Joint Scheduling and Resource Allocation in CDMA Systems

Vijay G. Subramanian, Randall A. Berry, and Rajeev Agrawal

Expanded Technical Report: A shorter version of this paper will apear in IEEE Transactions on Information Theory.

Abstract

In this paper, the scheduling and resource allocation problem for the downlink in a CDMA-based wireless network is considered. The problem is to select a subset of the users for transmission and for each of the users selected, to choose the modulation and coding scheme, transmission power, and number of codes used. We refer to this combination as the physical layer operating point (PLOP). Each PLOP consumes different amounts of code and power resources. The resource allocation task is to pick the "optimal" PLOP taking into account both system-wide and individual user resource constraints that can arise in a practical system. This problem is tackled as part of a utility maximization problem framed in earlier papers that includes both scheduling and resource allocation. In this setting, the problem reduces to maximizing the weighted throughput over the state-dependent downlink capacity region while taking into account the system-wide and individual user constraints. This problem is studied for the downlink of a Gaussian broadcast channel with orthogonal CDMA transmissions. This results in a tractable convex optimization problem. A dual formulation is used to obtain several key structural properties. By exploiting this structure, algorithms are developed to find the optimal solution with geometric convergence.

Index Terms

Cellular network, channel-aware scheduling, code division multiple access (CDMA), convex optimization, resource allocation, utility maximization.

I. Introduction

Efficient scheduling and resource allocation are essential components for enabling high-speed data access in wireless networks. In this setting, scheduling is complicated due to the time-varying fading of wireless channels. A variety of wireless scheduling approaches have been proposed that *opportunistically* exploit these temporal variations to improve the over-all system performance, e.g. [1]–[20]. These approaches attempt to transmit to users during periods when they have good channel quality (and can support higher transmission rates), while maintaining some form of fairness among the users.

Wireless scheduling approaches can be divided into two classes: (*i*) time-division multiplexed (TDM) systems, where a single user is transmitted to in each time-slot, as in the HDR system (CDMA 1xEVDO) [21], [22], and (*ii*) systems in which the transmitter can simultaneously transmit to multiple users in each time-slot, by using a combination of TDM and another multiplexing technique such as CDMA or OFDMA. In the latter case, in addition to deciding which users to schedule, the available physical layer resources, such as bandwidth and power, must be divided among the users. In this paper, we consider the

V. G. Subramanian is with the Hamilton Institute, NUIM, Maynooth, Co. Kildare, Ireland, e-mail: vijay.subramanian@nuim.ie. R. A. Berry is with the Dept. of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208, USA, email: rberry@eecs.northwestern.edu. R. Agrawal is with the Advanced Networks and Performance Group, Motorola Inc., Arlington Heights, IL, USA, e-mail: rajeev.agrawal@motorola.com.

The majority of this work was done while V. G. Subramanian was with the Mathematics of Communication Networks Group, Motorola Inc. His work is also supported in part by SFI grants IN3/03/I346 and 07/IN.1/I901.

The work of R. A. Berry was supported in part by the Northwestern-Motorola Center for Communications and NSF CAREER award CCR-0238382.

A preliminary version of this paper was presented at the 2nd Workshop on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt '04), Cambridge, UK, March 24-26, 2004.

second class of systems, where CDMA is used to multiplex users within a time-slot.[1] Examples of this type of system include the High Speed Downlink Packet Access (HSDPA) approach developed for W-CDMA [23, Chapter 11, pp. 279-304] or the 1x-EVDV approach for CDMA2000 [24]. In these systems, the physical layer resources and information rate assigned to a user are specified by selecting the number of spreading codes, the fraction of transmission power, and the modulation and coding scheme (MCS). We refer to a combination of these as the physical layer operating point (PLOP).

The main problem addressed in this paper is to specify the optimal PLOP at each scheduling instant, which in turn specifies the vector of user transmission rates. This problem must be solved once every time-slot (e.g., 2msec in HSDPA or 1.25 msec in 1x-EVDV), and so requires a computationally efficient solution. We consider this in the context of the gradient-based scheduling framework presented in [1], [2]. In this framework, in each time-slot the objective is to chose the transmission rate vector that has the largest projection onto the gradient of the total system utility. The utility is a function of each user's throughput and is used to quantify fairness. Several such gradient-based scheduling algorithms have been studied for TDM systems, including the proportionally fair algorithm [22], which is based on a log utility function. In [1], a larger class of utility functions is considered that allow efficiency and fairness to be traded-off.

The problem considered here can be viewed as finding the maximum weighted sum throughput for a downlink (broadcast) channel, where the weights are determined by the gradient of the utility. Our solution is general in that it also applies to other scheduling algorithms, which may provide these weights using different approaches. For example, these weights could be based on queue size information as in the "MaxWeight" scheduling algorithms studied in [3], [4], [17], [26]. For the model studied here, the feasible rate region is convex; hence, by varying these weights we can determine the boundary of this region. In related work, the problem of allocating resources to maximize the weighted sum capacity for the downlink channel has been considered from an information theoretic perspective in [28], [29]. Both of these works assume the use of optimal information theoretic (multi-user) coding/decoding.[2] The work in [29] also considers several sub-optimal transmission strategies, such as approaches based on TDM, CDMA without multiuser coding with all users orthogonalized and FDM; the focus in [29] is on deriving the long-term average throughputs over multiple fading states under a long-term average power constraint. Here, we focus on optimally allocating resources for the specific fading state realized in each scheduling time-slot; the total power is constrained within each time-slot as well. The problem within each time-slot can be viewed as a special case of the CDMA without multiuser coding approach in [29] where the fading is constant. However, focusing on this case enables us to generate a much simpler optimal algorithm. We also take into account additional "per-user" power and code constraints that are imposed by the capability of each mobile in a practical system.[3] The algorithms in [29] make use of specific properties of the function $a\log(1+bx)$ that do not generalize with the addition of these "per-user" constraints.

Simultaneously and independently of our work,[4] Kumaran and Viswanathan studied a similar problem in [31]. They also consider the problem of maximizing the weighted capacity within a time-slot and derive several related structural characteristics. We note that the work in [31] does not include per-user code constraints, but does contain an algorithm with a per-user rate constraint.

We begin with formulating the scheduling and resource allocation problem in Section II. This formulation is based on a gradient-based scheduling approach from [1], [2], which we also review. By substituting

[1]The model in this paper also applies to OFDMA systems when each sub-channel that may be assigned to a user has the same channel state (this may model a system in which OFDMA sub-channels are formed by interleaving tones from across the frequency band). A more detailed discussion of such problems for OFDMA systems can be found in [25], [36].

[2]In the special case of maximizing the equal weight sum capacity in a flat fading channel, the information theoretic optimal approach is to transmit to only one user in each time-slot [28] and hence, multi-user decoding is not required. However, this is not true if the users are not weighted equally or for other channel models, such a multiple antenna channel. It also does not hold when additional per user constraints are present, as is the case here.

[3]Moreover, these constraints may vary from mobile to mobile. For example, the initial mobile devices for HSDPA can receive up to 5 spreading codes, while future devices may be able to receive up to 15 spreading codes.

[4]A version of our work was first presented in [30].

an analytical formula relating the rate, power, codes, and SINR, we obtain an analytically tractable problem with nice convexity properties. In Sections III-IV, we use a dual formulation to study this problem. We obtain analytic formulas for many of the quantities of interest. For others we have to resort to a numerical search (aided with some heuristics based on the structure of the problem). However, these numerical searches are in a single dimension (due to the dual formulation) rather than over the multidimensional PLOP space. Also, thanks to the convexity of the problem, these algorithms converge geometrically fast. Along the way we obtain key structural properties of the optimal solution including:

1) A tight upper bound on the number of users scheduled as a function of the per-user code constraints; when each user can use all the codes, this bound implies at most two users will be scheduled.
2) Given a code assignment, the optimal power allocation is given by a “water-filling” algorithm, which is modified to take into account the different weights assigned to each user and any per-user power constraints.
3) For a fixed code assignment, the optimal “water-level” (Lagrange multiplier) can be found in finite time. Specifically, we give an iterative algorithm which will terminate in at most M steps, where M is the number of users allocated codes.
4) For a given water-level, the users that are scheduled are determined by simply sorting all the users based on a “per-user metric” that is given analytically.
5) Codes are only time-shared when ‘ties’ occur in the above sort. This corresponds to a point where the dual function is not differentiable. At these values the optimal time-sharing can be found using the subgradients of this function. We give a complete characterization of these subgradients.

We conclude the paper with simulation results comparing this algorithm with a base-line heuristic in Section V.

II. Gradient-based scheduling and resource allocation problem

We consider the downlink of a wireless communication system with K users. The channel conditions are time-varying and modeled by a stochastic channel state vector $\mathbf{e}_t = (e_{1,t}, \ldots, e_{K,t})$, where $e_{i,t}$ represents the channel state of the ith user at time t. Associated with each channel state vector is a rate-region $\mathcal{R}(\mathbf{e}_t) \subset \mathbb{R}_+^K$, which indicates the set of feasible transmission rates $\mathbf{r}_t = (r_{1,t}, \ldots, r_{K,t})$.

Our point of departure is the gradient-based scheduling framework in [1], [2]. In this framework, at each scheduling instant a rate vector $\mathbf{r}_t \in \mathcal{R}(\mathbf{e}_t)$ is selected that has the maximum projection onto the gradient of a system utility function $\nabla U(\mathbf{W}_t)$, where

$$U(\mathbf{W}_t) = \sum_{i=1}^{K} U_i(W_{i,t}),$$

and, for each user i, $U_i(W_{i,t})$ is a increasing concave utility function of the user's average throughput, $W_{i,t}$, up to time t. In other words, the scheduling and resource allocation decision is the solution to

$$\max_{\mathbf{r}_t \in \mathcal{R}(\mathbf{e}_t)} \nabla U(\mathbf{W}_t)^T \cdot \mathbf{r}_t = \max_{\mathbf{r}_t \in \mathcal{R}(\mathbf{e}_t)} \sum_i \left. \frac{dU_i(x)}{dx} \right|_{x=W_{i,t}} \cdot r_{i,t}. \tag{1}$$

For example, one class of utility functions given in [1], [33] is

$$U_i(W_{i,t}) = \begin{cases} \frac{c_i}{\alpha}(W_{i,t})^\alpha, & \alpha \le 1,\ \alpha \neq 0, \\ c_i \log(W_{i,t}), & \alpha = 0, \end{cases} \tag{2}$$

where $\alpha \le 1$ is a fairness parameter and c_i is a quality of service (QoS) weight. In this case, (1) becomes

$$\max_{\mathbf{r}_t \in \mathcal{R}(\mathbf{e}_t)} \sum_i c_i (W_{i,t})^{\alpha-1} r_{i,t}. \tag{3}$$

With equal QoS weights, $\alpha = 1$ results in a “maximum throughput” rule that maximizes the total throughput during each slot. For $\alpha = 0$, this results in the proportionally fair rule.

The preceding policy can be generalized to allow the utility to depend on other parameters such as a user's queue size or delay. For example, consider the utility

$$U_i(W_{i,t}, Q_{i,t}) = \frac{c_i}{\alpha}(W_{i,t})^{\alpha} - \frac{d_i}{p}(Q_{i,t})^{p},$$

where $Q_{i,t}$ represents the queue length of user i at time t, d_i is a QoS weight for user i's queue length and $p > 1$ is a fairness parameter associated with the queue length. In this case, (1) is replaced by[5]

$$\max_{\mathbf{r}_t \in \mathcal{R}(\mathbf{e}_t)} \sum_i \left(c_i(W_{i,t})^{\alpha-1} + d_i(Q_{i,t})^{p-1}\right) r_{i,t}. \tag{4}$$

Special cases of this policy with $c_i = 0$ have been shown to be stabilizing policies in a variety of settings [3], [4], [17], [26]. In [27] it was shown that for specific choices of c_i and d_i this policy will maximize the total network utility ($\sum_i \frac{c_i}{\alpha}(W_{i,t})^{\alpha}$) subject to a network stability constraint.

In general, we consider the problem

$$\max_{\mathbf{r}_t \in \mathcal{R}(\mathbf{e}_t)} \sum_i w_{i,t} r_{i,t}, \tag{5}$$

where $w_{i,t} \geq 0$ is a time-varying weight of the ith user at time t. In the preceding examples, these weights are given by the gradient of the utility; however, other methods for generating these weights are also possible. We note that (5) must be re-solved at each scheduling instant because of changes in both the channel state and the weights (e.g., the gradient of the utility). The former changes are due to the time-varying nature of the wireless channel, whereas the latter changes are due to new arrivals and past service decisions.

The solution to this problem depends on the state dependent capacity region $\mathcal{R}(\mathbf{e}_t)$, which we assume is known at time t.[6] In this paper, we consider a model that is appropriate for a CDMA system, such as HSDPA or 1xEVDV. This model is parameterized by two sets of physical layer parameters: the number of spreading codes, n_i and the transmission power p_i assigned to each user i. Each choice of these parameters specifies a PLOP, which must satisfy the following constraints:

$$n_i \leq N_i, \tag{6}$$

$$\sum_i n_i \leq N, \tag{7}$$

$$\sum_i p_i \leq P. \tag{8}$$

Here, (7) and (8) are system constraints on the total number of spreading codes and the total system power, while (6) is a per user constraint on the number of codes that can be assigned to user i.

We assume that all spreading codes are mutually orthogonal, so that the only interference is from other cells. Moreover, in a fully loaded system, the other cells use a constant total power and thus power allocation per user and code does not have an impact on the interference. Hence, we assume that the interference power is constant. We then let the channel state e_i indicate user i's received signal-to-interference plus noise ratio (SINR) per unit power, where we have suppressed the dependence on t for convenience.[7] In this case, the SINR per code for user i is given by $SINR_i = \frac{p_i}{n_i} e_i$. We model the achievable rate per code by

$$\frac{r_i}{n_i} = \Gamma(\zeta_i \cdot SINR_i).$$

[5]Note that we take the negative of the gradient of the utility with respect to queue length. This is because the queue length is decreasing in the transmission rate assigned to a user while the throughput is increasing.

[6]While, in a practical system, the exact channel state will not be perfectly known at the transmitter, some estimate of it is usually available, for example, via channel quality feedback.

[7]In other words, if we neglect other cell interference then e_i is simply the signal-to-noise ratio (SNR) of user i per unit power.

Here, Γ corresponds to the Shannon capacity for a Gaussian noise channel with the given SINR, i.e., $\Gamma(x) = B\log(1+x)$, where B indicates the symbol rate (i.e., the chip rate/spreading factor), and $\zeta_i \in (0,1]$ is a scaling factor that can be used to model the "gap from capacity" in a practical system. This is a reasonable model for systems that use sophisticated coding techniques, such as Turbo codes. Redefining e_i to be $e_i\zeta_i$, the rate region is then

$$\begin{aligned}\mathcal{R}(\mathbf{e}) = \Big\{ \mathbf{r} \geq 0 : r_i = n_i B \log\left(1 + \frac{p_i e_i}{n_i}\right), \\ n_i \leq N_i \ \forall i, \sum_i n_i \leq N, \sum_i p_i \leq P \Big\}.\end{aligned} \tag{9}$$

Without the per-user code constraints, this is equivalent to the achievable rate-region obtained in [29] for TDM, CDMA without multiuser coding and FDM, where in each case the user is subject to constant fading over the available degrees of freedom. Notice that in (9), we allow the number of codes per user to take on a non-integer value. Of course, in a practical system these must be integer valued. However, we will show that, in most cases, the solution to this relaxed problem results in integer values for n_i.

We can now state the optimization problem in (5) as

$$V^* := \max_{(\mathbf{n},\mathbf{p})\in\mathcal{X}} V(\mathbf{n},\mathbf{p}) \quad [\textbf{Primal problem}]$$

subject to:

$$\sum_i n_i \leq N, \tag{10}$$

$$\sum_i p_i \leq P,$$

where

$$V(\mathbf{n},\mathbf{p}) := \sum_i w_i n_i \ln\left(1 + \frac{p_i e_i}{n_i}\right), \tag{11}$$

$$\mathcal{X} := \big\{(\mathbf{n},\mathbf{p}) \geq \mathbf{0} : n_i \leq N_i \ \ \forall i\big\}, \tag{12}$$

$\mathbf{n}$ is a vector of code allocations, and $\mathbf{p}$ is a vector of power allocations. We have normalized the objective by $B/\ln(2)$ to simplify notation. Note that the constraint set $\mathcal{X}$ is convex. It can also be verified that V is concave in $(\mathbf{n},\mathbf{p})$.

A. Additional Constraints

In addition to (6)-(8), there may be several other constraints on the feasible PLOPs in a practical system. This includes the following "per user" constraints:

i.) peak power constraint:

$$p_i \leq P_i, \qquad \forall i.$$

ii.) maximum SINR (per code) constraint:

$$SINR_i = \frac{p_i e_i}{n_i} \leq S_i \Leftrightarrow p_i \leq S_i \frac{n_i}{e_i},\ \forall i.$$

iii.) maximum rate per code[8] constraint:

$$\begin{aligned}\frac{r_i}{n_i} = \ln\left(1 + \frac{p_i e_i}{n_i}\right) \leq (R/N)_i \\ \Leftrightarrow \quad p_i \leq (e^{(R/N)_i} - 1)\frac{n_i}{e_i},\ \forall i.\end{aligned}$$

[8]As in the previous section, we continue to normalize the rate, r_i, by $B/\ln(2)$.

iv.) minimum rate per code constraint:

$$\frac{r_i}{n_i} = \ln\left(1 + \frac{p_i e_i}{n_i}\right) \geq (\check{R}/N)_i$$
$$\Leftrightarrow \quad p_i \geq (e^{(\check{R}/N)_i} - 1)\frac{n_i}{e_i}, \ \forall i.$$

v.) maximum rate constraint:

$$\begin{aligned} r_i = n_i \ln\left(1 + \frac{p_i e_i}{n_i}\right) \leq R_i \\ \Leftrightarrow \qquad p_i \leq (e^{R_i/n_i} - 1)\frac{n_i}{e_i}, \ \forall i. \end{aligned} \tag{13}$$

vi.) minimum rate constraint:

$$r_i = n_i \ln\left(1 + \frac{p_i e_i}{n_i}\right) \geq \check{R}_i$$
$$\Leftrightarrow \qquad p_i \geq (e^{\check{R}_i/n_i} - 1)\frac{n_i}{e_i}, \ \forall i.$$

These constraints can arise due to various implementation considerations. For example, a constraint on the rate per code is imposed by the maximum or minimum rate of the available modulation and coding schemes: a modulation order limitation usually results in the former and minimum underlying coding rate results in the latter. On the other hand, a maximum rate constraint arises because there is only a finite amount of data available to send to each mobile at any time. A minimum rate constraint can be used to model the case where the system is trying to guarantee a certain level of service to that user.[9]

All of the above constraints can be viewed as special cases of a *per user power constraint* with the form:

$$SINR_i = \frac{p_i e_i}{n_i} \in [\check{s}_i(n_i), s_i(n_i)], \quad \forall i,$$

where the function $s_i(n_i)$ is also dependent on the fixed (for a given optimization problem) parameters $P_i, S_i, e_i, R_i, (R/N)_i$, and the function $\check{s}_i(n_i)$ is dependent on the parameters $\check{R}_i, (\check{R}/N)_i$. Non-negativity restrictions on power necessarily imply that $\check{s}_i(n_i) \geq 0$. We primarily focus on two special cases of this:

I. $s_i(n_i) \equiv s_i$ and $\check{s}_i(n_i) \equiv \check{s}_i$ do not depend on n_i,
II. $s_i(n_i) \equiv s_i = \infty$ and $\check{s}_i(n_i) \equiv s_i = 0$.

We refer to these as Type I and Type II per-user power constraints, respectively. A Type I constraint models the case where there is a maximum and minimum constraint on the SINR or rate per code. A Type II constraint corresponds to no per-user power constraints.

With the per user power constraints, the constraint set $\mathcal{X}$ is further restricted to

$$\mathcal{X} := \left\{(\mathbf{n}, \mathbf{p}) \geq \mathbf{0} : n_i \leq N_i, \frac{\check{s}_i(n_i) n_i}{e_i} \leq p_i \leq \frac{s_i(n_i) n_i}{e_i}, \ \forall i\right\}.$$

The set $\mathcal{X}$ continues to be convex if $s_i(n_i)n_i$ is a concave function of n_i and $\check{s}_i(n_i)n_i$ is a convex function of n_i. Note that $s_i(n_i)n_i$ is indeed concave for the two special cases (I-II) mentioned above, as well as the case of a peak power constraint, and $\check{s}_i(n_i)n_i$ is always convex in the previous examples. Unless otherwise mentioned, we will assume this set is convex in the following.

For the maximum rate constraint case (13), $s_i(n_i)n_i$ is convex in n_i, and so the set $\mathcal{X}$ will not be convex. However, one can still get a convex formulation [36] for this case by instead viewing the rate

[9]Of course, with minimum rate and minimum rate per code constraints the resulting optimization may be infeasible, depending on the other constraints and the channel states.

r_i as an additional optimization variable, so that the objective is now to maximize $\sum_i w_i r_i$, where r_i is constrained to satisfy

$$r_i \leq n_i \log\left(1 + \frac{p_i e_i}{n_i}\right),$$

and $r_i \in [0, R_i]$. The final solution in this case is quite similar to the analysis that follows in this paper. However, to simplify our discussion we do not consider this constraint here and simply focus on cases I and II above.

In addition to these per user power constraints, there may also be a constraint on the maximum number of users M scheduled in a time-slot, i.e., users with positive code and power assignments.[10] We will prove later (see Lemma 4.9) that such a constraint will in most cases automatically be satisfied by the optimal solution (assuming the selected users have enough data to send) as long as $M-1$ users can fully utilize the available code budget, i.e., the sum of the N_i's for any subset of $M-1$ users is greater than or equal to N. For example, if $N_i \geq 5$ for all i and $N \leq 15$, then no more than 4 users need to be scheduled in any time-slot under the optimal scheme.

III. The dual problem and convex optimization

In this section we begin considering the solution to (10), which determines the users to be scheduled as well as the amount of power and the number of codes to be assigned to each user. We solve the optimization problem by looking at the dual formulation. The objective is concave and since the constraints are linear, there will be no duality gap (see [34]). This allows us to use the solution of the dual to compute the solution of the primal.

A. The Dual Problem

Define a Lagrangian for the primal problem (10) by

$$\begin{aligned} L(\mathbf{n}, \mathbf{p}, \lambda, \mu) := \sum_i w_i n_i \ln\left(1 + \frac{p_i e_i}{n_i}\right) + \\ \lambda\left(P - \sum_i p_i\right) + \mu\left(N - \sum_i n_i\right). \end{aligned} \tag{14}$$

The corresponding dual function is

$$L(\lambda, \mu) := \max_{(\mathbf{n},\mathbf{p})\in\mathcal{X}} L(\mathbf{n}, \mathbf{p}, \lambda, \mu). \tag{15}$$

The dual problem is then given by:

$$L^* := \min_{(\lambda,\mu)\geq 0} L(\lambda, \mu) \qquad [\textbf{Dual problem}]. \tag{16}$$

Also, with some further abuse of notation, we define

$$L(\lambda) := \min_{\mu\geq 0} L(\lambda, \mu) = \min_{\mu\geq 0} \max_{(\mathbf{n},\mathbf{p})\in\mathcal{X}} L(\mathbf{n}, \mathbf{p}, \lambda, \mu). \tag{17}$$

[10]For example, in HSDPA such a constraint arises because the system cannot schedule more users than the number of shared control channels.

B. Results from duality and convex programming

From standard convex programming (see, e.g., Propositions 5.1.2 and 5.1.3 of [34]), we have the following:

Proposition 3.1: The dual function $L(\lambda, \mu)$ is convex over the set $\{(\lambda, \mu) \geq \mathbf{0}\}$ and

$$V^* \leq L(\lambda) \leq L(\lambda, \mu), \qquad \forall \lambda, \mu \geq 0.$$

From the concavity of V and convexity of the domain of optimization, it is easy to verify that Assumption 5.3.1 of [34] holds, and therefore, we have from Propositions 5.3.1, 5.1.4, and 5.1.5 in [34] that

Proposition 3.2: There exists at least one solution to the dual problem and there is no duality gap. Any optimal dual solution, (λ^*, μ^*) satisfies $V^* = L(\lambda^*, \mu^*)$. Furthermore, $((\mathbf{n}^*, \mathbf{p}^*), (\lambda^*, \mu^*))$ is a pair of optimal primal and optimal dual solutions if and only if

$$(\mathbf{n}^*, \mathbf{p}^*) \in \mathcal{X}, \sum_i n_i^* \leq N, \ \sum_i p_i^* \leq P \quad \text{Primal Feasibility} \tag{18}$$

$$(\lambda^*, \mu^*) \geq 0 \quad \text{Dual Feasibility} \tag{19}$$

$$(\mathbf{n}^*, \mathbf{p}^*) \in \arg \max_{(\mathbf{n},\mathbf{p}) \in \mathcal{X}} L(\mathbf{n}, \mathbf{p}, \lambda^*, \mu^*) \quad \text{Lagrangian Optimality} \tag{20}$$

$$\lambda^*(P - \sum_i p_i^*) = 0, \mu^*(N - \sum_i n_i^*) = 0 \quad \text{Complementary Slackness} \tag{21}$$

IV. Structure of the primal and dual problems

In this section, we give several properties of the dual problem in (16) and the corresponding primal problem in (10). First, we compute the dual function, $L(\lambda, \mu)$ in (15) for a given λ and μ. We then keep λ fixed and optimize the dual function over μ; this gives us $L(\lambda)$ in (17). We prove that $L(\lambda)$ is convex and provide bounds on the optimal λ. Using these properties, the optimal λ can be found with a one-dimensional convex search that has geometric convergence. We find primal variables ($\mathbf{n}$ and $\mathbf{p}$) that maximize the Lagrangian for a given λ and μ, and finding the optimal primal power allocation for a given $\mathbf{n}$.

A. Computing the dual function

To evaluate the dual function, we proceed in two steps. First, we optimize the Lagrangian (14) over $\mathbf{p}$, for a fixed λ, μ, and $\mathbf{n}$. We then optimize over $\mathbf{n}$ to obtain the value of the dual function. For the first step, we define the following two projections of the set $\mathcal{X}$: for a given $\mathbf{n}$, let $\mathcal{X}_n = \{\mathbf{n} \geq 0 : n_i \leq N_i, \forall i\}$ and let $\mathcal{X}_p(\mathbf{n}) = \{\mathbf{p} : (\mathbf{n}, \mathbf{p}) \in \mathcal{X}\}$. Then we have:

Lemma 4.1: For a fixed $\mathbf{n} \in \mathcal{X}_n$ and any $\lambda \geq 0$ and $\mu \geq 0$, the power allocation $\mathbf{p}^* \in \mathcal{X}_p(\mathbf{n})$ that maximizes $L(\mathbf{n}, \mathbf{p}, \lambda, \mu)$ is given by

$$p_i^* = \frac{n_i}{e_i} s^* \left(\frac{w_i e_i}{\lambda}, s_i(n_i), \check{s}_i(n_i) \right), \tag{22}$$

where

$$s^* \left(\frac{w_i e_i}{\lambda}, s_i(n_i), \check{s}_i(n_i) \right) := \max \left\{ \min \left\{ \left(\frac{w_i e_i}{\lambda} - 1 \right), s_i(n_i) \right\}, \check{s}_i(n_i) \right\}.$$

This lemma follows directly from the Kuhn-Tucker conditions for the optimization problem. Note that the "min" is not needed for Type II per user power constraints, i.e., $s_i(n) = \infty$. However, the maximum

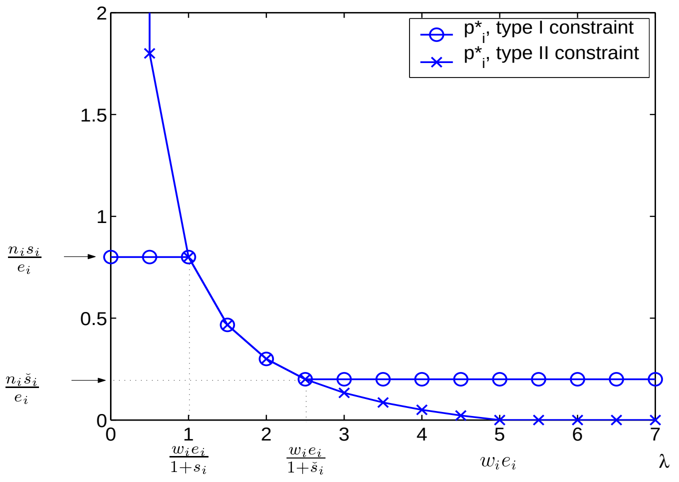

Fig. 1. An example of the optimal power allocation, p_i^* in (22) as a function of λ for both a Type I and type II power constraint.

is still necessary even if $\check{s}_i(n_i) = 0$, to restrict attention to non-negative power values. The solution can be viewed as a modified version of a water-filling power allocation across the users [32], where the "water-level" is modified to take into account each users weight, w_i, and the per-user power constraints are also taken into account. In the case of a Type I per-user power constraint ($s_i(n_i) \equiv s_i$ and $\check{s}_i(n_i) \equiv \check{s}_i$), the resulting SINR per code for a fixed λ, μ, and $\mathbf{n}$ is given by

$$\frac{p_i^* e_i}{n_i} = s^*\left(\frac{w_i e_i}{\lambda}, s_i(n_i), \check{s}_i(n_i)\right) = s^*\left(\frac{w_i e_i}{\lambda}, s_i, \check{s}_i\right), \tag{23}$$

which does not depend on the number of codes n_i. It follows that, in the Type I case, for a given λ the total power allocated to a user scales linearly in the number of codes.

An example of p_i^* as a function of λ is shown in Fig. 1 for both a Type I and Type II constraint. The horizontal segments of p_i^* under the Type II constraint correspond to when the maximum and minimum per user power constraints are active; when these are not active, the two curves overlap.

Substituting (22) into the Lagrangian we have

$$\begin{aligned} &L(\mathbf{n}, \mathbf{p}^*, \lambda, \mu) \\ &\quad = \sum_i w_i n_i \ln\left(1 + \frac{p_i^* e_i}{n_i}\right) \\ &\qquad + \lambda\left(P - \sum_i p_i^*\right) + \mu\left(N - \sum_i n_i\right) \end{aligned} \tag{24}$$

$$\begin{aligned} &\quad = \sum_i \left(w_i n_i h(w_i e_i, s_i(n_i), \check{s}_i(n_i), \lambda) - \mu n_i\right) \\ &\qquad + \lambda P + \mu N, \end{aligned} \tag{25}$$

where

$$\begin{aligned} &h(w_i e_i, s_i(n_i), \check{s}_i(n_i), \lambda) := \\ &\begin{cases} \ln(1 + \check{s}_i(n_i)) - \frac{\lambda}{w_i e_i}\check{s}_i(n_i), & \lambda \geq \frac{w_i e_i}{1+\check{s}_i(n_i)}, \\ \frac{\lambda}{w_i e_i} - 1 - \ln\frac{\lambda}{w_i e_i}, & \lambda \in [\frac{w_i e_i}{1+s_i(n_i)}, \frac{w_i e_i}{1+\check{s}_i(n_i)}), \\ \ln(1 + s_i(n_i)) - \frac{\lambda}{w_i e_i}s_i(n_i), & \lambda < \frac{w_i e_i}{1+s_i(n_i)}. \end{cases} \end{aligned} \tag{26}$$

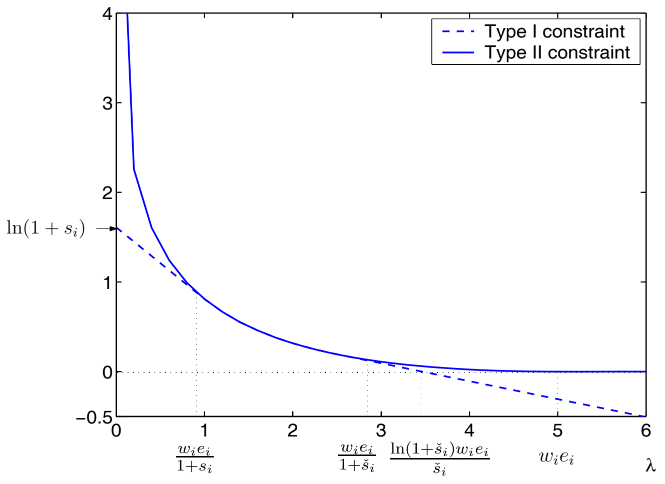

Fig. 2. An example of $h(w_i e_i, s_i, \check{s}_i, \lambda)$ as a function of λ under a Type I and Type II power constraint.

Notice that for a Type I per-user power constraint, $h(w_i e_i, s_i(n_i), \check{s}_i(n_i), \lambda) = h(w_i e_i, s_i, \check{s}_i, \lambda)$ also does not depend on n_i. For a Type II per-user power constraint,[11]

$$h(w_i e_i, s_i, \check{s}_i, \lambda) = \left[\frac{\lambda}{w_i e_i} - 1 - \ln\left(\frac{\lambda}{w_i e_i}\right)\right] 1_{\{w_i e_i > \lambda\}}.$$

An example of $h(w_i e_i, s_i, \check{s}_i, \lambda)$ as a function of λ is shown in Fig. 2 for both a Type I and Type II per-user power constraint. In both cases $w_i e_i = 5$. When $\frac{w_i e_i}{1+s_i} \le \lambda \le \frac{w_i e_i}{1+\check{s}_i}$ the two curves overlap. For $\lambda < \frac{w_i e_i}{1+s_i}$, h grows without bound under a Type II constraint, while it is linear in this range under a Type I constraint. For $\lambda > \frac{w_i e_i}{1+\check{s}_i}$, h decreases linearly under a Type II constraint, while under a Type I constraint it converges to 0 at $\lambda = w_i e_i$. For a Type II constraint, h crosses the x-axis at $\lambda = \frac{\ln(1+\check{s}_i) w_i e_i}{\check{s}_i}$. In either of these cases, since (25) is linear in $\mathbf{n}$, it is straightforward to optimize over $\mathbf{n}$.

Lemma 4.2: With a per-user power constraint of Type I or II, the vector of code allocations, $\mathbf{n}^*$, that maximizes (25) is given by

$$n_i^* = \begin{cases} 0, & \mu_i(\lambda) < \mu, \\ N_i, & \mu_i(\lambda) > \mu, \end{cases} \tag{27}$$

where

$$\mu_i(\lambda) = w_i h(w_i e_i, s_i, \check{s}_i, \lambda). \tag{28}$$

If $\mu = \mu_i(\lambda)$, every choice of n_i such that $0 \le n_i \le N_i$ maximizes the Lagrangian.

In other words, given μ, the optimal code allocation is determined for each user i by checking if $\mu_i(\lambda)$ is greater than or less than μ. The last part of this lemma follows because when $\mu = \mu_i(\lambda)$, (25) is not dependent on n_i. Using (27) we have[12]

$$w_i n_i^* \ln\left(1 + \frac{p_i^* e_i}{n_i^*}\right) - \lambda p_i^* - \mu n_i^* = [\mu_i(\lambda) - \mu]^+ N_i.$$

Substituting this into (25) yields the following characterization of the dual function $L(\lambda, \mu)$.

Lemma 4.3: With a Type I or II per-user power constraint,

$$L(\lambda, \mu) = \sum_i \left[\mu_i(\lambda) - \mu\right]^+ N_i + \mu N + \lambda P. \tag{29}$$

[11]The notation 1_X denotes the indicator function of the event X.
[12]We use the notation $[x]^+ = \max(x, 0)$.

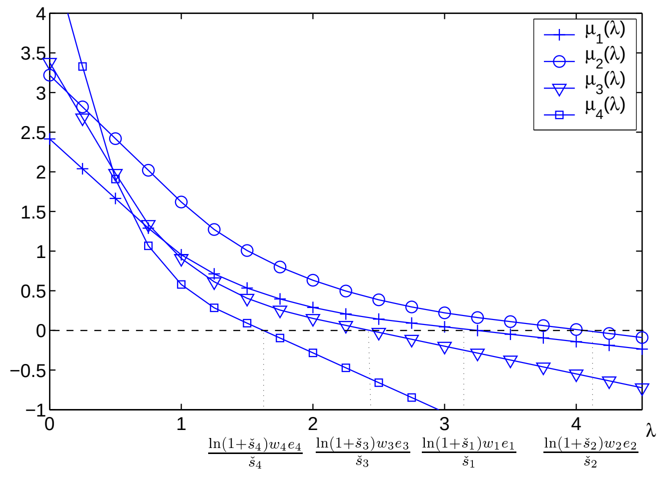

Fig. 3. An example of $\mu_i(\lambda)$ for a system with $K = 4$ users and a Type I per-user power constraint.

B. *Optimizing over* μ

We now turn to optimizing the dual function over μ. We restrict our attention to either a Type I or Type II per-user power constraint, so that the dual function is given by (29). To begin, we sort the users in decreasing order of $\mu_i(\lambda)$ in (28), where ties are broken arbitrarily. Assume that the users are numbered corresponding to their position in this ordering, i.e. so that $\mu_i(\lambda) \geq \mu_{i+1}(\lambda)$ for all i.[13]

Let $j^* - 1$ be the largest integer such that $\mu_{j^*-1}(\lambda) \geq 0$ and $\sum_{i=1}^{j^*-1} N_i < N$. If no such user can be found, set $j^* = 1$. Note that if $\check{s}_i = 0$ for all i, then $\mu_i(\lambda) \geq 0$ for all i, in which case j^* will be the first user that would fill up the total code budget if all users received their maximum per-user code allocation. By convention set $\mu_{K+1}(\lambda) = -1 - [\mu_K(\lambda)]^-$, where $[x]^- = [-x]^+$. Let $N'_{j^*} := N - \sum_{i=1}^{j^*-1} N_i$.

Lemma 4.4: With a Type I or Type II per-user power constraint,

$$\begin{aligned} L(\lambda) &:= \min_{\mu \geq 0} L(\lambda, \mu) \\ &= \sum_{i=1}^{j^*-1} \mu_i(\lambda) N_i + [\mu_{j^*}(\lambda)]^+ N'_{j^*} + \lambda P, \end{aligned} \tag{30}$$

and the minimizing μ is given by $\mu^*(\lambda) := [\mu_{j^*}(\lambda)]^+$.

Proof: For $\mu_i(\lambda) < \mu < \mu_{i-1}(\lambda)$, from (29) it can be seen that the derivative of $L(\lambda, \mu)$ in μ is given by $N - \sum_{j=1}^{i-1} N_i$. Hence, j^* is the largest integer for which $L(\lambda, \mu)$ will be increasing in the corresponding interval, i.e., $L(\lambda, \mu)$ will be increasing if and only if $\mu > \mu_{j^*}(\lambda)$. The lemma then follows. ■

From Lemma 4.2, μ is a threshold separating the users that get their full code allocation from the users that get allocated no codes. As μ is decreased, more users will be allocated their full code allocation. Lemma 4.4 shows that the threshold $\mu^*(\lambda)$ that minimizes the dual function is such that the full code budget is utilized.

Figure 3 shows an example of the curves $\mu_i(\lambda)$ as a function of λ for a system with $K = 4$ users, under a Type I per-user power constraint. Also indicated on the figure are the values of λ for which each curve $\mu_i(\lambda)$ crosses the x-axis. Consider the case where $N_i = N$ for all i. In this case, $j^* = 1$ (i.e. the user with the maximum value of $\mu_i(\lambda)$ for the given value of λ. Therefore, for $\lambda < \frac{\ln(1+\check{s}_2)w_2e_2}{\check{s}_2}$, $\mu^*(\lambda)$ will be the upper envelope of the curves shown in the figure. For $\lambda > \frac{\ln(1+\check{s}_2)w_2e_2}{\check{s}_2}$ all of the $\mu_i(\lambda)$ will be less than 0 and so $\mu^*(\lambda) = 0$.

[13]Of course, as λ changes this ordering will change, in which case we must re-number the users.

Remark: When $w_i \geq w_j$, $e_i > e_j$, and $s_i \geq s_j$ then it can be shown that $\mu_i(\lambda) \geq \mu_j(\lambda)$, for all λ. It follows that in this case, user i will be always be given a full code allocation before allocating any codes to user j. Furthermore, assume the scheduling rule is the "maximum throughput" version of (3), i.e. the case where $\alpha = 1$ and the class weights are all equal, so that the w_i's are constant and identical across users. In this case, (still assuming that if $e_i > e_j$ then $s_i \geq s_j$) packing users into the code budget in order of decreasing e_i's is optimal.

C. Finding a Lagrangian Optimal Primal Solution.

We next consider finding primal values $(\mathbf{n}^*, \mathbf{p}^*)$ such that

$$(\mathbf{n}^*, \mathbf{p}^*) = \arg \max_{(\mathbf{n},\mathbf{p}) \in \mathcal{X}} L(\mathbf{n}, \mathbf{p}, \lambda, \mu^*(\lambda)) \tag{31}$$

for a given $\lambda \geq 0$. Here, $\mu^*(\lambda)$ is the optimal μ given by Lemma 4.4. Given the optimal $\lambda = \lambda^*$, then from Proposition 3.2, such an $(\mathbf{n}^*, \mathbf{p}^*)$ will be an optimal solution for the primal problem if it also satisfies primal feasibility (18) and complimentary slackness (21). We give a procedure for selecting such a pair in the following. If the $\lambda \neq \lambda^*$, this procedure can also be used to find a candidate feasible $\tilde{\mathbf{n}}$. In the next section, we construct a feasible $\tilde{\mathbf{p}}$ corresponding to $\tilde{\mathbf{n}}$. From Proposition 3.1, we have [14]

$$V^* - V(\tilde{\mathbf{n}}, \tilde{\mathbf{p}}) \leq L(\lambda) - V(\tilde{\mathbf{n}}, \tilde{\mathbf{p}}).$$

We continue restricting our attention to Type I or II per-user power constraints.

From the results in Sections IV-A and IV-B, it can be seen that a solution to (31) is equivalent to finding

$$\mathbf{n}^* = \arg \max_{\{\mathbf{n} \in \mathcal{X}\}} \sum_i \left(\mu_i(\lambda) - \mu^*(\lambda)\right)_+ n_i, \tag{32}$$

and setting $\mathbf{p}^*$ as in Lemma 4.1.

As in the previous section, we again assume that the users are ordered in decreasing order of $\mu_i(\lambda)$ so that $\mu^*(\lambda) = \mu_{j^*}(\lambda)$. When[15] $\mu_{j^*-1}(\lambda) > \mu_{j^*}(\lambda) > \mu_{j^*+1}(\lambda)$ and $\mu_{j^*}(\lambda) \neq 0$, then there is a unique feasible $\mathbf{n}^*$ that optimizes (32) and satisfies $\mu^*(\lambda)(N - \sum n_i^*) = 0$. This is given by

$$n_i^* = \begin{cases} N_i, & i < j^*, \\ N'_{j^*}, & i = j^* \text{ and } \mu^*(\lambda) \neq 0, \\ 0, & i = j^* \text{ and } \mu^*(\lambda) = 0, \\ 0, & i > j^*. \end{cases} \tag{33}$$

Note that this solution will always satisfy $\sum n_i^* \leq N$, with equality if $\mu^*(\lambda) > 0$. Also note that n_i^* in (33) is always an integer code allocation.

Definition 4.1: A scalar $d \in \mathbb{R}$ is a *subgradient* of $L(\lambda)$ at λ if

$$L(\tilde{\lambda}) \geq L(\lambda) + (\tilde{\lambda} - \lambda)d, \ \forall \, \tilde{\lambda} \geq 0.$$

Proposition 4.1: Let $(\hat{\mathbf{n}}, \hat{\mathbf{p}})$ be a solution to (31) for a given λ which satisfies $\sum \hat{n}_i \leq N$, and $\mu^*(\lambda)(N - \sum \hat{n}_i) = 0$. Then $P - \sum_i \hat{p}_i$ is a subgradient of $L(\lambda)$ at λ.

[14]This can be used as a stopping criterion in a practical iterative algorithm.

[15]Recall that by convention $\mu_{K+1}(\lambda) = -1 - [\mu_K]^-$.

Proof: Using the definition of $\mu^*(\lambda)$ we have

$$\begin{aligned}
L(\tilde{\lambda}) &= L(\tilde{\lambda}, \mu^*(\tilde{\lambda})) \\
&= \max_{(\mathbf{n},\mathbf{p})\in\mathcal{X}} L(\mathbf{n}, \mathbf{p}, \tilde{\lambda}, \mu^*(\tilde{\lambda})) \\
&\geq L(\hat{\mathbf{n}}, \hat{\mathbf{p}}, \tilde{\lambda}, \mu^*(\tilde{\lambda})) \\
&= V(\hat{\mathbf{n}}, \hat{\mathbf{p}}) + \tilde{\lambda}(P - \sum_i \hat{p}_i) \\
&\qquad + \mu^*(\tilde{\lambda})(N - \sum_i \hat{n}_i) \\
&\geq V(\hat{\mathbf{n}}, \hat{\mathbf{p}}) + \tilde{\lambda}(P - \sum_i \hat{p}_i) \qquad (34) \\
&= V(\hat{\mathbf{n}}, \hat{\mathbf{p}}) + \lambda(P - \sum_i \hat{p}_i) \\
&\qquad + (\tilde{\lambda} - \lambda)(P - \sum_i \hat{p}_i) \\
&= L(\lambda) + (\tilde{\lambda} - \lambda)(P - \sum_i \hat{p}_i). \qquad (35)
\end{aligned}$$

The inequality in (34) follows because $N - \sum_i \hat{n}_i \geq 0$ and $\mu^*(\tilde{\lambda}) \geq 0$; equality in (35) holds because $\mu^*(\lambda)(N - \sum \hat{n}_i) = 0$. ■

Note that the code allocation given by (33) and the corresponding power allocation in Lemma 4.1 satisfy the assumptions of Proposition 4.1 and so provide a subgradient of $L(\lambda)$. Later in Corollary 4.1, we show that all subgradients of $L(\lambda)$ can be found in this way.

When there is a tie and more than one $\mu_j(\lambda) = \mu^*(\lambda)$, then there may be multiple $\mathbf{n}^*$ that optimize (32) and satisfy $\mu^*(\lambda)(N - \sum n_i^*) = 0$ and $\sum_i n_i^* \leq N$. There will also be multiple candidates for $\mathbf{n}^*$ if there is no tie, but $\mu_{j^*} = 0$.[16] However, for the optimal λ^*, every such $\mathbf{n}^*$ may not result in a power allocation that is feasible and satisfies complimentary slackness. For an arbitrary λ, different choices of $\mathbf{n}^*$ will result in different subgradients for $L(\lambda)$. Next, we examine resolving such ties. First, we show how to resolve these ties to find the maximum and minimum subgradients of $L(\lambda)$.[17]

Let there be $l \geq 0$ users with $i < j^*$ and $k \geq 1$ users with $i \geq j^*$ whose $\mu_i(\lambda)$ are tied with $\mu_{j^*}(\lambda)$, where $l + k \geq 1$, i.e.,[18]

$$\begin{aligned}
\mu_{j^*-l-1}(\lambda) > \mu_{j^*-l}(\lambda) &= \mu_{j^*}(\lambda) \\
&= \mu_{j^*+k-1}(\lambda) > \mu_{j^*+k}(\lambda).
\end{aligned}$$

Let $\mathcal{I}_\lambda = [j^* - l, j^* + k - 1]$ denote the set of these users. The objective in (32) will not depend on n_i, for $i \in \mathcal{I}_\lambda$. Note that the ordering of these users based on $\mu_i(\lambda)$ is arbitrary.

First we consider resolving this tie to find the maximum subgradient of $L(\lambda)$ at λ. It follows from

[16]It can be seen that if $\check{s}_i = 0$, then the case of $\mu_{j^*}(\lambda) = 0$ is trivial because user j^* will not receive any power regardless of its code allocation.

[17]That these are indeed the maximum and minimum follows from Corollary 4.1.

[18]The case where $l + k = 1$ captures the situation where there are no ties and $\mu_{j^*} = 0$.

Lemma 4.1 and Corollary 4.1 that this is the solution to the following linear program (LP):

$$\max_{\{n_i|i\in\mathcal{I}_\lambda\}} \quad P_{\rm res} - \sum_{i\in\mathcal{I}_\lambda} s^*\left(\frac{w_ie_i}{\lambda}, s_i, \check{s}_i\right)\frac{n_i}{e_i} \quad \textbf{[LPmax]}$$
$$\text{subject to: } 0 \le n_i \le N_i, \quad i\in\mathcal{I}_\lambda$$
$$\sum_{i\in\mathcal{I}_\lambda} n_i \le N_{\rm res},$$
$$\mu^*(\lambda)(N_{\rm res} - \sum_{i\in\mathcal{I}_\lambda} n_i) = 0.$$

Here, $P_{\rm res} := P - \sum_{i<j^*-l} s^*\left(\frac{w_ie_i}{\lambda}, s_i, \check{s}_i\right)\frac{N_i}{e_i}$ and $N_{\rm res} := N - \sum_{i<j^*-l} N_i$ are the residual power and codes available for the users in the tie. The minimum subgradient can also be found via a LP given by

$$\min_{\{n_i|i\in\mathcal{I}_\lambda\}} \; P_{\rm res} - \sum_{i\in\mathcal{I}_\lambda} s^*\left(\frac{w_ie_i}{\lambda}, s_i, \check{s}_i\right)\frac{n_i}{e_i}. \qquad \textbf{[LPmin]}$$

subject to the same constraints as in LPmax.

The structure of these linear programs permits a simple greedy solution. For LPmax, if $\mu^*(\lambda)=0$, then the solution to LPmax is clearly to assign $\hat{n}_i = 0$ for all $i\in\mathcal{I}_\lambda$. Otherwise, if $\mu^*(\lambda)>0$, order the users in $\mathcal{I}_\lambda$ in increasing order of $s^*\left(\frac{w_ie_i}{\lambda}, s_i, \check{s}_i\right)\frac{1}{e_i}$. Let $\hat{\Theta}:\mathcal{I}_\lambda\mapsto\mathcal{I}_\lambda$ be a permutation of $\mathcal{I}_\lambda$ according to this ordering, so that if $s^*\left(\frac{w_ie_i}{\lambda}, s_i, \check{s}_i\right)\frac{1}{e_i} < s^*\left(\frac{w_je_j}{\lambda}, s_j, \check{s}_j\right)\frac{1}{e_j}$, then $\hat{\Theta}(i)<\hat{\Theta}(j)$. For LPmin, we instead order the users in *decreasing* order of $s^*\left(\frac{w_ie_i}{\lambda}, s_i, \check{s}_i\right)\frac{1}{e_i}$ and denote this ordering by the permutation $\check{\Theta}$. Let $\hat{j}$ be the smallest integer such that $\sum_{i=j^*-l}^{\hat{j}} N_{\hat{\Theta}^{-1}(i)} \ge N_{\rm res}$; if no such integer exists, set $\hat{j} = j^*+k-1$. Let $\check{j}$ denote the corresponding integer using the $\check{\Theta}$ ordering. For $i\in\mathcal{I}_\lambda$, set

$$\hat{n}_i = \begin{cases} N_i, & \hat{\Theta}(i)<\hat{j}, \\ N_i', & \hat{\Theta}(i)=\hat{j}, \\ 0, & \hat{\Theta}(i)>\hat{j}, \end{cases} \tag{36}$$

where $N'_{\hat{\Theta}^{-1}(\hat{j})} = \min\{N_{\rm res} - \sum_{i=j^*-l}^{\hat{j}-1} N_{\hat{\Theta}^{-1}(i)}, N_{\hat{\Theta}^{-1}(\hat{j})}\}$. Let $\check{n}_i$ denote the corresponding code allocation using the $\check{\Theta}$ ordering.

Lemma 4.5: The code allocation $\hat{n}_i$ in (36) solves LPmax for $\mu^*(\lambda)>0$; the corresponding code allocation $\check{n}_i$ solves LPmin, for all values of $\mu^*(\lambda)$. When $\mu^*(\lambda)=0$, the solution to LPmax is $\hat{n}_i=0$ for all $i\in\mathcal{I}_\lambda$.

The proof of this lemma follows from a simple interchange argument. Finding both of these solutions involves a sort over the users involved in a tie, and thus each have a complexity of $O(|\mathcal{I}_\lambda|\log(|\mathcal{I}_\lambda|))$. Typically, if a tie occurs, only a small number of users will be involved. To gain some intuition as to why this is the case, note that ties occur whenever two or more of $\mu_i(\lambda)$ curves in Fig. 3 cross for a given value of λ. Moreover, it can be shown that any two such curves will only cross at one point. Hence, it follows that if the parameters w_i and e_i are independently chosen according to an absolutely continuous distribution, then with probability one a tie will not involve more than two users.

Given the solution to LPmax in (36), let

$$n_i^* = \begin{cases} N_i, & i<j^*-l, \\ \hat{n}_i, & j^*-l\le\hat{\Theta}(i)\le j^*+k-1, \\ 0, & i\ge j^*+k. \end{cases} \tag{37}$$

denote the corresponding complete code allocation. In two special cases, this will be a primal optimal code allocation.

Lemma 4.6: The pair $(\mathbf{n}^*, \mathbf{p}^*)$ given by (37) and (22) are a primal optimal solution if either

1) $\lambda = 0$ and LPmax has a non-negative solution,
2) The solution to LPmax is zero.

This lemma follows directly from noting that in both of these cases, the solution will satisfy both the complimentary slackness and primal feasibility conditions in Prop. 3.2. Note that when $\lambda = 0$, $s^*(\frac{w_i e_i}{\lambda}, s_i, \check{s}_i) = s_i$ for all i,[19] and thus the $\hat{\Theta}$-ordering corresponds to sorting the users based on $\frac{s_i}{e_i}$. A corresponding code allocation can be defined based on $\check{\Theta}$ and $\check{n}_i$; if this results in a solution to LPmin of zero, then it will also be primal optimal.

If the solution to LPmax is negative, then all the subgradients of $L(\lambda)$ at λ will be negative. Likewise, if the solution to LPmin is positive, then all the subgradients will be positive. However, if LPmax has a positive solution and LPmin has a negative one, then $L(\lambda)$ will have a zero subgradient at λ; a feasible code allocation corresponding to this zero subgradient will be primal optimal. In this case, there must exist an $\alpha \in [0, 1]$ such that

$$\begin{aligned} P_{\text{res}} = \alpha & \left(\sum_{i \in \mathcal{I}_t} s^* \left(\frac{w_i e_i}{\lambda}, s_i, \check{s}_i \right) \frac{\hat{n}_i}{e_i} \right) \\ & + (1-\alpha) \left(\sum_{i \in \mathcal{I}_t} s^* \left(\frac{w_i e_i}{\lambda}, s_i, \check{s}_i \right) \frac{\check{n}_i}{e_i} \right). \end{aligned}$$

Solving for α above, set

$$\tilde{n}_i = \alpha \hat{n}_i + (1-\alpha) \check{n}_i \tag{38}$$

for all $i \in \mathcal{I}_t$ and let $\mathbf{n}^*$ denote the corresponding complete code allocation as in (37).

Lemma 4.7: If the solution to LPmax is positive and the solution to LPmin is negative, then $\mathbf{n}^*$ constructed using (38) and the corresponding $\mathbf{p}^*$ are a primal optimal solution.

Once again, this follows from noting that by construction the code and power allocations satisfy the assumptions in Prop. 3.2. This gives a primal optimal solution; but depending on the number of users involved in the tie, it may not be the primal solution with the minimum number of users scheduled. As discussed in Sect. II-A, in practice there may be constraints on this number. The next lemma gives an upper bound on the minimum number of users scheduled in an optimal solution. Using typical parameter values for a HSDPA system, this bound will be no greater than 4.

Lemma 4.8: For a Type I or II power constraint, an optimal code allocation can always be found such that at most $\lceil N/N_{min} \rceil + 1$ users will be scheduled, where $N_{min} := \min_i N_i$.

Proof: At the optimal λ^*, if the conditions in Lemma 4.6 are satisfied then the code assignment in (37) is optimal and will result in no more than $\lceil N/N_{min} \rceil + 1$ users scheduled. Therefore, we need only consider the case where these conditions are not satisfied, i.e., $\lambda^* > 0$ and the solution to LPmax is strictly greater than 0.

When $\lambda^* > 0$, from complementary slackness and Prop. 4.1, a primal optimal code allocation must result in a zero subgradient of $L(\lambda)$. Such a code allocation is a solution to the following feasibility problem:

$$\begin{aligned} \text{maximize}_{\mathbf{n}} \; & 1 \\ \text{subject to: } & P - \sum_i n_i \frac{1}{e_i} s^* \left(\frac{w_i e_i}{\lambda^*}, s_i, \check{s}_i \right) = 0 \\ & \sum_i n_i = N \\ & 0 \leq n_i \leq N_i, \ \forall i. \end{aligned}$$

[19]This will arise only with a Type I power constraint.

This is a LP and the feasible set is a K dimensional bounded polyhedron.[20] By Lemma 4.7, this polyhedron is non-empty, i.e. the LP has a solution. However, the solution given in Lemma 4.7 may result in more than $\lceil N/N_{min} \rceil + 1$ users scheduled. In this case, we show that this LP must have another solution with the desired property. In particular, it must have an extreme point solution; we consider such an extreme point code allocation. At an extreme point, at least K constraints must be binding, two of which are the two equality constraints. This means that at least $K-2$ users must have n_i set equal to either 0 or N_i and so at most 2 users will have a fractional code assignment. First, assume N/N_{min} is an integer. If N/N_{min} users have $n_i = N_i$, then clearly to satisfy the second constraint, no other users can have positive code allocations. Likewise, if no more than $N/N_{min} - 1$ users have $n_i = N_i$, then from the above argument at most $N/N_{min} - 1 + 2 = N/N_{min} + 1$ users will have a positive code allocation. Similarly, if N/N_{min} is not an integer, then at most $\lceil N/N_{min} \rceil - 1$ users can have $n_i = N_i$ to satisfy the second equality, and so at most $\lceil N/N_{min} \rceil + 1$ users will have a positive code allocation. ■

Though in general (37) may result in more than $\lceil N/N_{min} \rceil + 1$ users being scheduled, in several key special cases this solution will also involve no more $\lceil N/N_{min} \rceil + 1$ users. This is useful in practice, since determining the solution in (37) is less complex than solving the LP in the proof of Lemma 4.8. [21]

Lemma 4.9: For a Type I or II power constraint, the code allocation in (37) results in no more than $\lceil N/N_{min} \rceil + 1$ users being scheduled in either of the following cases:

1) At most two users are involved in a tie;
2) For all users $i \in \mathcal{I}_\lambda$, $N_i \geq N_{\text{res}}$.

The second condition in this lemma implies that the per-user code constraints will be inactive for any solution to LPmax or LPmin. [22] In this case, the solution to LPmax and LPmin will involve one user each and the combination in (38) will involve only these two users.[23] Note that when $N_i = N$, this condition will always be satisfied.

Based on the above discussion, we outline a procedure for finding a primal feasible $\mathbf{n}^*$ given an arbitrary λ. This can be used to construct a feasible solution in a sub-optimal algorithm, which does not find the optimal λ.

Tie breaking rule:

1) Solve LPmax, if the solution is non-positive, or $\lambda = 0$, resolve the tie using $\hat{n}_i$.
2) Otherwise, solve LPmin,
    a) If the solution is negative use $\tilde{n}_i$ in (38) to resolve the tie,
    b) otherwise use $\check{n}_i$.

For a given λ, we denote by $\mathbf{n}^*(\lambda)$ the code allocation given by using this tie breaking rule. If the optimal choice of λ is used, $\mathbf{n}^*(\lambda)$ will be an optimal code allocation. Otherwise, it is the allocation that corresponds to the minimum positive subgradient (if all subgradients are positive) or the maximum negative subgradient (if all subgradients are negative).

D. Optimizing the power allocation

In this section, we consider the optimal primal power allocation, $\mathbf{p}$, given a fixed non-negative code allocation $\mathbf{n}$, i.e., we want to solve

$$\begin{aligned} V^*(\mathbf{n}) := \max_{\mathbf{p} \in \mathcal{X}_p(\mathbf{n})} \; & V(\mathbf{n}, \mathbf{p}) \\ \text{subject to:} \quad & \sum_i p_i \leq P. \end{aligned} \tag{39}$$

[20]Note, for convenience we formulate this LP as a function of all K users instead of just the $|\mathcal{I}_\lambda|$ users involved in the tie.

[21]Solving this involves listing all the extreme points and determining the one that works.

[22]In practical systems, this condition will often be satisfied. For example, in a HSDPA system with $N = 15$ and $N_i = 15$ or 10, then this condition will always be satisfied.

[23]If $\mu^*(\lambda) = 0$, then the solution of LPmax will involve zero users, and the combination in (38) will involve only one user.

This can be solved by finding $\lambda^*(\mathbf{n})$ using the dual formulation and then computing the optimal $\mathbf{p}^*(\mathbf{n})$ as in Lemma 4.1. We note that the results in this section are not restricted to Type I or Type II per user power constraints but will hold for any reasonable per-user constraints.[24] not just those discussed in Section II-A.

Without loss of generality, we remove any users with zero code allocations. Let M be the number of remaining users with positive code allocation, and assume these are numbered $i = 1, \dots, M$. We first need to check if the problem is infeasible, i.e., if

$$\sum_{i=1}^{M} p_i^{min} := \sum_i \frac{n_i}{e_i} \check{s}_i(n_i) \geq P.$$

If this is the case, then (39) will have no feasible solutions. We also check if the sum power constraint is inactive, i.e.,

$$\sum_{i=1}^{M} p_i^{max} := \sum_i \frac{n_i}{e_i} s_i(n_i) \leq P.$$

If this is the case, the optimal power allocation is simply $p_i^* = \frac{n_i}{e_i} s_i(n_i)$. Henceforth, we assume the problem is feasible and the power constraint is active. In this case, the sum power constraint must be satisfied with equality for the optimal powers, otherwise at least one of the powers can be increased resulting in a larger value of the objective function.

We can now construct a Lagrangian for (39) as

$$\begin{aligned} L_{\mathbf{n}}(\mathbf{p}, \lambda) := \sum_{i=1}^{M} w_i n_i \ln\left(1 + \frac{p_i e_i}{n_i}\right) \\ + \lambda\left(P - \sum_i p_i\right). \end{aligned} \tag{40}$$

Notice that if $\mu(N - \sum_i n_i) = 0$, $L_{\mathbf{n}}(\mathbf{p}, \lambda)$ will be equal to the original Lagrangian in (14). The dual function corresponding to (40) is given by

$$L_{\mathbf{n}}(\lambda) := \max_{\mathbf{p} \in \mathcal{X}_p(\mathbf{n})} L_{\mathbf{n}}(\mathbf{p}, \lambda). \tag{41}$$

Also, note that when optimizing over powers, the constraint set is always convex regardless of the function $s_i(n_i)n_i$. Maximizing $L_{\mathbf{n}}(\mathbf{p}, \lambda)$ over $\mathbf{p}$ is essentially the same as the problem for $L(\mathbf{p}, \mathbf{n}, \lambda, \mu)$ covered in Section IV-A. The optimal $\mathbf{p}$ is given by (22) as before. Substituting this into (41) yields

$$L_{\mathbf{n}}(\lambda) = \sum_{i=1}^{M} w_i n_i h(w_i e_i, s_i(n_i), \check{s}_i(n_i), \lambda) + \lambda P.$$

From basic convex optimization theory, we know that $L_{\mathbf{n}}(\lambda)$ is convex in λ. Furthermore, it can be shown that $L_{\mathbf{n}}(\lambda)$ is continuously differentiable in λ. To see this note that from (26), for each i,

$$\begin{aligned} \frac{d\, h(w_i e_i, s_i(n_i), \lambda)}{d\, \lambda} = \\ \begin{cases} -\frac{\check{s}_i(n_i)}{w_i e_i}, & \frac{w_i e_i}{1+\check{s}_i(n_i)} \leq \lambda, \\ \frac{1}{w_i e_i} - \frac{1}{\lambda}, & \frac{w_i e_i}{1+s_i(n_i)} \leq \lambda < \frac{w_i e_i}{1+\check{s}_i(n_i)}, \\ -\frac{s_i(n_i)}{w_i e_i}, & \lambda < \frac{w_i e_i}{1+s_i(n_i)}, \end{cases} \end{aligned} \tag{42}$$

[24]By reasonable constraints we refer to constraints such that $0 \leq \check{s}_i(n_i) \leq s_i(n_i)$.

which is continuous in the three intervals as well as at the two break points. This allows us to conclude that $L_{\mathbf{n}}(\lambda)$ is minimized by the set points at which the derivative is zero. Note that for each user i, (42) is constant in two of the three intervals; hence, it is possible that there are multiple points at which the derivative is zero. The following lemma gives an alternative characterization of the λ which minimizes $L_{\mathbf{n}}(\lambda)$. Let a_i and b_i be the two break points for each user $i = 1, \ldots, M$, i.e., $a_i := \frac{w_i e_i}{1+s_i(n_i)}$, and $b_i = \frac{w_i e_i}{1+\check{s}_i(n_i)}$.

Lemma 4.10: A $\lambda > 0$ is the solution to the dual problem $\min_{\lambda \geq 0} L_{\mathbf{n}}(\lambda)$ if and only if

$$\lambda = \frac{\sum_i n_i w_i 1_{[a_i,b_i)}(\lambda)}{P - \sum_i \frac{n_i}{e_i}\left(s_i(n_i)1_{[0,a_i)}(\lambda) - \check{s}_i(n_i)1_{[b_i,\infty)}(\lambda) + 1_{[a_i,b_i)}(\lambda)\right)}, \tag{43}$$

where, by convention, if numerator and denominator of the right-hand side are both zero, then we set this equal to λ.

Proof: Note that while the optimal λ^* may not be unique, the set of optimizers must form an interval by the convexity of $L_{\mathbf{n}}(\lambda)$. Since for any given λ, the $\mathbf{p}^*$ that maximizes the Lagrangian is unique, it follows from complementary slackness that $\lambda^* > 0$ is optimal if and only if the corresponding $\mathbf{p}^*$ satisfies $\sum_i p_i^* = P$. Substituting in p_i^* from (22) we have that $\lambda > 0$ is optimal if and only if

$$\begin{aligned} P = &\sum_i \frac{n_i}{e_i} s_i(n_i) 1_{[0,a_i)}(\lambda) + \sum_i \frac{n_i}{e_i} \check{s}_i(n_i) 1_{[b_i,\infty)}(\lambda) \\ &+ \sum_i \frac{n_i}{e_i}\left(\frac{w_i e_i}{\lambda} - 1\right) 1_{[a_i,b_i)}(\lambda). \end{aligned} \tag{44}$$

The desired result then follows from simple algebra. Note that if the right-hand side of (43) is $\frac{0}{0}$, then the first term on the left-hand side of (44) must be zero. This corresponds to all users either being assigned their maximum or minimum individual power, in such a way that the total power constraint is exactly met. Such a power allocation, will not depend on small variations in λ, provided that λ does not enter a new interval in (42) for some user.[25] ■

Let $\lambda^*(\mathbf{n})$ denote an optimal value of λ for a given code allocation, and let $p^*(\mathbf{n})$ denote the corresponding optimal power allocation given by (22). This lemma says that if $\lambda^*(\mathbf{n}) > 0$, it must satisfy (43). Next we show that a solution to this equation can be found in finite-time. Sort the set $\{a_i, b_i | i = 1, \ldots, M\}$ into a decreasing set of numbers $\{x[l]; l = 1, \ldots, 2M\}$, where ties are resolved arbitrarily. For $l = 1, \ldots, 2M$, let $P_{sum}[l]$ denote the total power $\sum_i p_i^*$ where p_i^* is given by (22) with $\lambda = x[l]$. Let l^* be the smallest value of l such that $P_{sum}[l] \geq P$. (Assuming that $\lambda^*(\mathbf{n}) > 0$ such an l^* must exist.)

Lemma 4.11: For a given $\mathbf{n}$, if the sum power constraint is active,[26] an optimal $\lambda^*(\mathbf{n})$ can be found in finite-time and is given by the right-hand side of (43) with $\lambda = x[l^*]$.

Proof: Note that as λ decreases, the right-hand side of (43) is right-continuous and only changes values when $\lambda = x[l], l = 1, \ldots, 2M$. (During any interval when the right-hand side is $\frac{0}{0}$, by our convention, the value changes continuously in λ; but this does not effect the following argument.) Hence, an optimal λ must be given by evaluating the right-hand side of (43) with $\lambda = x[l]$ for some $l = 1, \ldots, 2M$. Also, note that as λ decreases, the total power, $\sum_i p_i^*$ is increasing. By assumption the sum power constraint is active at the optimal solution. Thus, we have

$$x[l^* - 1] > \lambda^*(\mathbf{n}) \geq x[l^*].$$

Combining these observations, the lemma follows. ■

The idea behind this lemma is illustrated in Fig. 4, which shows an example where only two users have positive code allocations. The optimal power allocation for each user, p_i^* from (22) is shown as a function of λ, as well as the total power $p_1^* + p_2^*$. In this example, for a total power of P, $x[l^*] = a_1$, and the optimal λ can then be calculated using Lemma 4.10.

[25] Indeed, it follows that this is the only case in which the optimal λ^* is not unique.

[26] We make this assumption for simplicity of exposition. The algorithm can easily be modified to take into account the case where this constraint is not active and will still complete in finite time.

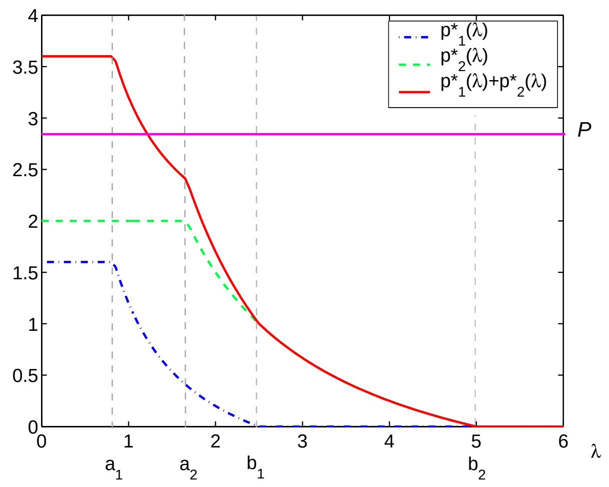

Fig. 4. Example illustrating Lemma 4.11.

Lemma 4.11 provides an algorithm for solving (43) by calculating $P_{sum}[l]$ starting with $l = 1$ and stopping when the total power constraint is violated. Also, note that with the above ordering, the right-hand side of (43) can be recursively calculated as l increases. The algorithm complexity is $O(M \log M)$ due to the sort of $\{x[l]\}$. Recall, M is the number of users with positive code allocations. As discussed after Lemma 4.9, this will typically be on the order of 1-4. Also, note that under a type II per-user power constraint, $a_i = 0$. Thus with no per-user power constraints, only the M values of $x[i]$ corresponding to the b_i's need to be considered in the above search, and a simpler algorithm results.

E. Optimizing the dual over λ

Recall, $L(\lambda)$ is the minimum of the dual function over $\mu \geq 0$. The solution to the dual problem, L^* is thus given by

$$L^* = \min_{\lambda \geq 0} L(\lambda).$$

We consider this problem and several characteristics of $L(\lambda)$ in the following. First we show that $L(\lambda)$ is convex in λ.[27]

Lemma 4.12: With a Type I or Type II per-user power constraint, $L(\lambda)$ is convex in λ.

Proof: From Lemma 4.4,

$$L(\lambda) = \sum_{i=1}^{j^*-1} \mu_i(\lambda) N_i + \left[\mu_{j^*}(\lambda)\right]^+ N'_{j^*} + \lambda P,$$

where the users are re-ordered according to $\mu_i(\lambda)$ for each λ. This can be re-written as:

$$\begin{aligned} L(\lambda) &= \max_{\mathbf{n} \in \mathcal{N}} \sum_i \mu_i(\lambda) n_i + \lambda P \\ &= \max_{\mathbf{n} \in \mathcal{N}} L_{\mathbf{n}}(\lambda), \end{aligned} \tag{45}$$

where,

$$\mathcal{N} = \left\{ \mathbf{n} : \sum_i n_i \leq N,\ 0 \leq n_i \leq N_i,\ \forall i \right\}.$$

[27]This lemma also follows from Prop. 4.1, since a function will only have a subgradient at every point if it is convex. Here we give an alternative proof that does not rely on subgradients.

We have already established in Sect. IV-D that for each $\mathbf{n}$, $L_{\mathbf{n}}(\lambda)$ is convex in λ. Since the maximum of a set of convex functions is also convex, it follows that $L(\lambda)$ is convex. ■

In (45), $L(\lambda)$ is expressed as the maximum of an infinite number of the functions $L_{\mathbf{n}}(\lambda)$. Next we show that in fact only a finite number of such functions are needed to characterize $L(\lambda)$, e.g.

$$L(\lambda) = \max_{\mathbf{n}\in\mathcal{N}_\Pi} L_{\mathbf{n}}(\lambda) \tag{46}$$

where $\mathcal{N}_\Pi$ is a finite subset of $\mathcal{N}$. Specifically, from Lemma 4.4, it follows that for each permutation of the users, we only need to consider a single greedy code allocations which uses all the codes, i.e. a code allocation as in (33) that sequentially assigns each user the maximum feasible number of codes until the code budget is full. We can then set $\mathcal{N}_\Pi$ to be the set of such code allocations, one for each permutation.

Now we turn to finding the optimal λ. From Lemma 4.12, this is the minimum of an univariate convex function, and so it can be found by using a one-dimensional convex search technique, such as the bisection method or a Fibonacci search [34]. Also note that, from (22) if $\lambda > \frac{\ln(1+\check{s}_i)}{\check{s}_i} w_i e_i$, then user i will be allocated zero power. Therefore the optimal λ^*, must satisfy

$$0 \le \lambda^* \le \max_i \frac{\ln(1+\check{s}_i)}{\check{s}_i} w_i e_i \le \max_i w_i e_i. \tag{47}$$

These bounds provide a starting point for the algorithms considered in the next section.

As noted in Section IV-D, $L_{\mathbf{n}}(\lambda)$ is continuously differentiable. From (46), we then have:

Lemma 4.13: With a Type I or II per user power constraint, $L(\lambda)$ is differentiable for all λ for which there exists a unique $\mathbf{n} \in \mathcal{N}_\Pi$, with $L_{\mathbf{n}}(\lambda) = L(\lambda)$.

When there is not a unique $n \in \mathcal{N}_\Pi$, this is exactly the tie case discussed in Section IV-C. This is illustrated in Fig. 5. Shown are three curves $L_{\mathbf{n}}(\lambda)$ corresponding to different code allocations; $L(\lambda)$ is the upper envelope of these curves which is shown in bold. $L(\lambda)$ is differentiable, except for at the two indicated places where a tie occurs. At the tie values, the derivatives of the $L_{\mathbf{n}}(\lambda)$ curves involved in the tie will be the corresponding subgradients discussed in Section IV-C. Indeed, as the next corollary shows, any subgradient of $L(\lambda)$ can be found in this way.

Corollary 4.1: Given any subgradient d of $L(\lambda)$ at λ, there exists primal values $(\hat{\mathbf{n}}, \hat{\mathbf{p}})$ that satisfy the assumptions of Proposition 4.1 so that $P - \sum_i \hat{p}_i = d$.

Proof: At any λ, if $L_{\mathbf{n}}(\lambda) = L(\lambda)$ for some $\mathbf{n} \in \mathcal{N}_\Pi$, then the primal values $(\mathbf{n}, \mathbf{p})$ which define $L_{\mathbf{n}}(\lambda)$ will satisfy the assumptions of Proposition 4.1 and give a subgradient of $L(\lambda)$ that corresponds to the derivative of $L_{\mathbf{n}}(\lambda)$ at λ.

If there is a unique $\mathbf{n} \in \mathcal{N}_\Pi$, with $L_{\mathbf{n}}(\lambda) = L(\lambda)$, then from Lemma 4.13, $L(\lambda)$ is differentiable and so has only one subgradient, which is given by the above.

Next consider the case where there are multiple $\mathbf{n} \in \mathcal{N}_\Pi$ such that $L_{\mathbf{n}}(\lambda) = L(\lambda)$. Since each $L_{\mathbf{n}}(\lambda)$ is continuously differentiable and convex and $L(\lambda)$ is the maximum of these, it follows that the maximum subgradient of $L(\lambda)$ must be given by the derivative of $L_{\mathbf{n}^+}(\lambda)$, where $\mathbf{n}^+$ is one of the $\mathbf{n}$ involved in the tie that satisfies $L(\lambda+\epsilon) = L_{\mathbf{n}^+}(\lambda+\epsilon)$ for small enough ϵ. Likewise, the minimum subgradient must be given by the derivative of $L_{\mathbf{n}^-}(\lambda)$, where $\mathbf{n}^-$ is one of the $\mathbf{n}$ involved in the tie that satisfies $L(\lambda-\epsilon) = L_{\mathbf{n}^-}(\lambda-\epsilon)$ for small enough ϵ. Any other subgradient can be found by considering a code allocation that is an appropriate convex combination of the maximum and minimum. ■

As λ decreases from the upper bound in (47), users receive a positive code allocation based on the ordering of $\frac{\ln(1+\check{s}_i)}{\check{s}_i} w_i e_i$. For large enough λ this ordering can determine the optimal code allocation. To be precise, for the remainder of this section, consider the case where $\check{s}_i = 0$ for all i. In this case, $\frac{\ln(1+\check{s}_i)}{\check{s}_i} w_i e_i = w_i e_i$ (by taking a limit as $\check{s}_i \to 0$). Assume the users are ordered in decreasing order of $w_i e_i$, in the case of a tie, order the users in decreasing order of w_i. If the w_i's are also tied, then order the users arbitrarily. Let Φ be a permutation of the users corresponding to this ordering. Using this permutation, let j^* denote the smallest value j such that

$$\sum_{i=1}^{j^*-1} N_{\Phi^{-1}(i)} < N \le \sum_{i=1}^{j^*} N_{\Phi^{-1}(i)}.$$

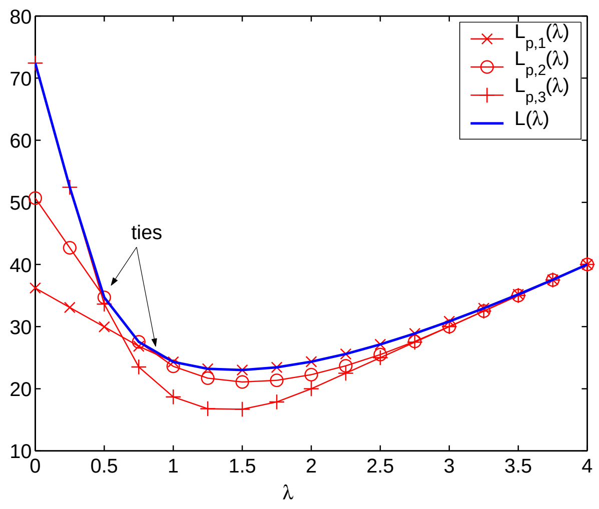

Fig. 5. An example of showing $L_{\mathbf{n}}(\lambda)$ versus λ for three different code allocations and the corresponding $L(\lambda)$.

Define the code allocation vector $\mathbf{n}_0$, where for each i,

$$n_{0,i} = \begin{cases} N_i, & \Phi(i) < j^*, \\ N - \sum_{i=1}^{j-1} N_i, & \Phi(i) = j^*, \\ 0, & \Phi(i) > j^*. \end{cases} \tag{48}$$

Lemma 4.14: Under a Type I or II per user power constraint with $\check{s}_i = 0$ for all i, the code allocation vector $\mathbf{n}_0$ is primal optimal if and only if

$$\begin{aligned} \frac{d\,L(\lambda)}{d\,\lambda} = P - \sum_i \frac{n_{0,i}}{e_i}(\frac{w_i e_i}{\lambda} - 1)1_{\{\frac{w_i e_i}{1+s_i(n_{0,i})} \le \lambda < w_i e_i\}} \\ - \sum_i \frac{n_{0,i}}{e_i} s_i 1_{\{\lambda < \frac{w_i e_i}{1+s_i}\}} \\ \le 0, \end{aligned}$$

for either

1) $\lambda = w_{\Phi^{-1}(j^*)} e_{\Phi^{-1}(j^*)}$ when $n_{0,\Phi^{-1}(j^*)} < N_{\Phi^{-1}(j^*)}$, or
2) $\lambda = w_{\Phi^{-1}(j^*+1)} e_{\Phi^{-1}(j^*+1)}$ when $n_{0,\Phi^{-1}(j^*)} = N_{\Phi^{-1}(j^*)}$.

Proof: When $\lambda \ge w_{\Phi^{-1}(j^*)} e_{\Phi^{-1}(j^*)}$, only those users with $\Phi(i) < j^*$ will have non-zero values of $\mu_i(\lambda)$. Hence for this case, $\mathbf{n}_0$ must be an optimal solution to (45). It can also be seen that $\mathbf{n}_0$ must be an optimal solution to (45) if $n_{0,\Phi^{-1}(j^*)} = N_{\Phi^{-1}(j^*)}$ and $\lambda \ge w_{\Phi^{-1}(j^*+1)} e_{\Phi^{-1}(j^*+1)}$. In either case,

$$L(\lambda) = \sum_i w_i h(w_i e_i, s_i, \lambda) n_{0,i} + \lambda P = L_{\mathbf{n}_0}(\lambda).$$

Differentiating this we have,[28]

$$\begin{aligned} \frac{d\,L(\lambda)}{d\,\lambda} = P - \sum_i \frac{n_{0,i}}{e_i} s_i 1_{\{\lambda < \frac{w_i e_i}{1+s_i}\}} \\ - \sum_i \frac{n_{0,i}}{e_i}(\frac{w_i e_i}{\lambda} - 1)1_{\{\frac{w_i e_i}{1+s_i(n_{0,i})} \le \lambda < w_i e_i\}}. \end{aligned} \tag{49}$$

[28]For simplicity, we assume that at λ a tie does not occur and so $L(\lambda)$ is differentiable. If this is not the case, the lemma is still true, except that (49) will be a subgradient of $L(\lambda)$

Since $L(\lambda)$ is convex, $\frac{d\,L(\lambda)}{d\,\lambda} \leq 0$ at $\lambda = \tilde{\lambda}$ if and only if $\lambda^* \geq \tilde{\lambda}$. Thus the condition in the lemma is both necessary and sufficient for $\mathbf{n}_0$ to be optimal. ■

The conditions in Lemma 4.14 are easily computable, and can help with the search for the optimal allocation. We will discuss this more in the next section.

It also follows from (49) that for $\lambda \geq w_{\Phi^{-1}(1)} e_{\Phi^{-1}(1)}$,

$$\left.\frac{d\,L(\lambda)}{d\,\lambda}\right|_{\lambda > w_{\Phi^{-1}(1)} e_{\Phi^{-1}(1)}} = P > 0.$$

This verifies that $\lambda^* < \max_i w_i e_i$, and using convexity provides another proof that if λ^* is greater than 0, then it occurs at a point where $L(\lambda)$ has a zero subgradient.

V. Algorithms

We next discuss algorithms for solving the primal problem (10). First, we present a family of optimal algorithms all with a geometric convergence rate. Several variations of these algorithms are discussed. Following this we give a class of *truncated* suboptimal algorithms which have a lower complexity. Finally, we give a family of baseline greedy algorithms that are based on splitting the scheduling and resource allocation decision into two parts.

A. Optimal Algorithm

The optimal algorithms we consider are all based on finding the dual optimal solution, L^* in (16), by solving

$$\min_{\lambda \geq 0} L(\lambda),$$

where $L(\lambda)$ is defined in (17). By strong duality this gives us the optimal primal value, V^*, and, given the dual optimal (λ^*, μ^*), the primal optimal $(\mathbf{p}^*, \mathbf{n}^*)$ are given by optimizing the Lagrangian as discussed in Sect. IV-C.

For Type I and II per-user power constraints, $L(\lambda)$ is given by Lemma 4.4. As shown in Section IV-E, this is a univariate convex function, and thus can be minimized using a convex search technique. Here we consider a bisection method, where at the kth iteration, the algorithm identifies a range $[\lambda_k^{LB}, \lambda_k^{UB}]$ known to contain the optimal λ^*. We also identify an estimate of λ^* given by $\lambda_k \in [\lambda_k^{LB}, \lambda_k^{UB}]$. These parameters are updated from iteration to iteration, by considering a candidate λ_k^{cand} in either $[\lambda_k^{LB}, \lambda_k]$ or $[\lambda_k, \lambda_k^{UB}]$, and then updating these parameters, depending on the relative values of $L(\lambda)$. Choosing λ_k^{cand} as the midpoint of the larger sub-interval ensures geometric convergence to the optimal dual solution. Note that each iteration of such an algorithm requires evaluating $L(\lambda)$. This can be done using Lemma 4.4, which has a complexity of $O(K \log(K))$ due to the required sort based on $\mu_i(\lambda)$. Also, note that as shown in Sect. IV-E, $\lambda^* < \max_i w_i e_i$; thus we can use the points $\lambda_{min} = 0$ and $\lambda_{max} = \max_i w_i e_i$ to begin our search. We have just described one optimal algorithm, which we will refer to as a pure bisection algorithm. In the following we discuss several enhancements to this algorithm, which further exploit the structure of the problem.

The first enhancement we consider is based on first checking if the code allocation vector $\mathbf{n}_0$ in (48) is optimal. As shown in Lemma 4.14, this can be easily done. If this code allocation is optimal, then we need simply calculate the optimal primal power allocation, $\mathbf{p}^*(\mathbf{n}_0)$, as in Section IV-D. and we are done. If $\mathbf{n}_0$ is not optimal, then λ^* must be less than $w_{\Phi^{-1}(j^*)} e_{\Phi^{-1}(j^*)}$, where j^* is as given in Lemma 4.14.[29]. Thus, instead of λ_{max}, we can use $w_{\Phi^{-1}(j^*)} e_{\Phi^{-1}(j^*)}$ as an upper-bound for beginning our search. Notice that calculating $\mathbf{n}_0$ requires a sort to generate the Φ ordering and so has a complexity of $O(K \log K)$. If

[29]More over, if $n_{0,\Phi^{-1}(j^*)} = N_{\Phi^{-1}(j^*)}$, then we have $\lambda^* < w_{\Phi^{-1}(j^*+1)} e_{\Phi^{-1}(j^*+1)}$

$\mathbf{n}_0$ is optimal, finding the optimal power allocation also requires a sort over the M users with non-zero code allocation, which has a complexity of $O(M \log M)$ [30].

The next enhancement we consider is to evaluate a feasible primal solution $\mathbf{n}_k = \mathbf{n}^*(\lambda_k)$ as in Section IV-C, for each iteration k. This serves two purposes which are as follows:

1) **Stopping Criterion**: This can be used for a stopping criteria. We give two possibilities here:
   a.) Calculate a primal feasible $\mathbf{p}_k = \mathbf{p}^*(\mathbf{n}_k)$, as in Section IV-D. Stop when the primal value and the dual value are sufficiently close, i.e.,

$$V(\mathbf{n}_k, \mathbf{p}^*(\mathbf{n}_k)) < (1-\epsilon)L(\lambda_k).$$

   Note that we need a sort operation in the optimal power calculation leading to additional complexity.

   b.) Calculate a power allocation $\mathbf{p}_k$ as given by Lemma 4.1. Stop when

$$|P - \sum_i p_{i,k}| < \epsilon.$$

   From Prop. 4.1, $P - \sum_i p_{i,k}$ a subgradient of $L(\lambda)$ at λ_k; thus, the stopping criteria checks if the subgradient is near zero.[31] Note that $\mathbf{p}_k$ is different from $\mathbf{p}^*(\mathbf{n}_k)$.

   Note that we have two different methods of obtaining a power vector $\mathbf{p}_k$ associated with the different stopping criteria.

2) **Update** λ_k: The second use of calculating $\mathbf{n}_k$ is to use this as a guide for picking the next λ_k. Once again there are several possibilities; we give two that correspond to the cases (a.) and (b.) above.
   a.) For case (a.), we consider the candidate

$$\lambda_k^{cand} = \lambda^*(\mathbf{n}_k) = \lambda^*(\mathbf{n}^*(\lambda_k)) =: \mathcal{T}(\lambda_k), \tag{50}$$

   where $\lambda^*(\mathbf{n})$ is given by Lemma 4.11. Note that any fixed point of the map $\mathcal{T}$ will correspond to an optimal λ^*. If λ_k^{cand} lies in the interval $[\lambda_k^{LB}, \lambda_k^{UB}]$, we can consider it instead of the bisection point of a sub-interval.[32] Note that evaluating this map using the iteration in Lemma 4.11 again has a complexity of $O(M \log M)$.

   b.) For case (b.), we can use the subgradient $d_k = P - \sum_i p_{i,k}$ to aid in choosing the next candidate λ. In particular, if $d_k < 0$ then the optimal λ must lie in $[\lambda_k, \lambda_k^{UB}]$, and if $d_k > 0$ then the optimal λ must lie in $[\lambda_k^{LB}, \lambda_k]$. We can make λ_k the mid-point of the appropriate interval, or we could "move in the subgradient direction using an appropriate step-size rule" [34].

Combining the above steps, we have an optimal algorithm with the basic structure shown in Fig. 6. The stopping criterion check and updating steps can be performed in either of the two ways discussed above.

B. Truncated Optimal Algorithm

As shown in Section IV-E, optimizing the Lagrangian for a given λ reduces to simply sorting the users based on the metric $\mu_i(\lambda)$ and then packing the code budget. Searching for the optimal λ can then be thought of as trying to find the optimal permutation of the users. Based on this, we next describe a *truncated optimal* algorithm with a structure as shown in Fig. 7. This algorithm begins by generating a set of J initial permutations based on various heuristic sort metrics, and then packing the code budget according to orderings. Some possible metrics include

[30]The Φ ordering can be used in the power allocation to accelerate the algorithm.

[31]As noted in Sect. IV-D, when $\mathbf{n}_0$ is not optimal, then $L(\lambda)$ having a zero subgradient at λ^* is both necessary and sufficient for λ^* to be optimal.

[32]Geometric convergence can still be guaranteed by only considering λ_k^{cand} is chosen such that it is sufficiently in the interior $[\lambda_k^{LB}, \lambda_k^{UB}]$ so the current interval will be reduced by a given percentage.

```
1) IF $\mathbf{n}_0$ is optimal, THEN STOP.
2) Initialize $\lambda_0^{LB}, \lambda_0^{UB}, \lambda_0$.
3) Set $k = 0$, $\mathbf{n}_k = \mathbf{n}^*(\lambda_k)$, and Choose $\mathbf{p}_k$.
4) WHILE Stopping Criterion fails DO
     i   k = k +1;
     ii  Update $\lambda_k$.
     iii Update $\lambda_k^{LB}$ and $\lambda_k^{UB}$.
     iv  Calculate $\mathbf{n}_k = \mathbf{n}^*(\lambda_k)$.
5) END WHILE
```

Fig. 6. Basic structure of optimal algorithm.

```
1) Generate initial code allocations, $\mathbf{n}^x$, $x = 1, \ldots, J$.
2) For each $x$, calculate $\lambda^*(\mathbf{n}^x)$, and $V(\mathbf{n}^x, \mathbf{p}^*(\mathbf{n}^x))$.
3) SET $\mathbf{n}^* = \arg\max_{\mathbf{n}^x} V(\mathbf{n}^x, \mathbf{p}^*(\mathbf{n}^x))$.
4) SET $\lambda^{J+1} = \lambda^*(\mathbf{n}^*)$.
5) Calculate $\mathbf{n}^{J+1} = \mathbf{n}^*(\lambda^{J+1})$, $V(\mathbf{n}^{J+1}, \mathbf{p}^*(\mathbf{n}^{J+1}))$.
6) SET $\mathbf{n}^* = \arg\max\{V(\mathbf{n}^*, \mathbf{p}^*(\mathbf{n}^*)), V(\mathbf{n}^{J+1}, \mathbf{p}^*(\mathbf{n}^{J+1}))\}$.
```

Fig. 7. Basic structure of the truncated algorithm.

i.) decreasing order of $w_i e_i$, i.e., using the Φ ordering;
ii.) decreasing order of $w_i N_i \left(\ln \left(1 + \frac{P_i e_i}{N_i} \wedge s_i(N_i) \right) \right)$;
iii.) decreasing order of $w_i N \log \left(1 + \frac{P e_i}{N} \right)$.

The first sort is the same as Φ which used in the optimal algorithm to calculate $\mathbf{n}_0$. Although we do not show this step in Fig. 7, we should again check if this code allocation is optimal using Lemma 4.14, and if so terminate the algorithm. The second and third sort metrics correspond to finding the maximum rate for each user assuming it was the only user in the system (TDM case) as is traditionally done for the user sort metric in a split scheduling and resource allocation algorithm. The two variations correspond to whether we include only the system constraints (iii) or also the per user constraints in calculating the maximum rate (ii). Given the set of initial feasible code allocations, we then select the allocation with the maximum primal value. This allocation is then updated as in the optimal algorithm in Fig. 6, where the updating is done using the transformation in (50).

For the truncated optimal algorithm, in all we consider $J + 1$ possible allocations and select the best from amongst those. The number of allocations can be chosen to trade-off complexity with performance. The simplest solution will be to just pick one of these; in this case the complexity is $O(M \log M)$ for the resource allocation, in addition to the $O(MK)$ for picking the top M in the "sort" traditionally done for scheduling. In both the optimal and truncated optimal we do not really have to do a full sort over all K eligible users. We at most need to pick the top M users, where M is the maximum number of users that can be scheduled in a time-slot. So everywhere we mention a sort over users, we really mean picking at most the top M users in the order of the suggested "sort metric".

Several variations of these algorithms can also be considered. This includes:

1) In both the optimal and truncated optimal we could initially pick the top m ($2M \leq m \leq K$) users using one of the sort metrics and then do all of the subsequent optimizations over those m users instead of all K users. This will reduce the complexity of finding the maximum. For instance, if we take $m = 2M$, the complexity of each iteration in the optimal algorithm is $O(m \log m) = O(M \log M)$.
2) If after Step 1 in the truncated algorithm, we obtain a code allocation $\mathbf{n}$ that fully packs the code budget but not the power budget, i.e., $\sum_i n_i = N$ but $\sum_i n_i s_i(n_i)/e_i < P$, then we should reduce

one or more of the n_is and consider allocating these to the user next as per the "sort metric" in consideration. For instance, we could pick the user with the smallest $s_i(n_i)$ and decrease its n_i till the power budget is packed. Other schemes are also possible.

3) In a practical system a final "polish up" step can be added at the end of the algorithms to take into account other constraints such as projecting the code allocation onto an integer solution and including any per-user rate constraints and re-optimizing over the power allocation.

C. Greedy Baseline Algorithm

In this section we describe a class of baseline greedy algorithms. These algorithms are based on splitting the scheduling decision and the resource allocation into two parts. First a scheduling order for the users is found. This can be done by ordering the users according to a given metric such as those in the previous section. Given the scheduling order, the resource allocation is then done by taking each user in order and choosing a PLOP that maximizes the transmission rate the user can receive, using the residual power and codes that are available. The main steps of the algorithm are the following:

1) Sort the users according to some metric (e.g., any of the metrics in Section V-B).
2) Set $i = 1$, $P_{\text{res}} = P$ and $N_{\text{res}} = N$ where P_{res} and N_{res} denote the residual power and code resources at every stage.
3) Find the maximum rate that is feasible for user i with $p_i \leq P_{\text{res}}$ and $n_i \leq N_{\text{res}}$.
4) If there is a unique PLOP (n_i, p_i) that achieves the maximum rate, then we are done.
5) In case of multiple PLOPs achieving the maximum rate, we maximize $f((P_{\text{res}} - p_i), (N_{\text{res}} - n_i))$. An example of f is $f(p, n) = \lambda p + \mu n$, in which case maximizing f is equivalent to minimizing $\lambda p_i + \mu n_i$.
6) Reduce P_{res} by p_i and N_{res} by n_i, respectively.
7) If $P_{\text{res}} > 0$, $N_{\text{res}} > 0$ and i is not the last user, set $i = i + 1$ and repeat from Step 3. If any of the checks fails, then exit.

It can be shown that in case the amount of data a user can transmit is not a constraint (i.e. there is no maximum rate constraint), the PLOP that maximizes the rate is unique. In the case where the amount of data is a constraint, the PLOP that maximizes f can easily be solved for analytically in the case of $\mu = 0$, i.e., we are interested in a minimum power solution. More generally, the solution can either be obtained by a search or by a table lookup. Since the search is for a convex function, it takes $\log N$ steps. A table look up or analytic formula is $O(1)$. So assuming we use an analytic solution or a table look up, the complexity for each of the steps is $O(1)$ and the complexity of the entire resource allocation algorithm is $O(M)$ (this does not include the "sorting" operation).

We provide the details next; here we include all of the per-user constraints discussed in Section II-A. Let i be the user selected in the ith stage of the algorithm with residual power P_{res} and residual codes N_{res}. The resource constraints for this user can be calculated as follows:

$$\begin{aligned} N_i' &= \min(N_i, N_{\text{res}}) \\ S_i' &= \min\left(S_i, \left(e^{(\frac{R}{N})_i} - 1\right)\right) \\ P_i' &= \min\left(P_i, P_{\text{res}}, \frac{S_i' N_i'}{e_i}\right) \end{aligned}$$

where N_i, P_i, S_i, and $(\frac{R}{N})_i$ are as defined in Section II-A. Based on these the highest rate user i can get is

$$R_i' = N_i' \ln\left(1 + \frac{P_i' e_i}{N_i'}\right).$$

So, if the users maximum rate constraint is R_i and $R_i \geq R_i'$, then by choosing $n_i = N_i'$, $p_i = P_i'$, we are done. In this is not the case, we set

$$p_i(n_i) = \frac{n_i}{e_i}\left(e^{\frac{R_i}{n_i}} - 1\right),$$

and minimize $\lambda p_i(n_i) + \mu n_i$ subject to the above "trade-off" between power and codes while satisfying the constraints:

$$\begin{aligned} n_i &\leq N_i', \\ \frac{e_i p_i(n_i)}{n_i} &\leq S_i', \\ p_i(n_i) &\leq P_i'. \end{aligned}$$

It can be seen that for $\mu = 0$, the solution to this is to set $n_i = N_i'$; thus, in this case the problem has complexity $O(1)$. Also, for any $\lambda \geq 0$ and $\mu \geq 0$, the cost function is convex in n_i and so the problem can be solved with a "bisection type" search with complexity $O(\log N)$. As noted above, the optimal n_i can also be obtained by a table lookup in (R_i, e_i), which is also $O(1)$.

VI. Simulation Results

We provide simulation results for the optimal and sub-optimal algorithms discussed above. Specifically, we consider

1) The optimal algorithm from Section V-A. However, for the simulation we modified the algorithm by projecting to integral code assignments. We expect this solution to be very close to the real optimum.
2) The truncated optimal algorithm from Section V-B with $J = 6$ initial code allocations; three of the initial code allocations correspond to the 3 examples given in Section V-B, the other three correspond to the code allocations obtained by applying the map $\mathcal{T}$ from (50) to each of these initial code allocations. In our simulations we omitted steps (4.) and (5.) of the algorithm.
3) The greedy baseline algorithm from Section V-C. We sort the users using the third sort metric from Section V-B and set $\mu = 0$ (i.e, we maximize the residual power) so that the algorithm has complexity $O(M)$.

We simulate each of these algorithms for a single cell system with K=40 users and with parameters chosen to match a HSDPA system. In particular we set $N = 15$, $N_i = 5$, $P = 11.9$W, $\check{s}_i = 0$ and $s_i = 1.59$. We assign each user a utility with the form given in (2); for a given simulation all the users have identical QoS weights (c_i) and fairness parameters (α). We simulate the combined scheduling and resource allocation for a single cell model that includes both large-scale and small scale fading. In particular, to model location-based attenuation and shadowing, each user receives and average SINR according to a distribution that is based upon measurements seen in more complex and realistic simulators. This is then modulated with a Rayleigh variable with the Clarke spectrum to yield a time-varying SINR representative of the variations mobiles encounter in real systems. Since we are assuming that one slot duration is long enough for information-theoretic analysis, we do not model transmission errors and retransmissions.

In Table 1, we give several performance metrics for each algorithm and for different choices of the fairness parameter α. Shown are:

- **Utility:** We calculate the time average utility given by $\frac{1}{T-K}\sum_{t=K+1}^{T} U(\mathbf{W}_t)$.
- **Log Utility:** We calculate the time average log utility given by $\frac{1}{T-K}\sum_{t=K+1}^{T} \ln(\mathbf{W}_t)$. We use this metric to compare the long-term throughputs achieved for different utility functions.
- **Number Scheduled (M):** The average number of users scheduled per time-slot.
- **Total Codes (N_s):** The average total number of codes used by all users in the sector ($N_s := \sum_{i=1}^{K} \frac{1}{T}\sum_{t=1}^{T} n_{i,t}$).
- **Sum Power (P_s):** The average sum power over all users in the sector ($P_s := \sum_{i=1}^{K} \frac{1}{T}\sum_{t=1}^{T} p_{i,t}$).
- **Sector Throughput:** We calculate the sum throughput over all users in the sector given by $\frac{1}{K}\sum_{i=1}^{K} \frac{1}{T}\sum_{t=1}^{T} r_{i,t}$.

Each quantity is averaged over 20 Monte Carlo drops. Also, in Figure 8, we show the empirical CDF of the user throughput for each algorithm in the $\alpha = 0$ case.

TABLE I
SIMULATION RESULTS

α	Algorithm	Utility	Log Utility	M	N_s	P_s	Sector Throughput (Mbps)
0.0	Optimal	231.944	231.944	3.35461	15	11.8997	8.8145
0.0	Truncated optimal	229.282	229.282	3	15	11.2689	7.87875
0.0	Greedy baseline	222.222	222.222	3	15	10.9659	6.36075
0.25	Optimal	173.646	231.669	3.33331	15	11.8998	9.28545
0.25	Truncated optimal	170.275	228.886	3	15	10.7793	8.54505
0.25	Greedy baseline	163.798	222.663	3	15	10.6948	7.2903
0.5	Optimal	806.085	228.404	3.36408	15	11.899	11.1392
0.5	Truncated optimal	749.531	224.379	3	15	9.83421	9.127
0.5	Greedy baseline	725.4	220.801	3	15	9.72985	8.6008
0.75	Optimal	4129.16	213.411	3.36341	15	11.8903	12.6934
0.75	Truncated optimal	3579.71	207.866	3	15	7.82554	10.1799
0.75	Greedy baseline	3538.96	201.87	3	15	7.79743	10.2524

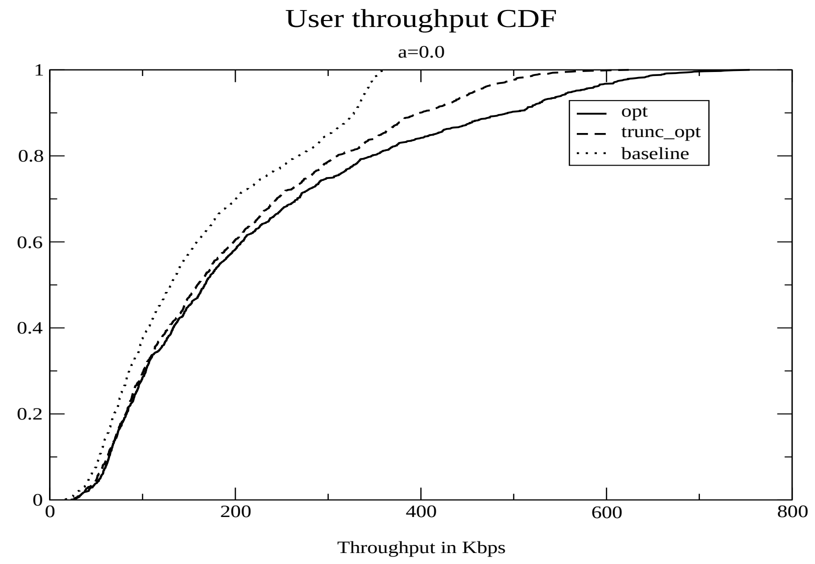

Fig. 8. Empirical CDF of users throughputs for $\alpha = 0$.

In these results, the optimal algorithm gives a higher utility as well as a higher sector throughput compared to the other algorithms. For the $\alpha = 0$ case (proportional fair) we get a 34% improvement over the greedy baseline algorithm. The truncated optimal algorithm is close to optimal and usually also gives a higher sector throughput than the greedy baseline algorithm. For $\alpha = 0$, we get a 23.87% improvement over the greedy baseline algorithm. Furthermore, not only is sector throughput higher for the optimal algorithm, but in fact, from Fig. 8 we see that all user throughputs are larger (in a stochastic ordering sense). In Figures 9, 10 and 11 we plot the user throughput distributions for other utility functions parameterized by $\alpha = 0.25,\ 0.5,$ and 0.75. In general, the optimal is better than truncated optimal which, in turn, is better than the greedy baseline when we compare user throughputs. In Figure 12 concentrating

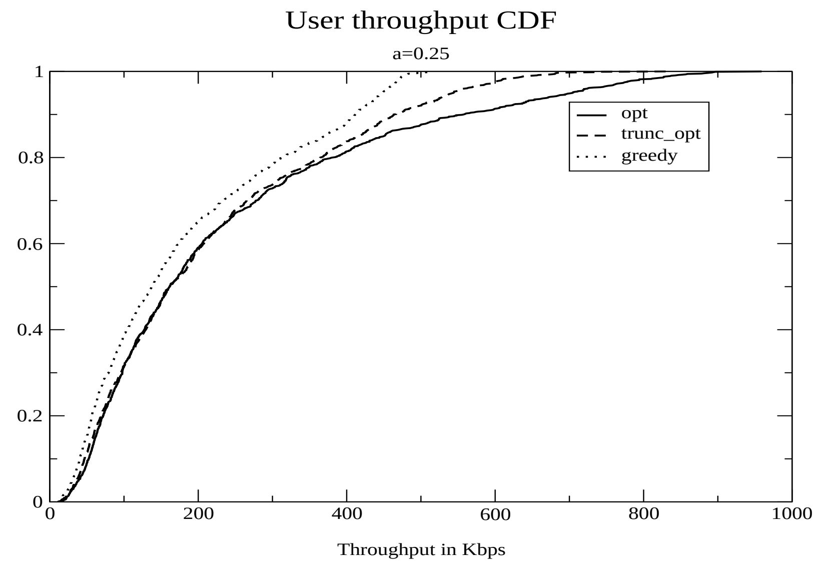

Fig. 9. Empirical CDF of users throughputs for $\alpha = 0.25$.

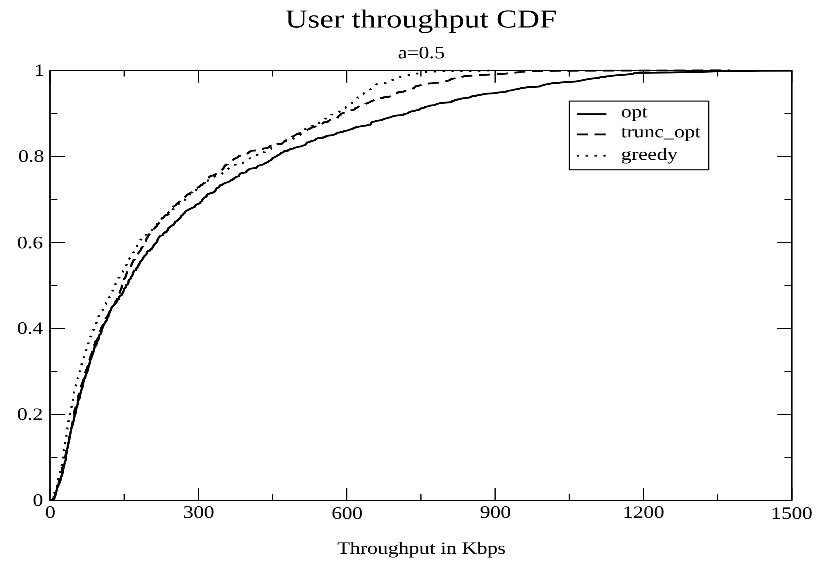

Fig. 10. Empirical CDF of users throughputs for $\alpha = 0.5$.

on the optimal algorithm we compare the effect of different values of α. Since an α closer to 1 emphasizes total system bit rate more than fairness amongst users, we find that the distributions get more spread out as we increase α. We also observe that the optimal algorithm schedules 3 or 4 users whereas the other algorithms only schedule 3 users. From Table 1, we see that the optimal algorithm does a better job of filling the power budget and that all algorithms used up all the codes.

VII. Conclusions

In this paper we studied optimally allocating codes and power for the downlink of a CDMA system, taking into account both system-wide and individual user constraints. The objective was to maximize the

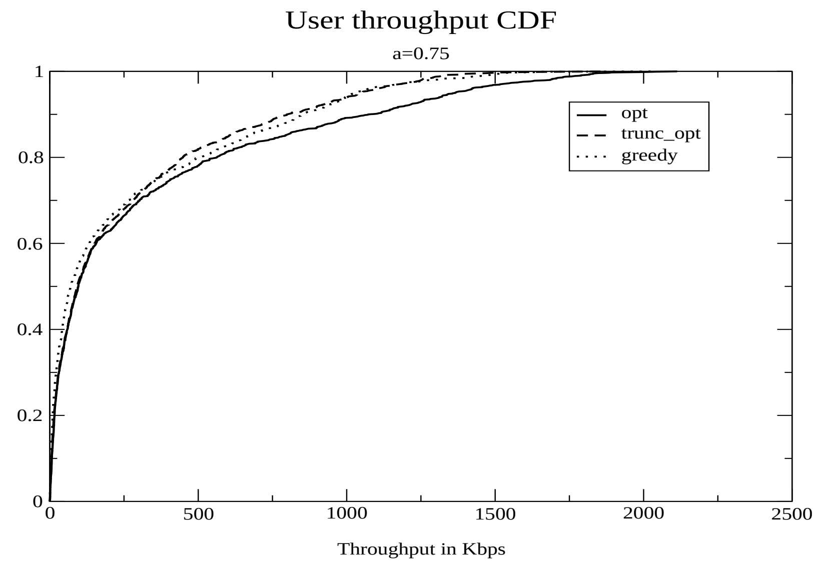

Fig. 11. Empirical CDF of users throughputs for $\alpha = 0.75$.

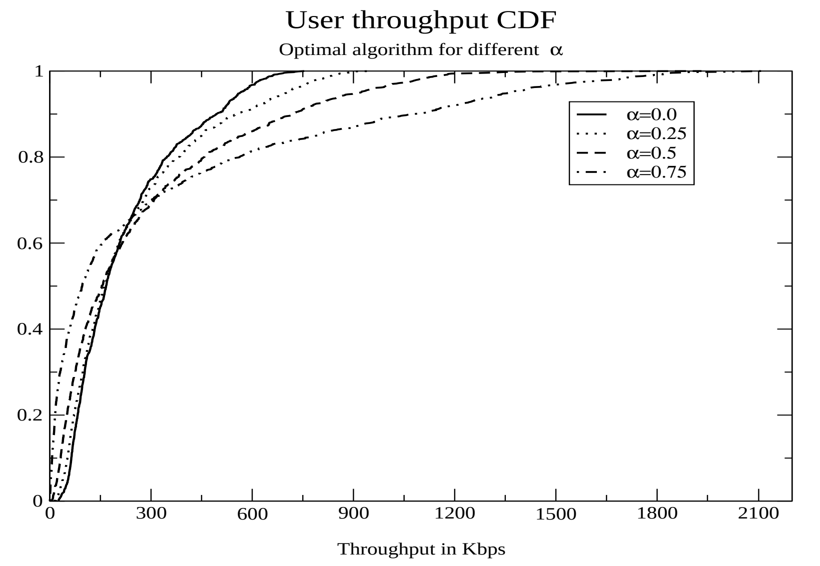

Fig. 12. Empirical CDF of users throughputs for the optimal algorithm with different α's.

weighted sum throughput, where the weights were determined by a gradient-based scheduling algorithm. By formulating this as a convex optimization problem, we were able use a dual approach to characterize the optimal solution. This provides a tight upper-bound on system performance that can be used as a benchmark for designing other low-complexity sub-optimal algorithms. We were also able to characterize several key structural properties of the optimal solution. In particular, a greedy code assignment was shown to be optimal based on a simple ordering of the users; the optimal power assignment was shown to be a modified water-filling allocation. Additionally, we showed that at most $\lceil N/N_{min} \rceil + 1$ users need to be scheduled in any time-slot and all but two will have their full code allocation. Furthermore, for a fixed code assignment, we gave a finite-time algorithm to determine the optimal power allocation and we characterized several properties of the dual functions arising in our analysis. Based on the results, we presented several variations of an optimal algorithm with geometric convergence. We also proposed severallower complexity heuristics. In numerical results, we observed that these algorithms yield better performance than a greedy baseline approach which splits the scheduling and resource allocation into two steps.

Here, we focused on the downlink in a CDMA-based systems. Related problems also arise for the uplink and for other multiplexing techniques such as OFDM [25], [36]. Also, we assumed perfect channel quality feedback and did not address retransmissions. In particular, approaches based on hybrid ARQ are part of most high-speed wireless data standards. One heuristic approach for dealing with this is to "bump up" e_i for packets that that are retransmitted, since they should require a lower SINR to be decoded successfully.

Acknowledgments

We would like to thank Anand Bedekar, Gokul Casheekar, Aniruddha Diwan and Phil Fleming for their help with this paper. We would also like to acknowledge the extremely helpful comments from the anonymous reviewers that helped improve the presentation of this document.

References

[1] R. Agrawal, A. Bedekar, R. La, V. Subramanian, "A Class and Channel-Condition based Weighted Proportionally Fair Scheduler," *Proc. of ITC 2001*, Salvador, Brazil, Sept. 2001.

[2] R. Agrawal and V. Subramanian, "Optimality of Certain Channel Aware Scheduling Policies," *Proc. of 2002 Allerton Conference on Communication, Control and Computing*, Oct. 2002.

[3] L. Tassiulas and A. Ephremides, "Dynamic server allocation to parallel queue with randomly varying connectivity", in *IEEE Transactions on Information Theory*, Vol. 39, pp. 466-478, March 1993.

[4] R. Leelahakriengkrai and R. Agrawal, "Scheduling in Multimedia Wireless Networks," *17th International Teletraffic Congress*, Salvador da Bahia, Brazil, Dec. 2-7, 2001.

[5] R. Leelahakriengkrai and R. Agrawal, "Scheduling in Multimedia CDMA Wireless Networks," *IEEE Trans. on Vehicular Technology*, 2002.

[6] M. Andrews, K. Kumaran, K. Ramanan, A. L. Stolyar, R. Vijayakumar and P. Whiting, "Providing Quality of Service over a Shared Wireless Link", in *IEEE Communications Magazine*, pp.150-154, 2001, Vol.39, No.2.

[7] S. Shakkottai and A. L. Stolyar, "Scheduling algorithms for a mixture of real-time and non-real-time data in HDR," in *Proceedings of the 17th International Teletraffic Congress*, pp. 793-804, Salvador da Bahia, Brazil, 24-28 Sept., 2001.

[8] P. Bhagwat, P. Bhattacharya, A. Krishna and S. K. Tripathi, "Enhancing throughput over wireless LANs using channel state dependent packet scheduling", in *Proceedings of IEEE INFOCOM*, San Francisco, CA, March 1996.

[9] S. Shakkottai and R. Srikant, "Scheduling real-time traffic with deadlines over a wireless channel," in *ACM/Baltzer Wireless Networks Journal*, Vol. 8, No. 1, pp. 13–26, January 2002.

[10] S. Shakkottai and A. L. Stolyar, "Scheduling for multiple flows sharing a time-varying channel: The exponential rule," in *Analytic Methods in Applied Probability*, Amer. Math. Soc. Transl., Series 2, Volume 207, pp. 185–201, Amer. Math. Soc., Providence, RI, 2002.

[11] A. L. Stolyar, "On the Asymptotic Optimality of the Gradient Scheduling Algorithm for Multi-user Throughput Allocation," *Operations Research*, 2005, Vol. 53, No. 1, pp. 12-25.

[12] Y. Liu and E. Knightly, "Opportunistic Fair Scheduling over Multiple Wireless Channels", in *Proc. of IEEE INFOCOM*, San Francisco, CA, March 2003.

[13] X. Liu, E. K. P. Chong, and N. Shroff, "Opportunistic transmission scheduling with resource sharing constraints in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 10, Oct. 2001.

[14] V. Bharghavan, S. Lu and T. Nandagopal, "Fair queuing in wireless networks: Issues and Approaches," *IEEE Personal Communications*, Vol. 6, pp. 44-53, Feb 1999.

[15] P. Liu, R. Berry, and M. Honig, "Delay-Sensitive Packet Scheduling in Wireless Networks," *Proc. of IEEE WCNC 2003*, New Orleans, LA, March 16-20, 2003.

[16] S. Borst, "User level performance of channel aware scheduling algorithms in wireless data networks," *IEEE/ACM Trans. Networking*, vol. 13, no. 3, pp. 636-647, June 2005.

[17] A. Eryilamz, R. Srikant, and J.R. Perkins, "Stable scheduling policies for fading wireless channels," *IEEE/ACM Trans. Networking*, vol. 13, no. 2, pp. 411-424, April 2005.

[18] P. Liu, R. Berry, and M. Honig, “A fluid analysis of a utility-based wireless scheduling policy,” *IEEE Transactions on Information Theory*, vol. 52, no. 7, pp. 2872-2889, July 2006.

[19] L. Ying, R. Srikant, A. Eryilmaz, and G. Dullerud, “A Large Deviations Analysis of Scheduling in Wireless Networks,” *IEEE Transactions on Information Theory*, vol. 52, no. 11, pp. 5088-5098, Nov. 2006.

[20] X. Wang, G.B. Giannakis, and A.G. Marques, “A Unified Approach to QOS-Guaranteed Scheduling for Channel-Adaptive Wireless Networks,” *Proceedings of the IEEE*, vol. 95, no. 12, pp. 2410-2431, December 2007.

[21] P. Bender, P. Black, M. Grob, R. Padovani, N. Sindhushayana and A. Viterbi. “CDMA/HDR: a bandwidth-efficient high-speed wireless data service for nomadic users,” in *IEEE Commun. Mag.*, pp. 70-77, July 2000.

[22] A. Jalali, R. Padovani, R. Pankaj, “Data throughput of CDMA-HDR a high efficiency - high data rate personal communication wireless system.,” in *Proc. VTC ’2000*, Spring, 2000.

[23] H. Holma and A. Toskala, Editors, “W-CDMA for UMTS: Radio Access for Third Generation Mobile Communications,” Wiley & Sons, 2002.

[24] *3GPP2 C.S0002-C: Physical Layer Standard for CDMA2000 Spread Spectrum Systems*, Release C, May 28, 2002, available at *http://www.3gpp2.org/Public_html/specs/C.S0002-C_v1.0.pdf*.

[25] J. Huang, V. Subramanian, R. Agrawal, and R. Berry, “Downlink Scheduling and Resource Allocation for OFDM Systems,” *IEEE Transactions on Wireless Communications*, vol. 8, no. 1, pp. 288-296, Jan. 2009.

[26] A. L. Stolyar, “MaxWeight scheduling in a generalized switch: state space collapse and equivalent workload minimization in Heavy Traffic,” *Ann. Appl. Probab.*, vol. 14, no. 1, pp. 1-53, 2004.

[27] A. L. Stolyar, “Maximizing Queueing Network Utility Subject to Stability: Greedy Primal-Dual Algorithm,” *Queueing Systems*, 2005, Vol. 50, No. 4, pp. 401-457.

[28] D. Tse, “Optimal Power Allocation over Parallel Gaussian Broadcast Channels,” *Proc. of ISIT*, 1997.

[29] L. Li and A. Goldsmith, “Optimal Resource Allocation for Fading Broadcast Channels- Part I: Ergodic Capacity,” *IEEE Trans. on Information Theory*, March 2001.

[30] R. Agrawal, V. Subramanian, and R. Berry, “Joint Scheduling and Resource Allocation in CDMA Systems,” *Proc. of 2nd Workshop on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt ’04)*, Cambridge, UK, March 24-26, 2004.

[31] K. Kumaran and H. Viswanathan, “Joint Power and Bandwidth Allocation in Downlink Transmission,” *IEEE Transactions on Wireless Communications*, Vol. 4, No. 3, pp. 1008-1016, May 2005.

[32] R. Gallager, *Information Theory and Reliable Communication*, John Wiley and Sons, 1968.

[33] J. Mo and J. Walrand, “Fair end-to-end window-based congestion control,” *IEEE/ACM Transactions on Networking*, vol. 8, no. 5, pp. 556 – 567, October 2000.

[34] D. Bertsekas, *Nonlinear Programming*, Athena Scientific, 1995.

[35] R. T. Rockafellar and R. J.-B. Wets, “Variational analysis,” *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*, Vol. 317, Springer-Verlag, Berlin, 1998.

[36] J. Huang, V. Subramanian, R. Berry and R. Agrawal, “Scheduling and Resource Allocation in OFDMA Wireless Communication Systems,” in *Orthogonal Frequency Division Multiple Access*, edited by Tao Jiang, Lingyang Song, Yan Zhang, Auerbach Publications, CRC Press, 2009.

[37] V. Subramanian, R. Agrawal, and R. Berry, “Joint Scheduling and Resource Allocation in CDMA Systems,” *Technical Report*, 2006.