma_l^*$.

For a link l with $\sigma_l^* = 1$, any set L containing l is a limiting set for l, since we can choose $\mu_L = \nu_L$ in Definition 5. Hence, when $\sigma_l^* = 1$, the notion of limiting set is trivial, and the corresponding limiting sets are called *trivial*. Only when $\sigma_l^* < 1$, the notion is consequential.

Lemma 11: Consider a link $l \in E$ with $\sigma_l^* < 1$, and let L be a limiting set for l. In the interference graph G^I, every node (link in the network graph G) in L has a (interference) degree of at least 2 with respect to L. Hence, the subgraph of G^I induced by L contains at least one cycle.

Proof: The proof is by contradiction. Consider the interference graph, G^I. Suppose in the limiting set L for l, some node in L, say p, has a degree either 0 or 1 in L. Then, we can construct a vector x as follows. When p has a degree 0 in L, the entry of x corresponding to p is set to 1 and all other entries are set to 0; when p's degree in L is 1, let the entries corresponding to p and p's only neighbor in L be equal to 1, and set all other entries to 0. Then, the problem (Dual) in Lemma 5 has the optimal value 1, which implies $\sigma_L^* = 1$. By Corollary 7, $\sigma_l^* \geq \sigma_L^* = 1$. Hence, we have $\sigma_l^* = 1$, which contradicts the assumption of the current lemma. Therefore, every node in the limiting set L should have a degree at least 2 in L. Then, there must exist a cycle in the subgraph of G^I induced by the node set L. ■

Lemma 11 gives a necessary condition for any non-trivial limiting set. When we want to find a link σ-local pooling factor or the limiting set for a link, we can apply this lemma to reach conclusions or prune the search space. For instance, in the 8-link network of Fig. 1, link 8 has $\sigma_8^* = 1$, since in the interference graph, the node corresponding to link 8 has a degree 1. More generally, any link that interferes with no more than one other link must have a link σ-local pooling factor equal to 1.

Lemma 12: For any link $l \in E$, one of its limiting sets induces a connected subgraph in the interference graph for the network.

Proof: Let G^I denote the interference graph. Suppose L is an arbitrary limiting set for l. If L induces a connected subgraph of G^I, then there is nothing to prove. Otherwise, let $L' \subseteq L$ with $l \in L'$ be the largest subset of L that contains l and induces a connected subgraph of G^I. We now only need to show that L' is also a limiting set for l.

Since L is a limiting set for link l, by Lemma 9, we must have $\sigma_l^* = \sigma_L^*$. From Lemma 3, we know there are two vectors $\mu, \nu \in Co(M_L)$ such that $\sigma_L^* \mu \geq \nu$. Since L' is the largest subset of L containing l that induces a connected subgraph of G^I, any (physical) link in L' does not interfere with any link in $L \setminus L'$. Thus, $[\mu]_{L'}, [\nu]_{L'} \in Co(M_{L'})$ and $\sigma_l^* [\mu]_{L'} \geq [\nu]_{L'}$. Therefore, by Lemma 3, $\sigma_l^* \geq \sigma_{L'}^*$. Also, since $l \in L'$, we have $\sigma_l^* \leq \sigma_{L'}^*$ by Corollary 7. Hence, $\sigma_l^* = \sigma_{L'}^*$. According to Lemma 9, L' is a limiting set for l. ■

Remark: Lemma 12 shows that for any link l in the graph, only those subsets of the links (containing l) that induce connected subgraphs of the interference graph may further limit link l's performance under LQF. When we calculate the local pooling factor for link l, we only need to inspect these subsets.

D. Performance Guarantees of LQF - A Revisit

With the development of set σ-local pooling, we can state the following sufficient condition for stability under LQF.

Therorem 13: Given a network graph $G = (V, E, \mathcal{I})$, suppose the arrival rate vector λ satisfies the condition that, for every non-empty $L \subseteq E$, $[\lambda]_L \in \sigma_L^* \Lambda_L^o$. Then, the network is stable under the LQF policy.

Proof: Since the proof is similar to that for Theorem 2, we will be brief and omit some arguments, which can be found in the proof for Theorem 2. We will consider the fluid limit of the queue process, denoted by $\{q_l(t)\}_{t \geq 0}$, for all $l \in E$. Consider a fixed time instance t. Let L be the set of those longest queues (with equal length) whose time derivatives at t, $\dot{q}_l(t)$, are the largest (also identical) under the particular LQF policy being used.

By the assumption of the theorem, there exists $\mu_L \in Co(M_L)$ such that $[\lambda]_L < \sigma_L^* \mu_L$. For this μ_L and any other $\nu_L \in Co(M_L)$, $\sigma_L^* \mu_L \not\geq \nu_L$ by the definition of σ_L^*. Hence, there exists a link $k \in L$ such that $\sigma_L^* \mu_k \leq \nu_k$. Then, $\lambda_k < \nu_k$. If ν_L is the service rate vector (in the fluid limit) for the queues in L, the queue at link k decreases at the rate $\nu_k - \lambda_k$. Since all queues in the set L change at the same rate, they all decrease at the rate $\nu_k - \lambda_k$, which is positive. ■

With the relationship between set and link σ-local pooling, we can show Theorem 2 is implied by Theorem 13. Hence, the condition of Theorem 13 for stability under LQF is more general than that of Theorem 2. This shows one of the utilities provided by our theoretical development of set and link σ-local pooling.

Proof: (Alternative Proof of Theorem 2) Consider any link set $L \subseteq E$. Let $l \in \mathrm{argmax}_{k \in L} \sigma_k^*$. Since $\lambda \in \Sigma^*(G)\Lambda^o$, there exists $\mu \in Co(M_E)$ such that $\lambda < \Sigma^*(G)\mu$. This implies $[\lambda]_L < \Sigma_L^* [\mu]_L$, where Σ_L^* denotes the restriction of $\Sigma^*(G)$ to L, i.e., the diagonal submatrix of $\Sigma^*(G)$ with only the rows and columns corresponding to the set L. Hence, $[\lambda]_L < \sigma_l^* [\mu]_L \leq \sigma_L^* [\mu]_L$, where we have used Corollary 7 in the second inequality. It is easy to see that there exists $\hat{\mu}_L \in Co(M_L)$ such that $[\mu]_L \leq \hat{\mu}_L$. Hence, $[\lambda]_L \in \sigma_L^* \Lambda_L^o$. By Theorem 13, the queues are stable under LQF. ■

V. Estimating $\Sigma^*(G)$ Matrix

A. Estimating σ-Local Pooling Factor for Set

In Section IV-A, we introduced a linear programming formulation (LP), (20)-(24), for calculating the σ-local pooling factor for a set of links. Although linear programs can be solved in polynomial time in terms of the problem size, our formulation contains exponentially many decision variables and is computationally intractable for large networks. This section concentrates on providing methods to estimate σ_L^*. We find defining the problem on the interference graph to be simpler. Accordingly, the following observations are made primarily on the interference graph. Recall that a node in the interference graph corresponds to a link in the original network. As a result, a maximal schedule corresponds to a maximal independent set in the interference graph. Unless mentioned otherwise, the interference graph in this subsection refers to the subgraph of G^I induced by the set L.

Consider the dual problem in Lemma 5. We observe that the dual LP is a weight assignment problem on the nodes in the interference graph. Consider a fixed set $L \subseteq E$. Let $\{s^1, s^2, s^3, \dots, s^t\}$ represent all the maximal schedules with respect to set L, i.e., each s^i is the i^{th} column of the matrix M_L. Consequently, the dual problem can be rewritten as follows.

$$\max_{x \geq 0, w} \quad w \tag{26}$$

$$\text{subject to} \quad \max_i x' s^i \leq 1 \tag{27}$$

$$\min_i x' s^i \geq w. \tag{28}$$

Entries of the x vector in the dual problem can be interpreted as the weights assigned to the nodes in the interference graph. We define *the weight of a schedule* to be the sum of the weights of all active nodes in the schedule. The dual problem strives to balance the weights of the maximal schedules. It is easy to see that, in an optimal solution to the dual problem, denoted by (w^*, x^*), equality is achieved in both (27) and (28) by some schedules. Otherwise, the objective value can be further improved. Hence, in an optimal solution, the weight of any maximum-weight schedule is forced to 1. This can be interpreted as normalizing the weight assignment according to the maximum-weight schedule. The weight of any minimum-weight schedule is w^*. With some thought, the weight assignment problem can be reformulated as finding node weights to maximize the ratio between the minimum and the maximum schedule weights. That is,[6]

$$\sigma_L^* = w^* = \max_{x \geq 0} \frac{\min_i x' s^i}{\max_i x' s^i}. \tag{29}$$

The new formulation in (29) provides a simple way to derive a lower bound for σ_L^*, which is by assigning some particular weights to the nodes in the interference graph and calculating the ratio between the minimum and the maximum schedule weights. Next, we will use this idea to derive lower-bounds on σ_L^*. We denote the component sum of a vector s by $||s||_1$, which is the 1-norm of s. Then, we have the following.

Lemma 14: For a non-empty set $L \subseteq E$, suppose the maximal schedules are $M_L = (s^1, s^2, s^3, \dots, s^t)$ where s^i is a vector corresponding to the i^{th} maximal schedule with respect to L. Then,

$$\sigma_L^* \geq \frac{\min_i ||s^i||_1}{\max_i ||s^i||_1}. \tag{30}$$

Proof: In (29), we assign identical weights to all nodes in L, i.e, $x_j = 1$ for all $j \in L$. ■

Remark: A similar result was also given in [14] and [15].

For an interference graph that forms a single cycle, the lower bound in Lemma 14 is in fact achieved.

Lemma 15: Suppose the interference graph corresponding to L forms a cycle. Then,

$$\sigma_L^* = \frac{\min_i ||s^i||_1}{\max_i ||s^i||_1}. \tag{31}$$

Proof: See Corollary 23 later. ■

Next, we give a lower bound of σ_L^* for interference graphs that are cycles.

Lemma 16: Suppose the interference graph corresponding to L forms a cycle. Then, we have $\sigma_L^* \geq 2/3$.

Proof: In any maximal schedule for a cycle interference graph, there must be at least one node active among any three consecutive nodes. So, for any schedule s^i, $||s^i||_1 \geq |L|/3$. On the other hand, since any two consecutive nodes cannot be activated simultaneously, there are at most $|L|/2$ active nodes in s^i. Therefore, $||s^i||_1 \leq |L|/2$. By applying Lemma 15, we have $\sigma_L^* \geq 2/3$. ■

Based on the proof of Lemma 16, as the number of nodes in the cycle increases, σ_L^* eventually approaches $2/3$. Hence, only cycles with a small number of nodes (but greater than 6) can have σ_L^* significantly different from $2/3$.

The bounding approach in Lemma 14 works well for cycles. However, the following example illustrates that this approach can produce arbitrarily small lower bounds for some network topologies. Consider the interference graph in Fig. 2 with 9 nodes, which is an instance of the star graph S_k for $k = 9$. If we assign identical weights to all nodes in the graph and compute the ratio in Lemma 14, we will end up getting the ratio between the largest and smallest cardinality of the maximal schedules, which is $1/8$. As $k \to \infty$, the ratio approaches 0. However, σ_L^* of the network is 1 by Lemma 11, since the star interference graph contains no cycles.

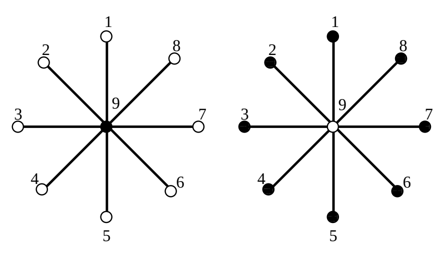


Fig. 2. A 9-node interference graph under equal node-weight assignment. Left: the minimum-weight schedule; Right: the maximum-weight schedules.

To improve the lower bound, we extend our weight assignment approach to subsets of L. For a set $L' \subseteq L$, let $||[s^i]_{L'}||_1$ represent the 1-norm of the vector $[s^i]_{L'}$, which is s^i restricted to the set L'.

[6]We assume the convention $\frac{0}{0} = 0$ so that $x = 0$ is not optimal.

Lemma 17: For a non-empty set $L \subseteq E$, suppose the maximal schedules are $M_L = (s^1, s^2, s^3, \ldots, s^t)$ where s^i is a vector corresponding to the i^{th} maximal schedule with respect to L. Then,

$$\sigma_L^* \geq \max_{L' \subseteq L} \frac{\min_i ||[s^i]_{L'}||_1}{\max_i ||[s^i]_{L'}||_1}. \quad (32)$$

Proof: Assign $x_j = 1$ for all nodes $j \in L'$ and $x_j = 0$ otherwise. ∎

Regardless of the weight assignment scheme on nodes, enumerating all the maximal schedules is intractable for large sets. As a result, it is difficult to find the schedules with the maximum or the minimum weights. This observation motivates us to find a simpler way to lower-bound σ_L^*. The following lemma states that it is possible to derive a lower bound for σ_L^* using the interference degree. The interference degree of a node is defined as the maximum number of nodes that can be scheduled simultaneously in the node's single-hop neighborhood, where the hop count is measured in the interference graph [1]. Since the interference graph here is restricted to L, the interference degree of a node $i \in L$ is denoted by $d_L(i)$.

Lemma 18: Let $d^* = \min_{i \in L} d_L(i)$. Then, we have $\sigma_L^* \geq 1/d^*$.

Proof: Let $l^* \in \text{argmin}_{i \in L} d_L(i)$. Let $L' \subseteq L$ be the set of neighbors of l^* in the interference graph (restricted to L), i.e., the largest subset such that each $l \in L'$ interferes with l^*. Here, we use the convention that $l^* \in L'$.

The maximum number of nodes in L' that can be activated simultaneously is d^*. There is at least one node that must be activated in L' for any maximal schedule. By applying Lemma 17, we have $\sigma_L^* \geq 1/d^*$. ∎

Remark: Lemma 17 can be used to derive Proposition 3 in [1].

B. Estimating σ-Local Pooling Factor for Link

By Lemma 9, the σ-local pooling factor for a link is equal to the σ-local pooling factor for its limiting set; by Lemma 12, there is a connected limiting set in the interference graph. In this part, we concentrate on deriving lower bounds for link performance based on these facts.

We present an algorithm (Algorithm 1), which provides a lower bound for the local pooling factor for each link based on its interference degree. Unlike the algorithm in [1], which aims at finding a single performance bound for the whole graph, our algorithm finds a separate performance bound for each link.

Therorem 19: Given $G = (V, E, \mathcal{I})$, let $(\sigma_l)_{l \in E}$ be the values returned by Algorithm 1. Then, $\sigma_l^* \geq \sigma_l$ for each l.

Proof: Consider the first round where L_1 contains all links. Suppose there is a link l satisfying $d_{L_1}(l) \leq d$. Then, if the link has a limiting set L, we have $d_L(l) \leq d$ since $L \subseteq L_1$. By applying Lemma 9 and Lemma 18, $\sigma_l^* = \sigma_L^* \geq 1/d_L(l) \geq 1/d = \sigma_l$.

For any later round, when a link l is chosen to be removed, there are two possibilities: (1) Link l's limiting set L contains a previously removed link; or (2) link l's limiting set L does not contain any previously removed link. If case (1) is true, we can assume L contains a previous removed link k. By Corollary 7, $\sigma_l^* = \sigma_L^* \geq \sigma_k^*$. Due to the monotonicity of d, $d_k \leq d_l$ where d_k is the d value when k is removed and d_l is the d value when l is removed. Hence, $\sigma_l^* = \sigma_L^* \geq \sigma_k^* \geq 1/d_k \geq 1/d_l$. Since $\sigma_l = 1/d_l$, we have $\sigma_l^* \geq \sigma_l$. In case (2), since the limiting set L for l does not contain any previously removed links, we must have $L \subseteq L_i$, and hence, $d_L(l) \leq d_{L_i}(l)$. Thus, $\sigma_l^* = \sigma_L^* \geq 1/d_{L_i}(l) \geq 1/d_l = \sigma_l$. Therefore, in either case, $\sigma_l^* \geq \sigma_l$ holds. ∎

Algorithm 1 σ-Local Pooling for Each Link

```
INPUT: A graph G = (V, E, I)
OUTPUT: σ-local pooling factors for all links, (σ_l)_{l∈E}
Initialization: L_1 ← E, d ← 1
for all 1 ≤ i ≤ |E| do
    Choose a link l from L_i with the minimum interference
    degree restricted to L_i.
    if d_{L_i}(l) ≤ d then
        σ_l = 1/d
        L_{i+1} ← L_i \ l
        i ← i + 1
    else
        d ← d + 1
    end if
end for
Return (σ_l)_{l∈E}.
```

Remark: Algorithm 1 can be further compared with the similar algorithm in [1]. Both algorithms generate a sequence of links $l_1, l_2, \ldots, l_{|E|}$ as they progress (but not the same sequence in general). Our algorithm produces a sequence of values $\{d_{l_i}\}_{1 \leq i \leq |E|}$, where d_{l_i} is the d value before link l_i is removed from the graph. The algorithm in [1] also implicitly generates a sequence of values $\{\hat{d}_{l_i}\}_{1 \leq i \leq |E|}$. In the end, our algorithm returns $(\sigma_{l_i})_{1 \leq i \leq |E|}$, where each σ_{l_i} is a lower bound of $\sigma_{l_i}^*$ and $\sigma_{l_i} = 1/d_{l_i}$ for each l_i. The algorithm in [1] returns $1/d_e$ as a lower bound of $\sigma^*(G)$, where $d_e = \max_i \hat{d}_{l_i}$. The following fact can be shown.

Lemma 20: Every σ_l returned by Algorithm 1 is greater than or equal to the returned value $1/d_e$ by the algorithm of [1].

Proof: The proof is by contradiction. Suppose there exists a link p with $\sigma_p < 1/d_e$. According to Algorithm 1, $\sigma_p = 1/d_p$, where d_p is the d value before p is removed from the graph. Then, $1/d_p = \sigma_p < 1/d_e$, which implies that $d_p > d_e$ or $d_p - 1 \geq d_e$. In Algorithm 1, d is increased by 1 only when in some round i, there is no link l in L_i such that $d_{L_i}(l) \leq d$. Hence, there exists a round i such that every link l in L_i satisfies that $d_{L_i}(l) > d_p - 1 \geq d_e$.

Now, consider the algorithm of [1] and consider any sequence of link removals in that algorithm. Suppose k is the first link removed from set L_i by that algorithm. Assume that just before removing k, the remaining set of links is $\tilde{L}$. Then $L_i \subseteq \tilde{L}$. This implies that $d_{L_i}(k) \leq d_{\tilde{L}}(k) \leq d_e$. This leads to a contradiction, because $d_{L_i}(k) > d_p - 1 \geq d_e$. Therefore, the lemma holds. ∎

Example: Consider the interference graph in Fig. 3, which

contains a 6-cycle connected with a tree. For this graph, Algorithm 1 works as follows.
1. Initially, $d = 1$ and $L_1 = E$.
2. Every leaf node, say l, satisfies $d_{L_1}(l) \leq d = 1$. Pick one leaf node l_1 and assign $\sigma_{l_1} = 1/d = 1$. Then, remove l_1 from L_1 to get L_2.
3. Continue to remove leaf nodes from the tree until only the cycle is left. Suppose this takes $k-1$ steps. Then, the node set of the cycle is L_k. For a cycle, we cannot find any node l with $d_{L_k}(l) \leq d = 1$. Hence, we increase d to 2.
4. Find node l_k in the cycle with $d_{L_k}(l_k) \leq d = 2$. Assign $\sigma_{l_k} = 1/d = 1/2$. Then, remove l_k from L_k to get L_{k+1}.
5. From now on, we can always find a node l_i from the remaining graph L_i, satisfying $d_{L_i}(l_i) \leq d = 2$. We obtain $\sigma_{l_i} = 1/2$ for all remaining nodes.

For the same graph, the algorithm of [1] can obtain a lower bound for $\sigma^*(G)$ no greater than $1/2$ for the following reason. Since there is a six-cycle in the graph, we know $\sigma^*(G) \leq 2/3$; but, $1/2$ is the best value the algorithm can obtain other than 1. Let $\sigma(G)$ denote the lower bound returned by the algorithm ($\sigma(G) = 1/d_e$). Comparing the two algorithms, we see that $\sigma_l \geq \sigma(G)$ for all l. Moreover, our algorithm obtains the exact σ_l^* for every node l on the tree; the algorithm of [1] underestimates the performance of those nodes on the tree by using a lower bound of σ_p^* for a node p in the cycle. Imagine the tree has many nodes. We see that a small part of the network can limit the performance characterization of the entire network.

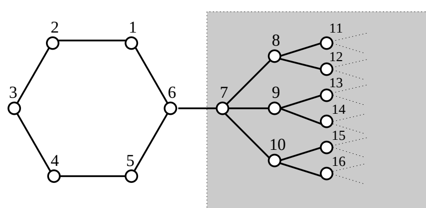


Fig. 3. An interference graph with a six cycle connected with a tree.

VI. Additional Theoretical Results

This section presents some theoretical results about σ-local pooling that can be useful for further research on this subject.

A. Optimality of Equal Weight Assignment for Set σ-Local Pooling in Special Classes of Graphs

In this section, we study the set σ-local pooling factor for some special classes of graphs. We use the word *schedule* to mean a *maximal* independent set of a graph.

Remark: In this section, unless mentioned other wise, all the graphs are understood as interference graphs or induced subgraphs of interference graphs for networks.

Given an interference graph (or induced subgraph) $G = (V, E)$, the set σ-local pooling factor is really a property of the graph itself. For clarity and simplicity, we will denote this factor by σ_G^* in this section (which should not be confused with $\sigma^*(G)$ introduced earlier). Let $w : V \rightarrow \mathbb{R}^+$ be a weight assignment on the nodes. For a schedule s, we use $\chi_w(s)$ to denote the total weight of the schedule s under w.

1) Weight-Balanced Graphs:

Definition 6: In a graph $G = (V, E)$, let w be an arbitrary weight assignment on the nodes. We say the graph is *weight-balanced*, if the following inequalities hold:

$$\bar{w}|M| \leq \chi_w(M) \tag{33}$$

$$\chi_w(m) \leq \bar{w}|m|, \tag{34}$$

where m and M are schedules with the minimum and maximum weight, respectively, and $\bar{w}$ is the average weight of all the nodes in G.

Lemma 21: A cycle graph is weight-balanced.

Proof: Let $G = (V, E)$ be a cycle graph. Let us index the nodes sequentially around the cycle from 1 to $|V|$. Given a schedule s, we let s^i denote the i^{th} rotation of s, for $0 \leq i \leq |V|-1$. That is, if we denote a schedule by the set of selected nodes, then $s^i = \{j | j = (k+i) \bmod |V|, \text{ where } k \in s\}$ (assuming we equate node 0 with node $|V|$). Note that $s^0 = s$.

Note that each rotation s^i of s is a schedule as well. Let $w = (w_1, w_2, \ldots, w_{|V|})$ be an arbitrary weight vector (weight assignment). By summing the weights of all these rotations, we get,

$$\sum_{i=0}^{|V|-1} \chi_w(s^i) = \sum_{i=1}^{|V|} w_i|s|. \tag{35}$$

Let M be a schedule with the maximum weight. It follows that $\chi_w(s^i) \leq \chi_w(M)$. Thus, we have the following.

$$\sum_{i=1}^{|V|} w_i|s| \leq |V|\chi_w(M)$$

$$\bar{w}|s| \leq \chi_w(M),$$

where $\bar{w}$ is the average node weight. This proves (33). The proof for (34) follows a similar argument. ■

We will consider computing the set σ-local pooling factor σ_G^* according to (29), which we re-write next using the new notations. Let M_G be the set of schedules for the graph G.

$$\sigma_G^* = \max_{x \geq 0} \frac{\min_{s \in M_G} \chi_x(s)}{\max_{s \in M_G} \chi_x(s)}. \tag{36}$$

The following lemma shows a special property if the graph G is weight-balanced.

Lemma 22: If the graph G is weight-balanced, then an equal weight assignment is optimal with respect to the optimization problem (36).

Proof: We consider two cases. First, consider the case where $\sigma_G^* = 1$. Suppose w is an optimal weight assignment. The inequalities (33) and (34) hold under w. Since $\sigma_G^* = 1$, all schedules have the same weight. Hence, if m and M are schedules with the minimum and maximum weight, respectively, we get $\chi_w(M) = \chi_w(m)$. Then, $\bar{w}|M| \leq \bar{w}|m|$ must hold. Using this inequality and the fact that any schedule can be considered as either a maximum-weight schedule or a minimum-weight schedule, we conclude that all schedules must have the same cardinality. Therefore, assigning equal weight to every node achieves $\sigma_G^* = 1$.

Consider the second case where $\sigma_G^* < 1$. Suppose there are n nodes in G and they are indexed from 1 to n. Take an arbitrary optimal weight assignment and suppose the average of the node weights is c, where c is some constant. For any weight vector $x = \{x_1, x_2, ..., x_n\}$, let $\phi(x) = \max_{i=1}^n x_i - \min_{i=1}^n x_i$, and let the average of x be denoted by $\bar{x}$.

Among all the optimal weight assignments with the average node weight equal to c, we pick one that minimizes ϕ and denote this assignment by x^*. That is, $x^* \in \arg\min\{\phi(x) \mid x \text{ is optimal and } \bar{x} = c\}$, with ties broken arbitrarily.

Notice that if $\phi(x^*) = 0$, the lemma holds. Next, we assume $\phi(x^*) > 0$ and we will show this leads to a contradiction. In particular, we will show it is always possible to construct a new optimal weight vector y with the same average node weight, c, such that $\phi(x^*) > \phi(y) > 0$.

We define $y = \{y_i | y_i = x_i^* - \epsilon(x_i^* - c)\}$. Note that, for any ϵ, $\bar{y} = \frac{1}{n}\sum y_i = \frac{1}{n}\sum x_i^* = c$. Also, there exists $\epsilon_1 > 0$ such that for all ϵ on $(0, \epsilon_1]$, we have $\phi(x^*) > \phi(y) > 0$. We will show that we can choose small enough ϵ on $(0, \epsilon_1]$ such that y is also an optimal weight assignment.

Since $\sigma_G^* < 1$, the maximum schedule weight is strictly greater than the minimum schedule weight under x^*. As a result, there is a gap between the maximum schedule weight and the second largest weight of all schedule weights. Furthermore, under the weight vector y, the weight of each schedule is a linear function of ϵ. Hence, if we choose small enough ϵ on $(0, \epsilon_1]$, we can make sure that some maximum-weight schedule under the weight assignment x^* remains a maximum-weight schedule under y. More precisely, there exists ϵ_2 on $(0, \epsilon_1]$ such that, for all $\epsilon \in (0, \epsilon_2]$, there exists a schedule M (independent of ϵ) that has the maximum weight under both x^* and y.

For a weight-balanced graph, we next show the maximum schedule weight is not increased when the weight assignment changes from x^* to y.

$$\begin{aligned}\chi_y(M) &= \sum_{i \in M} y_i \\ &= \sum_{i \in M} (x_i^* - \epsilon(x_i^* - c)) \\ &= \chi_{x^*}(M) - \epsilon(\chi_{x^*}(M) - |M|c).\end{aligned}$$

By the definition of a weight-balanced graph, $\chi_{x^*}(M) - |M|c \geq 0$. Thus, we conclude $\chi_y(M) \leq \chi_{x^*}(M)$.

Similarly, there exists ϵ_3 on $(0, \epsilon_1]$ such that, for all $\epsilon \in (0, \epsilon_3]$, there exists a schedule m (independent of ϵ) that has the minimum weight under both x^* and y; and furthermore, $\chi_y(m) \geq \chi_{x^*}(m)$. By choosing ϵ on $(0, \min(\epsilon_2, \epsilon_3)]$, we get $\sigma_G^* = \chi_{x^*}(m)/\chi_{x^*}(M) \leq \chi_y(m)/\chi_y(M)$. Hence, y is also an optimal weight assignment for the problem in (36). Considering $\phi(x^*) > \phi(y) > 0$ and the assumption that $x^* \in \arg\min\{\phi(x) \mid x \text{ is optimal and } \bar{x} = c\}$, we have reached a contradiction. ■

Lemma 22 implies that the inequality in Lemma 17 can be changed to equality in a weight-balanced graph. The next corollary follows immediately from Lemma 21 and 22.

Corollary 23: If the graph G is a cycle, then the optimal σ_G^* is achieved by assigning identical weights to the nodes.

2) Vertex-Transitive Graphs: We next consider another class of graphs for which an equal weight assignment is optimal for (36). We first need to introduce some definitions (see [16]). We consider undirected graphs with no loops and no more than one edge between any two different nodes, i.e., the *simple graphs*.

Definition 7: An isomorphism from a graph $G = (V_G, E_G)$ to a graph $H = (V_H, E_H)$ is a bijection $f : V_G \to V_H$ such that $(u, v) \in E_G$ if and only if $(f(u), f(v)) \in E_H$.

Definition 8: An automorphism of a graph G is an isomorphism from G to G.

Definition 9: A graph $G = (V, E)$ is vertex-transitive if for every pair $u, v \in V$ there is an automorphism that maps u to v.

Lemma 24: If a graph G is vertex-transitive, then the optimization problem (36) is achieved by an equal weight assignment on the nodes.

Proof: As in the proof of Lemma 22, we assume the nodes in G are indexed from 1 to n; for any weight vector x, we define $\phi(x) = \max_{i=1}^n x_i - \min_{i=1}^n x_i$. Note that $\phi(x) = 0$ if and only if the node weights are all identical. Let $\bar{x}$ denote the average node weight.

We restrict our attention to those weight assignments whose average node weight is equal to c, where c is some constant, e.g., $c = 1$. This "normalization" is without loss of generality. Among all the optimal weight assignments that have the average node weight equal to c, we pick one that minimizes ϕ and denote this assignment by x^*. That is, $x^* \in \arg\min\{\phi(x) \mid x \text{ is optimal and } \bar{x} = c\}$. Suppose $\phi(x^*) > 0$.

Let $a = \min_i x_i^*$ and $b = \max_i x_i^*$. Let $S_a = \{i | x_i^* = a\}$ and $S_b = \{i | x_i^* = b\}$. We observe that $S_a, S_b \neq \emptyset$ and $\phi(x^*) = b - a > 0$. Now, we pick a node $u \in S_b$ and a node $v \in S_a$. By the definition of vertex-transitivity, there is an automorphism f that maps u to v. We construct a new weight vector x' by $x'_i = 1/2(x_i^* + x_{f(i)}^*)$ for all i. Note that the weight vector y with $y_i = x_{f(i)}^*$ for all i is also optimal. Since x' is a convex combination of two optimal solutions, x' is also optimal. Also, the new weight vector x' has the same average weight as x^*. Notice that $\max_i x'_i \leq b$, $\min_i x'_i \geq a$ and $\phi(x') \leq \phi(x)$. Since $x_u^* > x_v^*$, it follows that $x'_v > x_v^* = a$. Hence, the set of nodes whose weight is equal to a has fewer elements under x', and we again call this set S_a. If S_a is empty, then $\phi(x') < \phi(x^*)$, which is a contradiction. For a similar reason, the set of nodes whose weight is equal to b has fewer elements under x' and we call this set S_b. If S_b is empty, then $\phi(x') < \phi(x^*)$, which is a contradiction.

If neither S_a nor S_b is empty, we can repeat the above procedure to construct a new weight vector from x' and update S_a and S_b. Eventually, either S_a or S_b first becomes empty under some new weight vector, say $\hat{x}$. Then $\phi(\hat{x}) < \phi(x^*)$, which is a contradiction. ■

Remark: A similar result was found independently in [14] [17]. It is easy to show that cycles are vertex-transitive. Therefore, Lemma 24 also implies Corollary 23.

B. Graphs with Arbitrarily Small Set σ-Local Pooling Factors

We will prove that set σ-local pooling factors can be arbitrarily small using a special class of graphs, the hypercubes. A similar result was found independently in [14] [17] using a different class of graphs. The hypercube graph Q_n is a regular graph with 2^n vertices. The hypercube graph Q_n can be constructed by labeling the 2^n vertices $0, 1, 2, \ldots, 2^n - 1$ and connecting two vertices whenever the Hamming distance of the binary representations of the labels is equal to 1. The vertex set of Q_n is denoted by V_{Q_n}.

Lemma 25: A maximum cardinality independent set of Q_n has at least 2^{n-1} elements.

Proof: We will inductively construct an independent set of the desired cardinality. The induction hypothesis is: Q_n has a maximal independent set, $Z \subseteq V_{Q_n}$, such that $V_{Q_n} - Z$ is a maximal independent set as well. For $n = 1$, the statement holds (for $n = 0$, the statement would hold with proper definitions). It is known that Q_{n+1} can be constructed by using two hypercubes Q_n and connecting corresponding vertices together. Take Q_n^1 and Q_n^2, two copies of Q_n, to construct Q_{n+1}. We will show Q_{n+1} has a maximal independent set Z, whose complement, $V_{Q_{n+1}} - Z$, is a maximal independent set as well. By the induction hypothesis, Q_n^1 can be partitioned into two maximal independent sets , Z^1 and $V_{Q_n^1} - Z^1$. Let Z^2 be the maximal independent set of Q_n^2 corresponding to Z^1. Now, observe $Z = Z^1 \cup (V_{Q_n^2} - Z^2)$ is a maximal independent set and its complement $V_{Q_{n+1}} - Z = V_{Q_{n+1}} - (Z^1 \cup (V_{Q_n^2} - Z^2)) = Z^2 \cup (V_{Q_n^1} - Z^1)$ is a maximal independent set as well. Thus the result follows.

We conclude that Q_n can be partitioned into two maximal independent sets and one of them must have at least 2^{n-1} elements. ■

Lemma 26: A minimum cardinality maximal independent set of Q_n has at most $2^n/(n+1)$ elements when $n = 2^k - 1$, where $k \in \mathbb{Z}^+$.

Proof: In [18], it is shown whenever n is in the form $n = 2^k - 1$ for some positive integer k, Q_n has a perfect dominating set. In perfect domination, every vertex can only be dominated by a single node. Since every node in Q_n has degree n, there are at most $2^n/(n+1)$ elements in a perfect dominating set. Note that an independent dominating set is a maximal independent set. ■

Lemma 27: Let $a_n = 2^n - 1$, then $\sigma^*_{Q_{a_n}} \leq 2^{1-n}$.

Proof: A hypercube is vertex-transitive. By Lemma 24, $\sigma^*_{Q_{a_n}}$ can be derived under an equal weight assignment on the vertices. Let $M_{Q_{a_n}}$ be set of all maximal independent sets of Q_{a_n}. By Lemma 25, $\max_{m \in M_{Q_{a_n}}} |m| \geq 2^{a_n - 1}$. By Lemma 26, $\min_{m \in M_{Q_{a_n}}} |m| \leq 2^{a_n}/(a_n + 1)$. Since

$$\sigma^*_{Q_{a_n}} = \frac{\min_{m \in M_{Q_{a_n}}} |m|}{\max_{m \in M_{Q_{a_n}}} |m|} \leq \frac{2^{a_n}/(a_n+1)}{2^{a_n - 1}} = 2^{1-n},$$

the lemma holds. ■

Corollary 28: Let $a_n = 2^n - 1$. Then, $\lim_{n \to \infty} \sigma^*_{Q_{a_n}} = 0$.

C. Computational Complexity of Calculating σ^*_G

Again, consider the interference graph (or subgraph) $G = (V, E)$. By Lemma 5, the set σ-local pooling factor, σ^*_G, is the optimal value of the following optimization problem.

$$\begin{aligned} \max \quad & w && (37) \\ \text{subject to} \quad & x' M_G \leq e' && (38) \\ & x' M_G \geq w e' && (39) \\ & x, w \geq 0. && (40) \end{aligned}$$

Here, M_G is a matrix where the columns are all the maximal independent sets of G, and $e = (1, 1, \ldots, 1)'$ of an appropriate dimension. We will investigate the computational complexity of calculating σ^*_G by solving the above optimization problem.

A separation oracle for the above linear program is a procedure to test whether a given vector is in the convex region defined by the constraints (38) - (40), and if not, find a violating constraint. Let SEPORC(G) denote such a separation oracle. We will first show SEPORC(G) is NP-hard by giving a reduction from the minimum cardinality maximal independent set problem to SEPORC(G).

Lemma 29: SEPORC(G) is NP-hard.

Proof: The decision version of the minimum cardinality maximal independent set (MCMIS, also known as the minimum independent dominating set) problem asks whether there is a maximal independent set of size K or less, where $1 \leq K \leq |V|$. It is known that this problem is NP-complete (page 190, [19]). We will provide a Turing reduction from the MCMIS problem to SEPORC(G). Given a vector $x \in \mathbb{Q}^{|V|}$ and $w \in \mathbb{Q}$, SEPORC(G) can decide the membership of (x, w) in the convex region defined by the constraints (38) - (40). Using this separation oracle, we will create a solver for the MCMIS problem (thus providing a Turing reduction). We set $x_i = \frac{1}{|V|}$ for $i = 1, \ldots, |V|$ and set $w = (K+1)/|V|$. The vector (x, w) always satisfies constraint (38). Hence, feeding (x, w) to SEPORC(G) will tell us whether (x, w) satisfies constraint (39) or not. If yes, an MCMIS is at least of the size $|V|w = K + 1$; if no, the MCMIS has less than or equal to K elements. ■

Remark: In the proof, only the feasibility aspect of the separation oracle is used. The proof really says the feasibility problem is NP-hard. But, SEPORC(G) is at least as hard as the feasibility problem.

Lemma 30: Computing σ^*_G by solving the optimization problem (37) - (40) is NP-hard.

Proof: In [20], the authors have established that the complexity of the separation oracle and that of the original optimization problem are polynomially equivalent. Combine this fact with Lemma 29. ■

Remark: The statement of Lemma 30 should not be understood as "finding σ^*_G is NP-hard". It means that solving the optimization problem (37) - (40) is NP-hard, which involves finding both an optimal solution and the optimal value.

D. Special Structures That Preserve $\sigma^*(G)$

In a graph, we will call a node r a *super-node* if it is connected to all other nodes. A super-node in an interference graph corresponds to a link in the network that interferes with all other links. Suppose we start with an interference graph $G = (V, E)$. We will investigate the effect of inserting a super-node to G. We will show that the local pooling factor (for graphs) is not altered by the insertion. We denote the graph after the insertion of the super-node by $G' = (V', E')$. With slight abuse of notation, we denote the graph local pooling factors by $\sigma^*(G)$ and $\sigma^*(G')$ for G and G', respectively (instead of the notation that uses the network graph). Given a subset $T \subseteq V$, we denote the set σ-local pooling factor corresponding T by σ_T^*.

Lemma 31: $\sigma^*(G) = \sigma^*(G')$.

Proof: We will consider computing the set σ-local pooling factor σ_T^*, $T \subseteq V$, according to (29), which we re-write next using some new notations. Let M_T be the set of maximum independent sets for the interference subgraph associated with T. Given a weight assignment x on the nodes and a set of nodes s and, let $\chi_x(s)$ be the total weight of the set s. We know that

$$\sigma_T^* = \max_{x \geq 0} \frac{\min_{s \in M_T} \chi_x(s)}{\max_{s \in M_T} \chi_x(s)}. \quad (41)$$

Also note (by combining Lemma 1 and Lemma 6)

$$\sigma^*(G) = \min_{T \subseteq V} \sigma_T^*. \quad (42)$$

Let the super-node be denoted by r. Then, $V' = V \cup r$. Consider the case where $T' \subseteq V'$ and $r \in T'$. Let $T = T' - \{r\}$. We will show $\sigma_T^* \leq \sigma_{T'}^*$. Let x be an optimal weight assignment on the nodes in T that achieves σ_T^*. Let a be an arbitrary value on $[\min_{s \in M_T} \chi_x(s), \max_{s \in M_T} \chi_x(s)]$, and assign the weight a to node r. The vector $y = (x', a)'$ is a weight assignment on the nodes in T'. Next, note that $M_{T'} = M_T \cup \{s\}$. Hence,

$$\min_{s \in M_{T'}} \chi_y(s) = \min(\min_{s \in M_T} \chi_x(s), a) = \min_{s \in M_T} \chi_x(s),$$

and

$$\max_{s \in M_{T'}} \chi_y(s) = \max(\max_{s \in M_T} \chi_x(s), a) = \max_{s \in M_T} \chi_x(s).$$

As a result,

$$\sigma_{T'}^* \geq \frac{\min_{s \in M_{T'}} \chi_y(s)}{\max_{s \in M_{T'}} \chi_y(s)} = \frac{\min_{s \in M_T} \chi_x(s)}{\max_{s \in M_T} \chi_x(s)} = \sigma_T^*. \quad (43)$$

Next,

$$\begin{aligned} \sigma^*(G') &= \min_{T' \subseteq V'} \sigma_{T'}^* \\ &= \min(\min_{T' \subseteq V', r \in T'} \sigma_{T'}^*, \min_{T' \subseteq V', r \notin T'} \sigma_{T'}^*) \\ &= \min(\min_{T' \subseteq V', r \in T'} \sigma_{T'}^*, \sigma^*(G)). \end{aligned} \quad (44)$$

By (43) and the one-to-one correspondence between T' and T through $T' = T \cup \{r\}$, where $T' \subseteq V'$, $r \in T'$ and $T \subseteq V$, we get

$$\min_{T' \subseteq V', r \in T'} \sigma_{T'}^* \geq \min_{T \subseteq V} \sigma_T^* = \sigma^*(G). \quad (45)$$

Combining (44) and (45), we have

$$\sigma^*(G') = \sigma^*(G).$$

■

VII. Additional Related Work

In this section, we cover some additional related work. The references cited by [1] are mostly related, but are not all repeated here. In [2], Lin et al. provided a distributed algorithm using schedules that correspond to *maximal* matchings (in the interference graph) for the 1-hop interference model. In each such schedule, no more links can be added to it without violating the interference constraint. They showed that the algorithm can achieve a stability region $\Lambda/2$ under the 1-hop model. For more general interference models, they showed that if one can find an approximation algorithm with the approximation ratio γ for a maximum weighted independent set subproblem, then one can achieve a stability region $\gamma\Lambda$.

In [3], the authors considered the 2-hop interference model that can successfully capture the IEEE 802.11 network and provided an algorithm to find an upper-bound for the network capacity. In [4] [5], the 2-hop interference is generalized to k-hop interference and the problem of finding a maximum-weight schedule was shown to be NP-Hard, for $k \geq 2$. The authors of [21] [7] provided lower bounds on the performance of the *maximal matching* algorithms for the cases of arbitrary and geometric graphs. More specifically, for geometric graphs, [21] showed that the efficiency ratio $\gamma^*(G) \geq 1/8$ in the 2-hop interference model; [5] showed $\gamma^*(G) \geq 1/49$ in the k-hop interference model where $k \geq 2$.

In addition to [13], several other papers also introduced algorithms of local scheduling that have performance guarantee. Lin and Rasool [22] introduced random scheduling schemes under the 1-hop and 2-hop interference models. There is a slight loss of efficiency in their schemes compared with what is achievable by the so-called distributed greedy scheduling algorithm by Wu et al. [23], which is also a local greedy algorithm. Joo and Shroff [24] and Gupta et al. [25] both discovered related random scheduling schemes that improve upon Lin and Rasool's schemes.

VIII. Conclusion

In this paper, we provide a refined framework on performance characterization of the LQF policy, based on the idea of local pooling introduced in [8] [1]. In particular, we introduce the concept of link σ-local pooling, which allows heterogeneous characterization of individual link performance, as opposed to treating all links the same. We define the $\Sigma^*(G)$ diagonal matrix, which contains the link σ-local pooling factors in the diagonal entries, as a generalization of the network σ-local pooling, $\sigma^*(G)$, in [1]. The matrix $\Sigma^*(G)$ provides a refined performance characterization for LQF. We show that our performance characterization captures a larger region of stability than previous results.

We then introduce a set of theory that helps to apply the new idea of link σ-local pooling. The core of this theory involves the concepts of σ-local pooling for a set of links and the limiting set for a link. We show how these concepts are related to link σ-local pooling, and how to calculate or bound both set and link σ-local pooling factors. Based on the developed theory, we derive new estimation methods for set and link σ-local pooling factors.

There are still open issues that may be addressed by further research. The following are three examples. First, the computational complexity of calculating $\Sigma^*(G)$ or σ_L^* is still unknown. Second, in light of the newly discovered fact that Joo's local greedy scheduling also achieves the stability region $\sigma^*(G)\Lambda^o$ [13], it would be interesting to investigate whether the enlarged stability region of LQF, $\Sigma^*(G)\Lambda^o$, can be preserved by similar local greedy algorithms. Third, there are other, possibly nonlinear, transformations of the capacity region Λ to $\phi(\Lambda)$ where LQF stabilizes the network. Further investigation on these transformations may have important theoretical and practical values.

References

[1] C. Joo, X. Lin, and N. B. Shroff, "Understanding the capacity region of the greedy maximal scheduling algorithm in multi-hop wireless networks," in *Proceedings of IEEE INFOCOM*, 2008.

[2] X. Lin, N. B. Shroff, and R. Srikant, "The impact of imperfect scheduling on cross-layer rate control in wireless networks," *IEEE/ACM Transaction on Networking*, vol. 14, no. 2, pp. 302–315, April 2006.

[3] H. Balakrishnan, C. Barrett, V. Kumar, M. Marathe, and S. Thite, "The distance-2 matching problem and its relationship to the MAC-layer capacity of ad hoc networks," *IEEE Journal On Selected Areas In Communications*, vol. 22, no. 6, pp. 1069–1079, 2004.

[4] G. Sharma, N. B. Shroff, and R. R. Mazumdar, "Maximum weighted matching with interference constraints," in *PERCOMW '06: Proceedings of the 4th annual IEEE international conference on Pervasive Computing and Communications Workshops*, 2006.

[5] G. Sharma, R. R. Mazumdar, and N. B. Shroff, "On the complexity of scheduling in wireless networks," in *Proccedings of ACM MobiCom*, 2006, pp. 227–238.

[6] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936–1948, Dec 1992.

[7] G. Sharma, N. B. Shroff, and R. R. Mazumdar, "Joint congestion control and distributed scheduling for throughput guarantees in wireless networks," in *Proceedings of IEEE INFOCOM*, 2007, pp. 2072–2080.

[8] A. Dimakis and J. Walrand, "Sufficient conditions for stability of longest-queue-first scheduling: Second-order properties using fluid limits," *Advances in Applied Probability*, vol. 38, pp. 505–521, 2006.

[9] A. Brzezinski, G. Zussman, and E. Modiano, "Local pooling conditions for joint routing and scheduling," in *Information Theory and Applications Workshop*, 2008, pp. 499–506.

[10] ——, "Enabling distributed throughput maximization in wireless mesh networks: a partitioning approach," in *Proceedings of ACM MobiCom*, 2006, pp. 26–37.

[11] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, 2000.

[12] J. G. Dai, "On positive harris recurrence of multiclass queueing networks: A unified approach via fluid limit models," *Annals of Applied Probability*, vol. 5, pp. 49–77, 1995.

[13] C. Joo, "A local greedy scheduling scheme with provable performance guarantee," in *Proceedings of MobiHoc*, 2008, pp. 111–120.

[14] B. Birand, M. Chudnovsky, B. Ries, P. Seymour, G. Zussman, and Y. Zwols, "Analyzing the performance of greedy maximal scheduling via local pooling and graph theory," in *Proceedings of IEEE INFOCOM*, March 2010.

[15] M. Leconte, J. Ni, and R. Srikant, "Improved bounds on the throughput efficiency of greedy maximal scheduling in wireless networks," in *Proceedings of MobiHoc*, 2009.

[16] D. West, *Introduction to Graph Theory*. Prentice Hall, 2001.

[17] B. Birand, M. Chudnovsky, B. Ries, P. Seymour, G. Zussman, and Y. Zwols, "Analyzing the performance of greedy maximal scheduling via local pooling and graph theory," Columbia University, Tech. Rep., July 2009.

[18] M. Livingston and Q. F. Stout, "Perfect dominating sets," *Congressus Numerantium*, vol. 79, pp. 187–203, 1990.

[19] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, USA: W. H. Freeman & Co., 1979.

[20] M. Grötschel, L. Lovàsz, and A. Schrijver, "The ellipsoid method and its consequences in combinatorial optimization," *Combinatorica*, vol. 1, pp. 169–197, 1981.

[21] P. Chaporkar, K. Kar, and S. Sarkar, "Throughput guarantees through maximal scheduling in wireless networks," in *Proceedings of 43d Annual Allerton Conference on Communication, Control and Computing*, 2005, pp. 28–30.

[22] X. Lin and S. B. Rasool, "Constant-time distributed scheduling policies for ad hoc wireless networks," in *Proceedings of the IEEE CDC*, 2006.

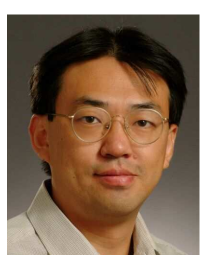
[23] X. Wu, R. Srikant, and J. R. Perkings, "Scheduling efficiency of distributed greedy scheduling algorithms in wireless networks," in *Proceedings of IEEE INFOCOM*, 2006.

[24] C. Joo and N. Shroff, "Performance of random access scheduling schemes in multi-hop wireless networks," in *Proceedings of IEEE INFOCOM*, 2007.

[25] A. Gupta, X. Lin, and R. Srikant, "Low-complexity distributed scheduling algorithms for wireless networks," in *Proceedings of IEEE INFOCOM*, May 2007.

Bo Li is a PhD student at the Computer and Information Science and Engineering department at the University of Florida, starting in August 2006. He has an MS degree and a BA degree from the Huazhong University of Science and Technology, China, in 2006 and 2004, respectively. His research interests are in the computer networking area, including link scheduling for wireless networks and network resource allocation. He is also interested in probability theory, stochastic processes and graph theory.

Cem Boyaci is a PhD student at the Computer and Information Science and Engineering department at the University of Florida. He has a BS degree from Bilkent University, Ankara, Turkey in 2005. His current research interests include complexity and performance characterization of scheduling algorithms in wireless computer networks.



Ye Xia is an associate professor at the Computer and Information Science and Engineering department at the University of Florida, starting in August 2003. He has a PhD degree from the University of California, Berkeley, in 2003, an MS degree in 1995 from Columbia University, and a BA degree in 1993 from Harvard University, all in Electrical Engineering. Between June 1994 and August 1996, he was a member of the technical staff at Bell Laboratories, Lucent Technologies in New Jersey. His research interests are in the computer networking area, including performance evaluation of network protocols and algorithms, wireless network link scheduling, network resource allocation, and content distribution algorithms on overlay networks. He is also interested in probability theory, stochastic processes and queueing theory.