Scheduling for Minimum Power on a Gaussian MAC

Arun Padakandla and Rajesh Sundaresan

Abstract

Users with rate requirements are to be scheduled in a time-slotted Gaussian multiple-access channel (GMAC). The number of users exceeds the number of slots available. For receivers employing a successive cancelation decoding, the problem of identifying a schedule for minimum sum power reduces to a combinatorial optimization problem related to the MULTIPROCESSOR SCHEDULING problem (also called MAKE SPAN) that attempts to make the slot-sums close to each other, in a sense specified in the paper. The computational complexity of the decision versions of this problem are addressed. When the rates are integers and the number of slots per unit time is fixed up front, an algorithm with a polynomial time complexity that terminates in $O\left(\text{Length}(I)^{N+1}\right)$ steps is provided. However, if the number of slots per unit time is a variable, the problem is shown to be NP-complete. When the rates are rational numbers, a related sum-product minimization problem is proposed and the complexities of their decision versions studied. When the number of slots per unit time is fixed, the problem is NP-complete. When the number of slots is a variable, the problem is strongly NP-complete. Performance bounds for the well-known largest processing time (LPT) heuristic are also provided.

I. Introduction and Prior Work

Given positive integers $r_1, r_2, \cdots, r_K$, number of slots N, with $N \leq K$, we consider the problem of placing the numbers into N slots so that the slot-sums are as close to each other as possible. Let $S_1, S_2, \cdots, S_N$ be a partition of $\{1, 2, \cdots, K\}$ indicating the indices of numbers in slots $1, 2, \cdots, N$, respectively. Let

$$r(S_n) = \sum_{k \in S_n} r_k$$

be the sum in the nth slot. The MULTIPROCESSOR SCHEDULING problem (also called MAKE SPAN) [13, SS8] views the N slots as processors, requires minimization of the maximum of the slot-sums, $\max\{r(S_n), n = 1, 2, \cdots, N\}$, and is NP-hard. Since the overall sum across all slots is a constant, another well-studied problem is to minimize the L_p metric for $p > 1$. (Chandra & Wong [1]). In this paper we first show how a related problem of minimizing the exponential cost

$$\sum_{n=1}^{N} 2^{r(S_n)}$$

arises in a simple multiple-access communication setting, provide several results on the complexity of the problem, and analyze a simple heuristic known as Graham's largest processing time (LPT) rule [2].

A. Padakandla is with the University of Michigan, Ann Arbor.

R. Sundaresan is with the Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore 560012, India.

This work was supported by the Department of Science and Technology, and by the University Grants Commission.

Consider a Gaussian multiple-access channel (GMAC) with K users. User k demands reliable communication at rate $\frac{r_k}{2}$ bits per second[1]. There are N slots every second[2] and each user transmits in at most one slot per second. We consider an overloaded system where $K \geq N$. Let S_n denote the set of users that transmit in slot $n, 1 \leq n \leq N$. The received signal in slot n is given by

$$Y_n = \sum_{k \in S_n} X_k + W_n$$

where X_k is the information symbol transmitted by user k. The additive noise random variables W_n are independent and identically distributed as $\mathcal{N}(0,1)$. The goal is to schedule users in each of these n slots so that users' rate requirements are met and sum power over all users is minimized.

The above problem, where only a subset of users are scheduled in a slot, can be motivated as follows. It is well-known that if the goal is only to minimize sum power, then all users should use all slots [3, Lemma 3.4]. Such a consideration leads to the classical multiple-access channel with K users accessing the channel in a slot. Optimal decoding however requires the receiver to do a joint decoding across all K users; the decoding complexity is exponential in K. Moreover, such an access scheme is prone to jamming as a jammer can affect all users' coded signals. On the other hand, one could schedule at most one user per slot. This significantly simplifies the multiple-access coding problem, but is power inefficient. Moreover, each user has to wait K slots before getting an opportunity to transmit; this may not meet a user's delay constraint. A trade-off is to schedule the K users in N slots, $N < K$, where each user transmits in at most one of these N slots. N is small enough to meet the delay constraint, yet large enough to provide jamming resilience. We assume throughout that N is obtained as per system and delay requirements. It may either be fixed up front or may be supplied as part of the optimization problem. Since $N < K$, there is at least one slot with two or more users. We are therefore studying an uplink analog of multipacket downlink transmission of low data rate packets used in 1xEV-DO Rev A [4], where several voice packets are grouped together in a time-code slot to meet voice application's tight delay constraints.

Clearly, this problem can be posed in other settings as well. For example in a frequency-flat channel, subcarriers pertaining to an OFDM system and codes pertaining to CDMA system play the role of time slots in this paper. Our attention to scheduling in time slots is only to ease exposition.

Let us first focus on one slot, say n. Recall that S_n is the subset of users that transmit in slot n. Users in this slot use up one degree of freedom and do not interfere with users in other slots. In order to meet the rate requirements, the sum power of users in this slot should satisfy [5, Section 14.1.2]

$$\frac{1}{2} r(S_n) := \frac{1}{2} \sum_{k \in S_n} r_k \leq \frac{1}{2} \log \left(1 + \sum_{k \in S_n} p_k \right)$$

so that[3]

$$\sum_{k \in S_n} p_k \geq 2^{r(S_n)} - 1. \tag{1}$$

Furthermore, it is known that the above lower bound on sum power is achieved via a successive cancelation decoder. (See for e.g. [3, Lemma 3.4]). Thus we may assume equality in (1) for a fixed S_n.

[1]Reason for the appearance of the factor $\frac{1}{2}$ will soon be clear.

[2]Second as unit of time has been chosen for concreteness. In practice, this unit of scheduling time is of the order of milliseconds in cellular systems (3GPP HSUPA).

[3]Note the disappearance of the factor 2 in the exponent; this is the reason for the rather strange appearance of $\frac{1}{2}$ in the rate requirement.

Given a partition $S_n : n \in [N]$, where $[N]$ denotes the set $\{1, 2, \cdots, N\}$, the minimum sum power for the partition is given by

$$\sum_{n=1}^{N} \sum_{k \in S_n} p_k = \sum_{n=1}^{N} 2^{r(S_n)} - N \tag{2}$$

The minimum is over all encoding and decoding schemes for the given partition. We now pose the following question: What is the minimum power (2) over all partitions?

Observe that if it were possible to make all partition sums $r(S_n)$ the same, the objective function in (2) is minimized. Of course such a partition may not exist. With this ideal as a benchmark, the ideal partition sum is then $A := N^{-1} \sum_{k=1}^{K} r_k$, and $r(S_n) - A$ is a measure of deviation from the best value in slot n. The objective function in (2) attempts to minimize an exponential cost of these deviations, with positive deviations more heavily penalized than negative deviations.

In this paper, we address the complexity of the decision versions of this problem. In Section III-A, we show that when N is input as part of the problem instance, the problem is NP-complete, i.e., if this problem can be solved in polynomial time by a deterministic Turing machine, we will have obtained a polynomial time algorithm to several problems that have thus far resisted such solutions. In Section III-B we show that a version of the problem where N is fixed can be solved in polynomial time by a deterministic Turing machine. This result is some what surprising because it is at variance with the corresponding results for MULTIPROCESSOR SCHEDULING or for the optimization of L_p metrics.

When the scaled rates r_k are rational numbers, computation of the possibly irrational $2^{r(S_n)}$, cannot in general be done in finite time with infinite precision. It is then natural to look at approximations to $2^{r(S_n)}$. To move the approximation out of the computation framework, we assume that users supply a rational approximation of $2^{r_k} \approxeq \frac{f_k}{g_k}$. The above combinatorial optimization problem then takes the form: Given $\frac{f_k}{g_k} : k \in [K]$, rational approximations to exponentiated rates, minimize

$$\sum_{n=1}^{N} \prod_{k \in S_n} \frac{f_k}{g_k} - N$$

over all partitions $S_n : n \in [N]$ of $[K]$. In Section IV, as in Section III we study the decision versions of two variants of this problem. When N is a variable and input as part of the problem instance we prove the problem is strongly NP-complete. When N is assumed known and fixed we prove the problem is NP-complete.

The intractability of the above scheduling problems leads us to study an approximation algorithm. In section V, we analyze the LPT algorithm and prove worst case upper bounds for the power of the partition output by LPT, measured in the logarithmic scale. Our worst case upper bound is a function of the average offered input load per slot.

Algorithms for power allocation to nodes in a network have been studied in different contexts. In a wireless communication setting where a node's transmission range being proportional to its transmit power, the topology of the network is a function of the assigned powers. Chen and Huang [6] show the problem of minimizing sum power for full connectivity is NP-hard. Kirousis and others [7] study a simpler problem when n nodes lie along a line separated by unit distance and provide an $O(n^4)$ algorithm for power assignment. The above problems deal with power allocation for connectivity, whereas we focus on power allocation to meet certain rate requirements. Arikan [8] considers the problem of assigning powers to nodes in a packet radio network such that specific origin-destination pairs communicate at specified rates. He proves that the problem is NP-hard. Assuming that the only cause for packets to be received in error is simultaneous transmissions he schedules at most one radio at any time. In contrast we allow for multiple users per slot and compensate for simultaneous transmissions via larger

powers and the use of a successive cancelation decoder. Tassiulas [9] proposes randomized algorithms for the dual problem of scheduling for maximum throughput in a wireless network. A summary of several recent works on scheduling for maximum throughput on a wireless network is the monograph by Georgiadis et al. [10]. A problem analogous to (2) but with L_p metric as cost was studied by Chandra & Wong [1], Leung & Wei [11], and Goldberg & Shapiro [12]. Particularly, Graham's LPT rule [2] was analyzed and bounds from optimality were provided.

In Section II, we introduce some notation and relevant concepts from complexity theory. In sections III and IV, we prove the intractability of the scheduling problems. In section V we analyze the LPT algorithm. We conclude with some remarks in Section VI.

II. Preliminaries

We begin with some remarks on notation. Recall that for an integer $K \geq 1$, $[K]$ denotes the set $\{1, 2 \cdots, K\}$. For $x, y \in \mathbb{Z}_+$, let $s(x) = \lfloor \log_2 x \rfloor + 1$ represent span of x when represented in binary. Let $p_S(x)$, $p_M(s(x), s(y))$, $p_C(s(x), s(y))$ be the number of steps required to compute 2^x, compute xy, and compare x and y respectively, where $p_s(\cdot)$, $p_M(\cdot, \cdot)$, and $p_C(\cdot, \cdot)$ are fixed univariate and bivariate polynomials.

We next collect here some well known facts for ease of reference.

- Given $x \in \mathbb{Z}_+$, x^2 can be computed in time polynomial in $s(x)$.
- If x is a perfect square, $\sqrt{x}$ can be computed in time polynomial in $s(x)$. Indeed, a binary search performed on the ordered set $[x]$ will require computing squares of $s(x)$ numbers each of value at most x, followed by a comparison with x. This can be done in $O\left(s\left(x\right)\left(p_M\left(s(x), s(x)\right) + p_C\left(s(x), s(x)\right)\right)\right)$ steps, thus polynomial in $s(x)$.
- Given positive integers $x_k : k \in [K]$, $\prod_{k=1}^{K} x_k$ can be computed in
$$\sum_{l=1}^{K-1} p_M\left(\sum_{i=1}^{l} s\left(x_i\right), s\left(x_{l+1}\right)\right) \leq K p_M\left(K s_{\max}, s_{\max}\right)$$
steps, where $s_{\max} := \max_{k \in [K]} s(x_k)$. Since $K \leq \sum_{k=1}^{K} s\left(x_k\right)$, the right hand side above is a polynomial in the input length. Thus $\prod_{k=1}^{K} x_k$ can be computed in time polynomial in the input length. Since $s\left(\prod_{k=1}^{K} x_k\right) \leq K s_{\max}$, if $\prod_{k=1}^{K} x_k$ is a perfect square, $\sqrt{\prod_{k=1}^{K} x_k}$ can be computed in time polynomial in the input length.

For a problem Π, let domain $D(\Pi)$ denote the set of all valid instances of Π, and $Y\left(\Pi\right)$ the set of all yes-instances of Π. Let $\max_\Pi : D(\Pi) \to \mathbb{Z}_+$ map a valid instance I to the magnitude of the largest integer in I, or 0 if no integer occurs in I. Let $\text{Length}_\Pi : D(\Pi) \to \mathbb{Z}_+$ map a valid instance I to the length of its encoding[4]. The subscript Π is omitted when the problem under consideration is clear from the context.

A problem Π' is a subproblem of Π if $D(\Pi') \subseteq D(\Pi)$ and $Y(\Pi') = D(\Pi') \cap Y(\Pi)$. Note that a problem Π and a restricted domain $D(\Pi') \subseteq D(\Pi)$ define the subproblem Π' uniquely.

Let $p(\cdot)$ be a polynomial. Π_p is a subproblem of Π defined through its domain

$$D(\Pi_p) = \left\{I \in D(\Pi) : \max\nolimits_\Pi(I) \leq p(\text{Length}_\Pi(I))\right\}.$$

[4]Any encoding scheme referred to in this paper is a *reasonable* encoding scheme. For a discussion on *reasonable* encoding schemes refer to [13, Section 2.1]

We quickly recall some basic complexity concepts. See [13, Chapter 2] for a detailed discussion. Π is said to be in class P if it can be solved by a deterministic Turing machine in polynomial time. Π is said to be in class NP if it can be solved by a non-deterministic Turing machine in polynomial time. We say problem Λ can be reduced to Π in polynomial time if there exists an $f : D(\Lambda) \rightarrow D(\Pi)$ that satisfies the following :

1) for all $I \in D(\Lambda)$, $I \in Y(\Lambda) \Leftrightarrow f(I) \in Y(\Pi)$, and
2) given I, $f(I)$ can be computed in time polynomial in $\text{Length}_{\Lambda}(I)$.

Π is NP-complete if $\Pi \in$ NP and every problem $\Lambda \in$ NP can be reduced to Π in polynomial time.

Definition 1: [13, p.95] A problem Π is strongly NP-complete if there exists a polynomial $p(\cdot)$ such that Π_p is NP-complete.

Example 2: Consider the following three dimensional matching (3DM) problem.

3DM : Given disjoint sets X, Y, Z and a set $V \subseteq X \times Y \times Z$, is there $V' \subseteq V$ that forms a matching for X, Y, Z ? In other words, does every element of X, Y, Z belong to exactly one triplet in the matching V'? □

3DM is NP-complete. It is also strongly NP-complete because no integer occurs in its description.

The following definition and lemma relate difficulty levels of two problems.

Definition 3: [13, p.101] For two problems Π and Λ, $f : D(\Pi) \rightarrow D(\Lambda)$ is a *pseudo polynomial transformation* (PPTM) if the following hold:

(i) For all $I \in D(\Pi)$, we have $I \in Y(\Pi)$ if and only if $f(I) \in Y(\Lambda)$;
(ii) f can be computed in time polynomial in the two variables $\max_{\Pi}(I)$ and $\text{Length}_{\Pi}(I)$;
(iii) there exists a polynomial q_1 such that if $I \in D(\Pi)$, then $q_1(\text{Length}_{\Lambda}(f(I))) \geq \text{Length}_{\Pi}(I)$; and
(iv) there exists a two-variable polynomial q_2 such that for all $I \in D(\Pi)$ we have $\max_{\Lambda}(f(I)) \leq q_2(\max_{\Pi}(I), \text{Length}_{\Pi}(I))$.

Lemma 4: Let Π be a strongly NP-complete problem and Λ another problem. If f is a PPTM from Π to Λ, then Λ is strongly NP-complete.

See [13, Lemma 4.1] for a proof.

III. Slotted allocation for power minimization

Recall from Section I that the problem of minimizing total received sum power (2) needed to satisfy a set of rate requirements $\frac{r_k}{2}$, $k \in [K]$ bits/second reduces to the following combinatorial optimization problem:

Given scaled rates $r_k : k \in [K]$ and $N \leq K$, identify a partition $S_n : n \in [N]$ of $[K]$ that minimizes $\sum_{n=1}^{N} 2^{r(S_n)}$.

We investigate the computational complexity of this problem. Two cases are of interest. In the *variable bandwidth* case the number of slots N per second is a variable that is input as part of the problem instance. In the *fixed bandwidth* case, N is assumed known and fixed. We study the complexity of both these variations by looking at their decision versions.

A. Variable Bandwidth Case

SLOTTED PMIN : Given positive integer rates $r_1, r_2, \cdots, r_K$, number of slots N per second, $N \leq K$, a positive integer power P, is there a partition $S_n : n \in [N]$ of $[K]$ such that the inequality

$$\sum_{n=1}^{N} 2^{r(S_n)} \leq P \tag{3}$$

holds? □

Our first result is the following.

Theorem 5: SLOTTED PMIN is NP-complete.

Proof: 1) We first show SLOTTED PMIN ∈ NP by providing a polynomial time algorithm to check validity of a certificate partition. Assume without loss of generality that $r_1, r_2, \cdots, r_K$ are in increasing order. Clearly $P > 2^{r_K}$, and therefore $s(P) > r_K$, is a necessary condition for the existence of a partition $S_n : n \in [N]$ that satisfies (3). Compare $s(P)$ and r_K (in time polynomial in the input size) and reject the instance when $s(P) \leq r_K$. Hence we may focus on the instances that satisfy $s(P) > r_K$; these are fortunately instances where the rate values are bounded by the size of the input. This leads to the following algorithm. Let $(1 \ll x)$ denote the left shift operation on 1 to obtain the binary representation of 2^x.

```
Check Certificate(r_k : k ∈ [K], N, S_n, n ∈ [N], P):
    if (s(P) ≤ r_K)
        RETURN Certificate is invalid
    else {
        for (n = 1, 2, ··· , N) {
            r(S_n) = Σ_{k∈S_n} r_k
            P_n = (1 ≪ r(S_n))
        }
        P_tot = Σ_{n=1}^{N} P_n
        if (P_tot ≤ P)
            RETURN Certificate is Valid
        else
            RETURN Certificate is Invalid
    }
```

We only need to analyze the complexity of the "else" part; we show that despite the exponentiation the numbers involved are small. Since $r(S_n) \leq Kr_K < Ks(P)$, the number of operations needed to compute P_n (via left shifts) is $p_S(Ks(P))$. The span of P_n is at most $Ks(P)$. The span of P_{tot} is thus at most $Ks(P)\log N$. This is multiplied by N because of the "for" loop. Since $N \leq K \leq \text{Length}(I)$, the time needed to compute P_{tot} and compare it with P is $NKs(P)\log N$ which is $O(\text{Length}(I)^4)$. Thus `CheckCertificate` runs in polynomial time.

2) We next show that a subproblem of a strongly NP-complete problem 4-PARTITION [13, p.96] can be reduced in polynomial time to SLOTTED PMIN.

4-PARTITION : Given positive integers $a_1, a_2, \cdots, a_{4N}$ such that $\sum_{k=1}^{4N} a_k = NB$ where B is a positive integer, and $\frac{B}{5} < a_k < \frac{B}{3}$ for every $k \in [K]$, is there a partition $S_n : n \in [N]$ of $[4N]$ such that $a(S_n) = B$ for all $n \in [N]$? □

This is termed 4-PARTITION because if a partition exists, every set in the partition will have exactly 4 elements on account of $\frac{B}{5} < a_k < \frac{B}{3}$. Observe that B and N need not be directly input as part of the problem instance. As 4-PARTITION is strongly NP-complete [13, Theorem 4.3], there exists a polynomial $p(\cdot)$, such that 4-PARTITION$_p$ is NP-complete.

2a) Consider the transformation $f : D(\text{4-PARTITION}_p) \rightarrow D(\text{SLOTTED PMIN})$ defined as follows

$$\begin{aligned} \text{SLOTTED PMIN} &\leftarrow \text{4-PARTITION}_p \\ r_k &:= a_k \text{ for } k = 1, 2, \cdots, 4N \\ N &:= N \\ P &:= N2^B \end{aligned}$$

This is a polynomial time reduction because of the following. For any instance $I \in D(\text{4-PARTITION}_p)$, $B < 5a_k < 5p(\text{Length}\,(I))$. Since $N <$ Length(I), P can be computed in at most $p_M(s(N), 5p(\text{Length}\,(I)))$ steps proving the polynomial complexity of the reduction.

2b) We now prove $I \in Y(\text{4-PARTITION}_p)$ if and only if $f(I) \in Y(\text{SLOTTED PMIN})$. It is easy to see that I is a yes-instance of 4-PARTITION$_p$ with partition $S_n : n \in [N]$, then $f(I)$ is a yes-instance of SLOTTED PMIN with the same partition. In fact equality holds in (3). Conversely, if $S_n : n \in [N]$ is a desired partition for a yes-instance $f(I)$ of SLOTTED PMIN, we then have

$$\begin{aligned} P = N2^B &\geq \sum_{n=1}^{N} 2^{r(S_n)} \\ &\geq N\left(\prod_{n=1}^{N} 2^{r(S_n)}\right)^{\frac{1}{N}} \qquad (4) \\ &= N2^B \end{aligned}$$

where (4) follows from the arithmetic mean - geometric mean (AM-GM) inequality. Consequently, all inequalities are equalities, leading to $r\,(S_n) = B$ for all $n \in [N]$. Thus $S_n : n \in [N]$ is a desired partition for 4-PARTITION$_p$ and I is a yes-instance of 4-PARTITION$_p$. This completes the proof. ■

B. Fixed Bandwidth

We now look at the case when the number of slots N is fixed. In practice for a fixed communication technology, N is usually a fixed parameter.

N-SLOTTED PMIN : Given positive integer rates $r_1, r_2, \cdots, r_K$, where $N \leq K$, a positive integer power P, is there a partition $S_n : n \in [N]$ of $[K]$ such that the inequality

$$\sum_{n=1}^{N} 2^{r(S_n)} \leq P \qquad (5)$$

holds? □

Our second result is the following.

Theorem 6: N-SLOTTED PMIN ∈ P. In particular, there is an algorithm that solves N-SLOTTED PMIN with a running time $O\left(\text{Length}\,(I)^{N+1}\right)$.

Proof: We first show that number of partitions of $[K]$ that need to be checked is polynomial in the size of the input. We then argue that computing $\sum_{n=1}^{N} 2^{r(S_n)}$ for each of these partitions $S_n : n \in [N]$ of $[K]$ can be done in polynomial time. Subsequently, we provide a polynomial time algorithm that solves N-SLOTTED PMIN.

1) Associate the N-length vector $(r(S_n) : n \in [N])$ with the partition $S_n : n \in [N]$ of $[K]$. In order to solve N-SLOTTED PMIN we may focus on partitions whose associated vectors have components with values at most $s(P)$. This is because $r(S_n) < s(P)$ for every $n \in [N]$ is a necessary condition for partition $S_n : n \in [N]$ to satisfy (5). Let $T^{(K)}$ be set of vectors associated with all such partitions. As there are at most $s(P)$ values taken by each component, we have $|T^{(K)}| \leq s(P)^N$.

2) Assume without loss of generality $r_1, r_2, \cdots, r_K$ is in increasing order. Consider a candidate partition $S_n : n \in [N]$. As in `CheckCertificate` (see proof of Theorem 5), we reject if $s(P) \leq r_K$ (in polynomial time) and therefore focus on those instances with $s(P) > r_K$. From the discussion following `CheckCertificate`, $r(S_n), P_n, n \in [N]$, and P_{tot} can be computed in polynomial time for surviving partitions.

3) We now provide a dynamic programming algorithm `NSlottedPMIN` to solve N-SLOTTED PMIN. Let e_n denote the unit vector with 1 in the n^{th} component and 0 elsewhere. `NSlottedPMIN` computes $T^{(k)}$, defined as the set of vectors associated with partitions of $[k]$, recursively from $T^{(k-1)}$. Take $T^{(0)} := \{(0, 0, \cdots, 0)\}$ The set of vectors obtained by adding r_k to the n^{th} component of vectors in $T^{(k-1)}$, i.e.,

$$T^{(k-1)} \oplus_{s(P)} r_k e_n := \left\{ t + r_k e_n : t \in T^{(k-1)}, t_n + r_k < s(P) \right\}, \tag{6}$$

are the vectors associated with partitions of $[k]$ with $k \in S_n$. A union of these sets in (6) over n yields $T^{(k)}$.

`NSlottedPMIN`$(r_k : k \in [K], P)$:
`if` $(s(P) \leq r_k)$
    `RETURN` *No*
`else` {
    $T^{(0)} :=$ $\{(0, 0, \cdots, 0)\} \subseteq {\mathbb{Z}_+}^N$
    `for` $(k = 1, 2, \cdots, K)$
        $T^{(k)} :=$ $\cup_{n=1}^{N} \left(T^{(k-1)} \oplus_{s(P)} r_k e_n \right)$
    `for` $(t \in T^{(K)})$ {
        `for` $(n = 1, 2, \cdots, N)$
            $P_n = (1 \ll t_n)$
        $P_{tot} = \sum_{n=1}^{N} P_n$
        `if` $(P_{tot} \leq P)$
            `RETURN` *Yes*
    }
    `RETURN` *No*
}

We now analyze complexity of `NSlottedPMIN`. Since $|T^{(k-1)}| \leq s(P)^N$ for every $k \in [K]$, computation of (6) from $T^{(k-1)}$ requires at most $s(P)^N$ additions and as many comparisons. The values involved in these operations are at most $s(P)$. Therefore $T^{(k)}$ can be computed in $O\left(Ns(P)^N\right)$ steps. Since the components of vectors in $T^{(K)}$ are bounded in value by $s(P)$, computation of $P_n : n \in [N]$, computation of P_{tot}, and its comparison with P, can all be done in $O\left(s(P)^N \cdot N \log N\right)$ time.

Doing this for every vector in $T^{(K)}$ requires at most $O\left(s(P)^{N+1} \cdot N \log N\right)$ steps. Thus the "else" part of `NSlottedPMIN` runs to completion in $O\left(N \cdot \log N \cdot (\log P)^{N+1}\right)$ steps. This leads to an overall time complexity of $O\left(\text{Length}\,(I)^{N+1}\right)$, a polynomial in Length (I), since N is a fixed constant. ■

IV. SUM PRODUCT

In this section we assume that users provide rational approximations $\frac{f_k}{g_k}$ to exponentiated rate requirements 2^{r_k}. The problem of minimizing sum power then reduces to minimizing the sum of products of a given set of rational numbers. In Section IV-A we consider the case when N is input as part of the problem instance and in Section IV-B we study the case when N is known and fixed. The decision versions of both problems are proved to be NP-complete. Furthermore, we prove that when N is part of the problem instance the problem is strongly NP-complete. When N is fixed and g_k's are 1, a PPTA can be found as indicated in Section VI.

A. Variable Bandwidth

SUM PRODUCT : Given positive rational numbers $\frac{f_1}{g_1}, \frac{f_2}{g_2}, \cdots, \frac{f_K}{g_K}$ representing approximations to exponentiated rates, number of slots N per second, and a positive rational power $\frac{P}{Q}$, is there a partition $S_n : n \in [N]$ of $[K]$ such that the inequality

$$\sum_{n=1}^{N} \prod_{k \in S_n} \frac{f_k}{g_k} \leq \frac{P}{Q} \tag{7}$$

holds? □

Theorem 7: SUM PRODUCT is strongly NP-complete.

Proof: It is easy to verify SUM PRODUCT ∈ NP. We now look for a reduction from the following problem.

PRODUCTS OF 8 : Given positive rational numbers $\frac{f_1}{g_1}, \frac{f_2}{g_2}, \cdots, \frac{f_{4N}}{g_{4N}}$ such that $\prod_{k=1}^{4N} \frac{f_k}{g_k} = 8^N$, is there a partition $S_n : n \in [N]$ of $[4N]$ such that

$$\prod_{k \in S_n} \frac{f_k}{g_k} = 8 \tag{8}$$

for every $n = 1, 2, \cdots, N$? □

PRODUCTS OF 8 is strongly NP-complete. See Appendix A for a proof.

A PPTM from PRODUCTS OF 8 to SUM PRODUCT : Consider the following transformation $f : D(\text{PRODUCTS OF 8}) \rightarrow D(\text{SUM PRODUCT})$.

$$\begin{aligned} \text{SUM PRODUCT} &\leftarrow \text{PRODUCTS OF 8} \\ \frac{f_k}{g_k} &:= \frac{a_k}{b_k} \text{ for } k = 1, 2, \cdots, 4N \\ N &:= N \\ \frac{P}{Q} &:= 8N. \end{aligned}$$

Suppose $I \in Y(\text{PRODUCTS OF 8})$. A partition $S_n : n \in [N]$ of $[K]$ that satisfies (8) also satisfies (7) with equality. Hence $f(I) \in Y(\text{SUM PRODUCT})$.

Conversely let $S_n : n \in [N]$ be a partition of $[K]$ $(= [4N])$ that satisfies (7). We have

$$\begin{aligned} \frac{P}{Q} = 8N &\geq \sum_{n=1}^{N} \prod_{k \in S_n} \frac{f_k}{g_k} \\ &\geq N \left(\prod_{n=1}^{N} \prod_{k \in S_n} \frac{f_k}{g_k} \right)^{\frac{1}{N}} \qquad (9) \\ &= N \left(\prod_{k=1}^{4N} \frac{f_k}{g_k} \right)^{\frac{1}{N}} = 8N \end{aligned}$$

where (9) follows from AM-GM inequality. Thus all inequalities above are equalities, and

$$\prod_{k \in S_n} \frac{f_k}{g_k} = 8 \text{ for each } n \in [N]\,. \tag{10}$$

Thus $f(I) \in Y(\text{SUM PRODUCT}) \Rightarrow I \in Y(\text{PRODUCTS OF 8})$.

We have thus verified that f satisfies condition (i) of Definition 3. Conditions (ii), (iii), and (iv) are obviously satisfied, and f is a PPTM from PRODUCTS OF 8 to SUM PRODUCT. Lemma 4 implies SUM PRODUCT is strongly NP-complete. ■

B. Fixed Bandwidth

The number of slots is now fixed to N.

N-SUM PRODUCT : Given pairs of positive integers $\frac{f_1}{g_1}, \frac{f_2}{g_2}, \cdots, \frac{f_k}{g_k}$ representing exponentiated rates and $\frac{P}{Q}$ a pair of positive integers representing power, is there a partition $S_n : n \in [N]$ of $[K]$ such that the inequality

$$\sum_{n=1}^{N} \prod_{k \in S_n} \frac{f_k}{g_k} \leq \frac{P}{Q} \tag{11}$$

holds? □

Theorem 8: N-SUM PRODUCT is NP-complete.

Proof: It is easy to verify N-SUM PRODUCT ∈ NP. To prove N-SUM PRODUCT is NP-complete, we provide a reduction from the following problem.

SQRT SUBSET : Given positive integers $a_1, a_2, \cdots, a_K$ positive integers such that $\prod_{k=1}^{K} a_k$ is a perfect square, is there an $S_Q \subseteq [K]$ such that

$$\prod_{k \in S_Q} a_k = \sqrt{\prod_{k=1}^{K} a_k} \tag{12}$$

holds? □

SQRT SUBSET is NP-complete. See Appendix B for a proof.

Consider the transformation $f : D(\text{SQUARE-ROOT}) \rightarrow D(\text{N-SUM PRODUCT})$ as given below

$$\begin{aligned}
\text{N-SUM PRODUCT} &\leftarrow \text{SQRT SUBSET} \\
\frac{f_k}{g_k} &:= a_k \quad \text{for } k = 1, 2, \cdots, K \\
\frac{f_k}{g_k} &:= \sqrt{\prod_{k=1}^{K} a_k} \quad \text{for } k = K+1, K+2, \cdots, K+N-2 \\
\frac{P}{Q} &:= N\sqrt{\prod_{k=1}^{K} a_k}
\end{aligned}$$

For an instance $a_k : k \in [K]$ of SQRT SUBSET, denoted I, $\prod_{k=1}^{K} a_k$ is a perfect square. Thus $\sqrt{\prod_{k=1}^{K} a_k}$ is a positive integer and $f(I) \in D(\text{N-SUM PRODUCT})$. Moreover, f can be computed in time polynomial in the length of SQRT SUBSET (since N is a constant). If we can show $I \in Y(\text{SQRT SUBSET}) \Leftrightarrow f(I) \in Y(\text{N-SUM PRODUCT})$, we would have shown N-SUM PRODUCT is NP-complete. To this end we do the following.

Suppose $I \in Y(\text{SQRT SUBSET})$ with desired subset S_Q. Consider the following partition of $[K+N-2] : S_1 = S_Q, S_2 = [K] \setminus S_Q$, and $S_n = \{K+n-2\}, n = 3, 4, \cdots, N$. The sum of products for this partition is

$$\begin{aligned}
\sum_{n=1}^{N} \prod_{k \in S_n} \frac{f_k}{g_k} &= \sum_{n=1}^{N} \sqrt{\prod_{k=1}^{K} a_k} \\
&= N\sqrt{\prod_{k=1}^{K} a_k},
\end{aligned}$$

i.e., (11) holds with equality and therefore $f(I) \in Y(\text{N-SUM PRODUCT})$. Conversely, suppose $S_n : n \in [N]$, a partition of $[K+N-2]$, is a desired partition for N-SUM PRODUCT. We have

$$\begin{aligned}
\frac{P}{Q} &\geq \sum_{n=1}^{N} \prod_{k \in S_n} \frac{f_k}{g_k} \\
&\geq N \left(\prod_{n=1}^{N} \prod_{k \in S_n} \frac{f_k}{g_k} \right)^{\frac{1}{N}} \qquad (13) \\
&= N\sqrt{\prod_{k=1}^{K} a_k} \\
&= \frac{P}{Q}.
\end{aligned}$$

where (13) follows once again from the AM-GM inequality. Hence all inequalities are equalities and we have

$$\prod_{k \in S_n} \frac{f_k}{g_k} = \sqrt{\prod_{k=1}^{K} a_k} \text{ for each } n \in [N].$$

But each of $\frac{f_k}{g_k} : k \in \{K+1, K+2, \cdots, K+N-2\}$ belong $N-2$ different sets of the partition, call them $S_3, S_4, \cdots, S_N$. Thus we have $S_1 \subseteq [K]$, a desired subset for SQRT SUBSET. This proves $I \in Y(\text{SQRT SUBSET})$. ■

V. Analysis of the LPT algorithm

In this section, we extend Chandra & Wong's analysis in [1] of the performance of the LPT algorithm under the cost function

$$\log \frac{1}{N} \sum_{n=1}^{N} 2^{r(S_n)}, \qquad (14)$$

which is the logarithm of the normalized objective function in problem SLOTTED PMIN. This objective function is a measure of the power required per slot in the logarithmic scale (usually in decibel units) and happens to be the appropriate scale in practice for making comparisons.

The LPT algorithm is as follows. Let $r_1, r_2, \cdots, r_K$ be strictly positive and in descending order. Start with N empty sets. At step k, include r_k in any one of the slots with the least subset-sum.

For SUM PRODUCT, the LPT algorithm should include the index to the subset with the least subset-product instead of subset-sum. This is of course the same as including the index to the subset with the least subset-sum of exponents. It therefore follows that the bounds obtained below hold equally well for both problems even though they are presented in the context of SLOTTED PMIN.

Let lpt be the value of the objective function (14) on the output of the LPT algorithm, and let opt be that of the optimal one. Further, let $A := \frac{1}{N}\sum_{k=1}^{K} r_k$ be the average load per slot. The analysis of Chandra & Wong [1, Sec.5] shows that for the normalized L_p metrics, $p > 1$, the performance of the LPT algorithm relative to that of the optimal algorithm is upper bounded by

$$\max_{q} \left(\frac{\int_0^1 (q(t))^p \; dt}{A^p} \right)^{1/p}, \tag{15}$$

where the maximization is over $q : [0,1] \to \mathbb{R}_+$ that are piecewise-continuous, monotonically nonincreasing, with the following additional constraints:

$$\begin{aligned} q(0) &\leq \frac{3}{2} q(1), \\ \int_0^1 q(t) \; dt &= A. \end{aligned}$$

The same analysis holds for any strictly increasing strictly convex cost function of which the exponential cost function is but a particular example. We therefore have

$$\frac{\mathsf{lpt}}{\mathsf{opt}} \leq \max_{q} \frac{\log \int_0^1 e^{q(t)} \; dt}{A}, \tag{16}$$

where q is subjected to the same constraints indicated above. There is, however, one significant difference between the upper bounds in (15) for L_p metrics and the upper bound in (16) for the exponential cost function. The upper bound in (15) is invariant to scaling in q, while the upper bound in (16) is not. The latter upper bound will therefore depend on the parameter A which is the average load per slot of the given problem instance. This enables us to give better bounds that is based on the problem parameters. We next proceed to evaluate the upper bound as a function of average offered load per slot.

It is straightforward to see that worst-case (maximizing) q is of the form

$$q^*(t;\lambda) = \begin{cases} \frac{3A}{\lambda+2} & t \in [0, \lambda] \\ \frac{2A}{\lambda+2} & t \in (\lambda, 1], \end{cases}$$

where $\lambda \in [0,1]$ is a parameter. This choice of q^* satisfies all the specified constraints. We thus have

$$\frac{\mathsf{lpt}}{\mathsf{opt}} \leq \max_{\lambda \in [0,1]} \frac{\log\left(\lambda e^{\frac{3A}{\lambda+2}} + (1-\lambda) e^{\frac{2A}{\lambda+2}}\right)}{A}.$$

A plot of this worst-case inefficiency factor is given in Figure 1. The upper bound, and therefore ratio $\mathsf{lpt}/\mathsf{opt}$, is near 1 for small values of A, i.e., the LPT algorithm is asymptotically efficient as $A \to 0$. As might be expected, the worst-case inefficiency factor $\mathsf{lpt}/\mathsf{opt}$ is upper bounded by 3/2, and the upper bound approaches this limiting value for large A.

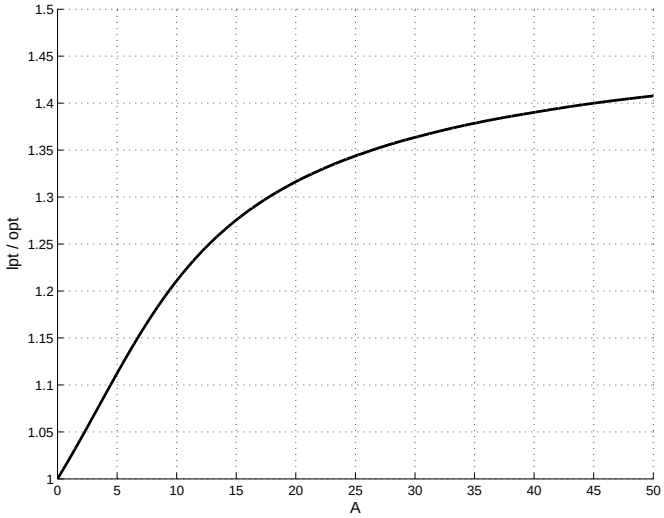

Fig. 1. Upper bound on the performance of the LPT algorithm relative to the optimal one, as a function of the load per slot.

VI. Summary and Concluding Remarks

We now summarize and provide a compilation of related open questions.

1) We showed that SLOTTED PMIN is NP-complete and N-SLOTTED PMIN $\in$ P. The running time of our algorithm for the latter problem is $O\left(\text{Length}\,(I)^{N+1}\right)$. Open question : Is SLOTTED PMIN strongly NP-complete?
2) We showed SUM PRODUCT is strongly NP-complete and N-SUM PRODUCT is NP-complete. Our reduction from SQRT SUBSET to N-SUM PRODUCT generates only integers. Thus the problem of N-SUM PRODUCT restricted to positive integers is also NP-complete. A dynamic programming approach can easily be employed to arrive at a pseudo polynomial time algorithm [13, p.94] to solve N-SUM PRODUCT, when restricted to integers. Thus N-SUM PRODUCT when restricted to integers is *not* strongly NP-complete. Is N-SUM PRODUCT with rational numbers strongly NP-complete?
3) SUM PRODUCT when restricted to integers is also NP-complete. This can be shown for example via a reduction from MULTIPROCESSOR SCHEDULING [13, SS8]. Is it strongly NP-complete?
4) Our intractability results on SUM PRODUCT (and those in the appendices) are of independent interest since they could be starting points of reductions to show intractability of other similar problems.
5) We identified performance bounds on Graham's LPT rule. Our upper bound is rather crude in that it is multiplicative in the logarithmic scale. It would be interesting to get other schemes with better performance.

Appendix A
PRODUCTS OF 8 is Strongly NP-Complete

Proof: It is easy to verify PRODUCTS OF 8 $\in$ NP. We now provide a PPTM from 3DM (three-dimensional matching) to prove PRODUCTS OF 8 is strongly NP-complete. The proof was inspired by the proof of [13, Theorem 4.3] and uses a similar argument.

Let $|X| = |Y| = |Z| = t$ and denote $X = \{x_1, x_2, \cdots, x_t\}$ and similarly Y and Z. For a $v \in V$, let $v\,(x)$ denote the index in X of the x-component, and so on, so that $v = \left(x_{v(x)}, y_{v(y)}, z_{v(z)}\right)$. Let $N\,(x_i)\,, N\,(y_j)\,, N\,(z_k)$ denote number of triples in

V that contain $x_i \in X, y_j \in Y, z_k \in Z$, respectively. Let $p_1, p_2, \cdots, p_t, q_1, q_2, \cdots, q_t, r_1, r_2, \cdots, r_t$ be $3t$ consecutive primes starting from 3. We now identify $4|V|$ rational numbers as follows:

- For $i = 1, 2, \cdots, t$, associate with $x_i \in X$ the *primary* rational number $2p_i$, and $(N(x_i) - 1)$ *secondary* rational numbers $p_i, p_i \cdots, p_i$.
- For $j = 1, 2, \cdots, t$, associate with $y_j \in Y$ the *primary* rational number $2q_j$, and $(N(y_j) - 1)$ *secondary* rational numbers $q_j, q_j, \cdots, q_j$.
- For $k = 1, 2, \cdots, t$, associate with $z_k \in Z$ the *primary* rational number r_k, and $(N(z_k) - 1)$ *secondary* rational numbers $4r_k, 4r_k, \cdots, 4r_k$.
- Associate with each triplet $v \in V$ the rational number $\frac{2}{p_{v(x)} q_{v(y)} r_{v(z)}}$.

We now verify that these rational numbers form a valid instance of PRODUCTS OF 8 with $N = |V|$. Note that

$$\sum_{i=1}^{t} N(x_i) = \sum_{j=1}^{t} N(y_j) = \sum_{k=1}^{t} N(z_k) = |V|.$$

The product of numerators of numbers associated with elements of X is

$$\prod_{i=1}^{t} (2p_i)\, p_i^{(N(x_i)-1)} = 2^t \prod_{i=1}^{t} p_i^{N(x_i)} = 2^t \prod_{v \in V} p_{v(x)}$$

Using similar relationships for rational numbers associated with elements of Y and Z we see that ratio of numerators and denominators is

$$2^t \cdot 2^t \cdot 4^{\sum_{k=1}^{t} (N(z_k)-1)} \cdot 2^{|V|} = 8^{|V|}$$

and therefore $4|V|$ rational numbers form a valid instance of PRODUCTS OF 8 with $N = |V|$.

We now prove that under the above transformation I is a yes-instance of 3DM iff it is a yes-instance of PRODUCTS OF 8.

Let $V' \subseteq V$ be a matching for X, Y, Z. For $v' \in V'$, group the rational number associated to v' with *primary* rational numbers corresponding to components of v', i.e.,

$$\left\{ \frac{2}{p_{v'(x)} q_{v'(y)} r_{v'(z)}}, 2p_{v'(x)}, 2q_{v'(y)}, r_{v'(z)} \right\}$$

For $v \in V \setminus V'$ group the rational number associated to v with *secondary* rational numbers corresponding to components of v, i.e.,

$$\left\{ \frac{2}{p_{v(x)} q_{v(y)} r_{v(z)}}, p_{v(x)}, q_{v(y)}, 4r_{v(z)} \right\}$$

By our choice of the number of secondaries, such a grouping exists, and all numbers will belong to exactly one subset of four elements. Each subset has product 8, and we have our desired partition for PRODUCTS OF 8.

Conversely suppose $S_n : n \in [N]$ is an appropriate partition for PRODUCTS OF 8. We claim that every S_n contains one rational number corresponding to a triplet in V, and three other rational numbers, each corresponding to elements in X, Y, Z, respectively.

Consider an S_n. There is at least one rational number corresponding to a triplet because otherwise the odd primes cannot be cancelled. Moreover, there are N subsets and N triplets, which implies that there is exactly one rational number corresponding to a triplet in each subset.

Let the denominator of this rational number in S_n be $p_{i(n)}q_{j(n)}r_{k(n)}$. Then the other rational numbers in S_n contribute a product equal to

$$4p_{i(n)}q_{j(n)}r_{k(n)}. \tag{17}$$

By the uniqueness of prime factor decomposition there must be exactly one element from the rational numbers corresponding to $x_{i(n)}$, exactly one element from those corresponding to $y_{j(n)}$, and exactly one from those corresponding to $z_{k(n)}$, i.e., there are exactly 4 elements in S_n. Since the product in (17) has 4 as a factor, the three elements are either all primary or all secondary. We thus conclude that S_n contains one rational number corresponding to V and three rational numbers corresponding to $x_{i(n)}, y_{j(n)}, z_{k(n)}$ all primary or all secondary. Since $S_n, n \in [N]$ is a partition of the rational numbers, t of these sets contain primary rational numbers and $N-t$ of them contain secondary rational numbers. Consider the t sets with the primaries. The set of triplets associated with these t subsets is a matching V' for X, Y, Z, and the instance is a yes-instance of 3DM.

The size of an input instance to 3DM is polynomial in $|V|$ ($\geq t$). We now prove that the rational numbers in the transformed instance of PRODUCTS OF 8 can be generated in time polynomial in $|V|$. This requires identification of $(3t+1)$ primes. We identify in a brute-force fashion $3t+1$ prime numbers starting from 3. The nth prime does not exceed $12n\left(\log n + \log \frac{12}{e}\right)$ [15, Theorem 4.7]. Since $12n\left(\log n + \log \frac{12}{e}\right) \leq 12n\left(\log n + 2\right) \leq 12n\left(2\log n + 2\right) \leq 24n^2$, a polynomial in n, we need to check at most $24\left(3t+1\right)^2$ numbers to identify $(3t+1)$ primes. Checking a number for prime can be done in polynomial time [16]. Thus $(3t+1)$ primes can be identified in time polynomial in t. Consequently, the rational numbers can be computed in polynomial time, and we have a polynomial time reduction.

Finally, the numbers that represent the rational numbers are bounded by $q_t^3 \leq 24^3\left(3t+1\right)^6$, i.e., a polynomial function of size of input instance. The transformation is therefore a PPTM from 3DM to PRODUCTS OF 8. This proves the strongly NP-complete nature of the latter. ■

APPENDIX B
SQRT SUBSET IS NP-COMPLETE

Proof: It is easy to verify SQRT SUBSET $\in$ NP. We provide a reduction from SUBSET PRODUCT, a product analog of SUBSET SUM.

SUBSET PRODUCT : Given positive integers $b_1, b_2, \cdots, b_K$ and P such that P divides $\prod_{k=1}^{K} b_k$, is there a subset $S \subseteq [K]$ such that $\prod_{k\in S} b_k = P$ holds? □

SUBSET PRODUCT, as defined above, is restricted to those instances of SUBSET PRODUCT [SP14] [13, p.224] where P divides $\prod_{k=1}^{K} b_k$. Given an instance of SUBSET PRODUCT [SP14], this condition is a necessary condition for it to be a yes-instance. Furthermore, this necessity can be checked in polynomial time. Thus the NP-completeness of SUBSET PRODUCT [SP14] [13, p.224] implies SUBSET PRODUCT as defined above is NP-complete.

Consider the following reduction $f : D\,(\text{SUBSET PRODUCT}) \rightarrow D\,(\text{SQRT SUBSET})$.

$$\begin{aligned} \text{SQRT SUBSET} \quad &\leftarrow \quad \text{SUBSET PRODUCT} \\ a_k \quad &:= \quad b_k \text{ for } k = 1, 2, \cdots, K \\ a_{K+1} \quad &:= \quad \left(\prod_{k=1}^{K} b_k \right)^2 / P \\ a_{K+2} \quad &:= \quad P \prod_{k=1}^{K} b_k \end{aligned}$$

Note that $\prod_{k=1}^{K+2} a_k$ is a perfect square and a_{K+1} is a positive integer; thus $a_k : k \in [K+2]$ is a valid instance of SQRT SUBSET. From the observations made in Section II, it is clear that f can be computed in time polynomial in the size of SUBSET PRODUCT. It remains to prove $I \in Y(\text{SUBSET PRODUCT}) \Leftrightarrow f(I) \in Y(\text{SQRT SUBSET})$.

Suppose $I \in Y(\text{SUBSET PRODUCT})$. Let $S \subseteq [K]$ be the desired subset for SUBSET PRODUCT. It is easily verified that $S_Q = S \cup \{K+1\}$ is a desired subset for SQRT SUBSET. Thus $f(I) \in Y(\text{SQRT SUBSET})$. Conversely, suppose S_Q is a square root subset for SQRT SUBSET. Exactly one of $K+1, K+2$ is an element of S_Q. Without loss of generality we assume $K+1 \in S_Q$. Then $S := S_Q \setminus \{K+1\} \subseteq [K]$ is the desired subset for SUBSET PRODUCT. This completes the proof. ■

ACKNOWLEDGEMENT

The authors are indebted to Dr. Kavitha Telikepalli of the CSA department, Indian Institute of Science, for an initial proof of Theorem 7 and for several discussions that led to the results in this paper.

REFERENCES

[1] A. K. Chandra and C. K. Wong, "Worst-case analysis of a placement algorithm related to storage allocation," *SIAM Journal of Computing*, vol. 4, no. 3, pp. 249–263, 1975.

[2] R. L. Graham, "Bounds on multiprocessing timing anomalies," *SIAM J. Appl. Math.*, vol. 17, pp. 263–269, 1969.

[3] D. N. C. Tse and S. Hanly, "Multi-access fading channels - Part I: Polymatroid structure, optimal resource allocation and throughput capacities," *IEEE Trans. Inf. Theory*, vol. 44, pp. 2796–2815, Nov. 1998.

[4] TIA/EIA/IS-856-A, "cdma2000 High Rate Packet Data Air Interface Specification," Telecommunications Industry Association, March 2004 (http://www.3gpp2.org).

[5] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: John Wiley & Sons, 1991.

[6] W.-T. Chen and N.-F. Huang, "The strongly connecting problem on multihop packet radio networks," *IEEE Transactions on Communications*, vol. 37, pp. 293–295, March 1989.

[7] L. M. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc, "Power consumption in packet radio networks," in *Lecture Notes In Computer Science; Vol. 1200 archive, Proceedings of the 14th Annual Symposium on Theoretical Aspects of Computer Science*. Nice, France: Springer-Verlag, London, UK, 1997, pp. 363 – 374.

[8] E. Arikan, "Some complexity results about packet radio networks," *IEEE Trans. Inf. Theory*, vol. 30, pp. 681–685, Jul. 1984.

[9] L. Tassiulas, "Linear complexity algorithms for maximum throughput in radio networks and input queued switches," in *Proc. of IEEE INFOCOM*, Apr. 1998, pp. 533–539.

[10] L. Georgiadis, M. J. Neely, and L. Tassiulas, *Resource Allocation and Cross-Layer Control in Wireless Networks*, ser. Foundations and Trends in Networking. Hanover, MA, USA: now Publishers Inc., 2006, vol. 1, no. 1.

[11] J. Y. T. Leung and W. D. Wei, "Tighter bounds on a heuristic for a partition problem," *Inform. Process. Lett.*, vol. 56, pp. 51–57, 1995.

[12] R. R. Goldberg and J. Shapiro, "A tight upper bound for the k-partition problem on ideal sets," *Operations Research Letters*, vol. 24, pp. 165–173, 1999.

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

[14] A. W. Marshall and I. Olkin, *Inequalities: Theory of Majorization and Its Applications*. Academic, 1979.

[15] T. M. Apostol, *Introduction to Analytic Number Theory*, Springer International Student Edition ed. New Delhi: Narosa Publishing House, 1993.

[16] M. Agrawal, N. Kayal, and N. Saxena, "Primes is in P," *Annals of Mathematics*, vol. 160, pp. 781–793, Sep. 2004.