Abstract—In Universal Mobile Telecommunication Systems (UMTS), the Downlink Shared Channel (DSCH) may be used to provide streaming services. The traffic model for streaming services is different from the continuously-backlogged model used in much of the literature. Each connection specifies a required service rate over an interval of time. In this paper, we are interested in determining how $K$ DSCH frames should be allocated among a set of $I$ connections. We need a scheduler that is channel-aware, so that channels presently enjoying low fading losses can be exploited to achieve higher aggregate throughput. On the other hand, the scheduler is also required to be fair, so that each connection obtains a throughput as close as possible to what it requires. We introduce the notion of discrepancy to capture the inherent trade-off between aggregate throughput and fairness. We show that the discrepancy criterion provides a flexible means for balancing efficiency, as measured by aggregate throughput, and fairness. Our problem, then, is to schedule mobiles so as to minimize the discrepancy over the control horizon.

We provide a simple low-complexity heuristic called ITEM that is provably optimal in certain cases. In particular, we show that ITEM is optimal when applied in the UMTS context. Next, we compare the performance of ITEM with those of other algorithms, and show that it performs better in terms of both fairness and aggregate throughput. Thus, ITEM provides benefits in both dimensions — fairness and efficiency — and is therefore a promising algorithm for scheduling streaming connections.

I. INTRODUCTION

Universal Mobile Telecommunications System (UMTS) [1], [2] is the third generation mobile communication system developed by ETSI, the European Telecommunications Standards Institute. UMTS uses the UMTS Terrestrial Radio Access (UTRA), a Wideband Code Division Multiple Access (WCDMA) as the radio access scheme. Uplink and downlink communications are usually differentiated using the Frequency Division Duplex (FDD) mode. In UTRA FDD WCDMA, there are three types of downlink transport channels for transmitting bursty data: common, dedicated and shared channels. Of particular interest in this paper is the Downlink Shared Channel (DSCH) [1] which is a time-shared channel used for high bit-rate transmission. By definition, the DSCH capacity can be shared among several users on a frame-by-frame basis. Each frame has a duration of 10 ms.

A streaming application (for example, video on demand) results in a data flow that is almost entirely one-way. We consider $I$ mobiles in a cell that have subscribed to the streaming service. These mobiles expect data flows at “contracted” rates $r_1, r_2, \ldots, r_I$, respectively. Assuming that the original sources of data are connected to the Radio Access Network (RAN) by high speed links, the bottleneck is the last-hop shared wireless link, i.e., the DSCH. The DSCH must be shared among the $I$ flows such that each receives a long-term bit rate as close as possible to its contracted rate. Because of the contracted rates, our traffic model is different from that considered in much of the literature. The usual continuously-backlogged model assumes that the mobiles are downloading very large files (e.g., using ftp) and there is no desired rate that a mobile has specified. In this scenario, the objective is to ensure fair and efficient sharing of the wireless link. Our model is different because the mobiles specify the contracted rates $r_1, r_2, \ldots, r_I$. The bit error rates that the mobiles can tolerate are also specified.

Our problem is different in another aspect too: We explicitly consider a control horizon, which is the time period over which the scheduling problem is to be solved. In the continuously-backlogged model, scheduling problems are formulated in terms of rates only, without reference to a finite control horizon — which essentially implies that the downloads go on forever. In reality, of course, control problems must always be solved over finite intervals of time, and the interval lengths can possibly influence scheduling decisions. In our formulation, we take the control horizon into account explicitly.

We state our problem in terms of contracted service, which is just the product of a mobile’s contracted rate and the control horizon. The basic problem considered is that of scheduling the mobiles on the downlink such that the aggregate service received by each over the control horizon is as close as possible to its contracted service.

When making scheduling decisions, the scheduler must consider the prevailing channel conditions to each mobile. If mobile $i$ has a poor channel currently, then the maximum rate at which it can receive, given its target bit error rate, is small. This means that over some time, the average bit rate seen by $i$ can drop below $r_i$. Later, when channel conditions improve, $i$ may be able to receive at a higher rate and thereby the lost service may be compensated for. Thus, over the control horizon, the service allotted to each mobile may vary from the ideal amount that it desires over the horizon.

To capture the difference between the ideal and allotted amounts of service for the collection of $I$ mobiles, we introduce a natural criterion called the “discrepancy criterion.” With this, our problem can be stated as: Schedule the mobiles
so as to minimize the discrepancy criterion over the control horizon. This already suggests that we view our problem in the framework of optimization theory; for this reason, we also refer to the discrepancy criterion as the “objective function.”

The discrepancy criterion is a notion that allows us to address the conflicting issues of fairness and aggregate throughput in a unified manner. As we will see later, the discrepancy criterion is precisely defined by choosing a function from a set of functions, and the chosen function can be used to trade off fairness for aggregate throughput. Thus, the discrepancy criterion is a flexible means for balancing fairness and efficiency. By choosing an appropriate discrepancy criterion, the operating point of the system can be adjusted as desired.

In [3], a scheduler with provable statistical temporal fairness properties was proposed and analytical bounds for fairness were given. Our definition of fairness is different from the temporal fairness defined there. Moreover, in [3], an algorithm was proposed and its properties were studied, whereas we start with the definition of the discrepancy criterion and design an algorithm to minimize the discrepancy.

In this paper, we design and analyze ITERative Minimizer (ITEM), a provably optimal scheduling algorithm for minimizing the discrepancy criterion. We show that it results in the optimal value of the discrepancy when applied to the UMTS Rate Sets in a computationally efficient manner. We evaluate the performance of ITEM in terms of the discrepancy measure, execution time and the aggregate throughput achieved. We compare its performance with other scheduling algorithms and show that it performs better both in terms of fairness and aggregate throughput.

The remainder of the paper is organized as follows. In Section II, we review the related work. Section III describes the system model and the assumptions made. Section IV proposes ITEM and compares it with the Exhaustive Search method. Section V studies the performance of ITEM in comparison to the other scheduling algorithms and Section VI concludes the work.

II. RELATED WORK

The problem of scheduling on the downlink of wireless networks has received considerable attention in the literature. One group of papers considers the scheduling of data traffic, utilizing the knowledge of the file sizes to be transferred [4]. A second group of papers is concerned with adaptations of packet-fair queuing algorithms—that have been suggested for wireline networks—to the wireless context [5], [6]. A third group considers plausible heuristics for scheduling connections, and evaluates performance by simulations; see, for example, [7], [8], [9] and references therein. Our approach is different in that we do not start with algorithms and then obtain their performance properties. Rather, we start with a criterion to optimize and this drives the development of our algorithm.

In [10], the problem of intra-cell scheduling was considered. The objective was to minimize the total energy consumed over the scheduling interval, \( T \), subject to a limit on the total base station transmission power. The scheduling interval \( T \) plays the role of control horizon in our context and average rate of user \( i \) plays the role of contracted rate \( r_i \). However, there is no consideration of fairness in [10]. Since fairness is not considered, certain users may be denied channel access. In our work, fairness is inherent because it is built into the discrepancy criterion.

We compare ITEM with the Proportional Fairness scheduler (PFS) proposed in [11]. In addition, the objective function value and the aggregate throughput achieved using ITEM are compared with those for the Adaptive Proportional Fairness Scheduler (APFS) proposed in [12], the Credit Based Algorithm (CREDIT) proposed in [13] and Wireless Credit Based Fair Queuing (WCFQ) proposed in [3]. We show that ITEM outperforms the others on both counts. A short description of each of these scheduling algorithms and the numerical comparison results are given in Section V.

III. SYSTEM MODEL

UMTS is one of the major new third generation mobile communications systems being developed within the framework which has been defined by the International Telecommunications Union (ITU). UMTS provides increased capacity and data capability, which enables the delivery of not only voice but also multimedia content such as pictures, graphics and video. This is made possible through the use of various transport channels—common, dedicated and shared.

DSCH [1], [2] provides a fast time-multiplexed resource shared by a large number of users, where different users can access the system on a frame-by-frame basis. The DSCH is always associated with a Downlink Dedicated Channel (DCH) [1]. The DCH provides information for power control and other transport format parameters, as well as an indication to the terminal about which code from the DSCH it has to despread and when to do so. The transmission power of the DSCH is computed on the basis of the power of the DCH associated to the user actually transmitting on the DSCH [14]. The main advantage of DSCH is that many packet data users with bursty downlink traffic can share a single downlink channelization code. The shared channels allow users to receive data bursts at high rates by using short leases on the radio resource in a time-division-multiplexed manner.

The selection of a user in any frame is based on striking a balance between aggregate throughput and fairness. In each frame, one cannot always choose the user with the highest rate even though it would lead to high aggregate throughput, because users with lower SNRs will be starved. Fair sharing will lower the total throughput over the maximum possible, but it will provide more acceptable levels to users with poorer SNRs. Thus, there is a trade-off between fairness and efficiency and scheduling algorithms have to be developed which balance both.

We consider \( I \) mobiles that have subscribed to the streaming service. These mobiles expect data flows at contracted rates \( r_1, r_2, \ldots, r_I \) respectively. The desired service is obtained from the contracted bit rate of the mobile, the number of frames \( k \) and the time duration of each frame \( \tau \). For example, \( r_i k \tau \) is the overall requirement of user \( i \) (in bits) and \( k \tau \) is
the control horizon. Occasionally, we refer to \( k \) itself as the control horizon; it is then understood that the unit of time is \( \tau \). The set of feasible transmission rates available for user \( i \) is denoted by \( S_i \).

The objective is to determine the scheduling rule which ensures that each mobile receives a service that is as close as possible to its desired service. We use a distance metric \( f(\cdot) \) in the \( I \)-dimensional space to measure the discrepancy between the received and required service. Let \( x, y \) and \( z \) be \( I \)-dimensional points. The distance metric \( f(\cdot) \) should satisfy:

\[
\begin{align*}
    f(x,y) &\geq 0 \\
    f(x,y) &= f(y,x) \\
    f(x,z) &\leq f(x,y) + f(y,z)
\end{align*}
\]

\( f(x,y) = 0 \) if and only if \( x = y \). For example, the Euclidean distance metric can be used as the discrepancy measure: \( f(x,y) = \sqrt{(x_1-y_1)^2 + \ldots + (x_I-y_I)^2} \), where \( x = (x_1, x_2, \ldots, x_I)^T \) and \( y = (y_1, y_2, \ldots, y_I)^T \). However, we exclude the discrete distance metric [15] from the list of choices for \( f(\cdot) \). In the next section, we will give an example showing how the choice of \( f(\cdot) \) influences scheduling decisions, and thereby allows us to trade off fairness with efficiency.

Variations in channel conditions can be attributed to three basic phenomena: fast fading on the order of milliseconds, shadow fading on the order of tens of hundreds of milliseconds and finally, long timescale variations due to user mobility. We assume that: (i) Rayleigh or fast fading is perfectly compensated by power control and (ii) variations in the channel due to shadow fading and user mobility are absent over the control horizon. Similar assumptions have been made by several authors; see, for example, [4]. However, each user \( i \) sees a different channel (therefore, a different \( S_i \)) depending on the loss factor between the base station (BS) and user \( i \). The loss factor for a user remains the same over the entire control horizon. Therefore, \( S_i \) for user \( i \) remains invariant over the control horizon.

### IV. Optimal Scheduling

#### A. Problem Definition

We assume that a DSCH has been set up between the Radio Network Controller (RNC) and the mobiles. The Packet Scheduler (PS) at the RNC has the task of determining which of the connections must be scheduled in each frame. Various transmission rates are achieved by modifying the spreading factor associated with mobile \( i \), keeping the chip rate fixed at the value specified by UMTS technology.

There are \( I \) mobiles. The sets of feasible rates for the \( I \) mobiles are \( S_1, S_2, \ldots, S_I \) respectively. We have a horizon of \( k \) time steps called frames and need to obtain scheduling decisions for each time step in this horizon. The epochs demarcate the frames. The time duration between the epochs \( l \) and \( (l+1) \) defines the frame \( (l+1) \), \( 0 \leq l \leq (k-1) \). The scheduling decision at epoch \( l \) determines the mobile and the rate at which it should be scheduled in frame \( (l+1) \). Let \( j_i(l) \), \( 0 \leq l \leq (k-1), 1 \leq i \leq I \), be binary decision variables that indicate whether mobile \( i \) is scheduled in frame \( (l+1) \). Further, let \( x_i(l) \), \( 0 \leq l \leq (k-1), 1 \leq i \leq I \) denote the rate assigned to the mobile \( i \) if it is scheduled in frame \( (l+1) \). The vector of required services \( (r_1k\tau, r_2k\tau, \ldots, r_kk\tau)^T := r(k) \) can be thought of as a point in \( I \)-dimensional space, where \((\cdot)^T \) indicates the transpose of a row vector. Let \( S_i(k) = \tau \sum_{l=0}^{k_i(l)-1} j_i(l)x_i(l), 1 \leq i \leq I \), indicate the total service allotted to mobile \( i \) till epoch \( k \). Then, \( (S_1(k), S_2(k), \ldots, S_I(k))^T := S(k) \) gives the vector of allotted services up to frame \( k \). \( f(r(k), S(k)) \) gives the measure of the discrepancy between the required and the allotted service.

For example, we consider the case of two mobiles and let \( f(\cdot) \) be the Euclidean norm. Here, the objective function is

\[
f(r(k), S(k)) = \sqrt{(r_1k\tau - S_1(k))^2 + (r_2k\tau - S_2(k))^2}
\]

Introducing the term \( T_i(k) = \frac{S_i(k)}{k\tau}, i = 1, 2 \), the above equation can be written as

\[
f(\cdot) = k\tau \sqrt{\left(1 - T_1(k) \right)^2 + \left(1 - T_2(k) \right)^2}
\]

\( T_i(k) \) is the throughput actually achieved at the end of the control horizon and \( r_i \) is the target. \( 1 - T_i(k)^2 \) gives the square of the absolute deviation from the ideal value of 1. Thus, \( f(r(k), S(k)) \) in Equation 1 can be interpreted as the square root of the aggregate squared deviation from the ideal vector \( (1, 1)^T \), with \( r_1^2 \) and \( r_2^2 \) being the relative weights of the two terms.

If, for each mobile, the allotted service is equal to its requirement, then \( f(r(k), S(k)) = 0 \) because the two points \( r(k) \) and \( S(k) \) coincide. It may not be possible to allot service in this perfect manner. A schedule which leads to a small value of overall discrepancy (small \( f(r(k), S(k)) \)) is preferable because it balances the required and allotted service to the extent possible. With the above objective, we pose the following problem:

\[
\text{minimize } f(r(k), S(k))
\]

subject to

\[
\begin{align*}
    x_i(l) &\in S_i, l = 0, 1, \ldots, (k-1), i = 1, 2, \ldots, I \\
    \sum_{i=1}^{I} j_i(l) &\leq 1, l = 0, 1, \ldots, (k-1), i = 1, 2, \ldots, I \\
    j_i(l) &\in \{0, 1\}
\end{align*}
\]

The unknowns to be solved for are \( j_i(l) \), and \( x_i(l), 0 \leq l \leq (k-1), 1 \leq i \leq I \). The right hand sides of Equation (2) are fixed and given. Further, \( x_i(l) \) must belong to the corresponding discrete set \( S_i \). The constraint in Equation (3) ensures that at most one mobile can be scheduled in each frame.

Now we give a simple example to show how different functions \( f(\cdot) \) can lead to different actions being chosen even though the requirements and allotted services remain unchanged. Suppose we have only 2 mobiles \( A \) and \( B \), and a control horizon \( k = 1 \). The requirements of the mobiles are represented by the point \((4, 3)\) in some units. Action 1 corresponds to scheduling mobile \( A \) and action 2 corresponds
to scheduling mobile $B$. If $A$ is scheduled, it receives 1 unit of service, while if $B$ is scheduled, it receives 2 units of service. Then we have $S(1) = (1, 0)$ for action 1, and $S(1) = (0, 2)$ for action 2.

If $f(.)$ is the Euclidean metric, then $f(r(1), S(1)) = \sqrt{18}$ for action 1, and $f(r(1), S(2)) = \sqrt{17}$ for action 2, showing that action 2 will be chosen. However, when $f(.)$ is the $3^{rd}$-norm, we find that $f(r(1), S(1)) = (54)^{\frac{1}{3}}$ for action 1, while $f(r(1), S(2)) = (65)^{\frac{1}{3}}$ for action 2; hence action 1 will be chosen. Moreover, it can be seen that choosing the $3^{rd}$-norm leads to fairer operation because the minimum discrepancy is closer to 0 for this choice; but the price paid is lower aggregate throughput.

B. Exhaustive Search

A straightforward approach for solving the above problem is the exhaustive search procedure. If $I$, the number of mobiles, $(m + 1)$, the number of rates feasible for each user (including the rate 0) and $k$, the time horizon are given, then the exhaustive search procedure searches through a total of $(I(m + 1) + k - 1) C_k$ points to find the optimal solution where $JC_r$ is the number of ways of choosing $r$ objects from $n$ objects. If $m_r$ is the number of rates feasible for user $i$, then the number of points to be searched to find the optimal solution is equal to $(\sum_{i=1}^{I} m_i + k - 1) C_k$. Naturally, the computational effort increases drastically with increasing $I$, $m$ or $k$. In practice, heuristics are required.

C. The ITErative Minimizer (ITEM) Heuristic

The ITEM heuristic solves the problem iteratively at each epoch, taking into account the service received by each mobile till the present epoch. The updated requirements of a mobile $i$ at epoch $l$, $u_i(l)$ is equal to its required service minus the service allotted to it till epoch $l$, i.e., $u_i(l) = r_i(l) - S_i(l)$. A negative $u_i(l)$ means that the service allotted to mobile $i$ till epoch $l$ exceeds its required service. The updated requirements of mobiles at epoch $l$ is denoted by $u(l) = (u_1(l), u_2(l), \ldots, u_I(l))^t$. Of course, $u(0) = r(0)$ and $S(0) = 0^t$, where 0 is the zero row vector. The vectors $u(.)$ and $S(.)$ are updated after the decision at each epoch. The binary variables $pos_i$, $1 \leq i \leq I$ indicate whether $u_i(k - 1)$ is positive (value 1) or negative (value 0).

Let $l \in \{0, 1, \ldots, (k - 1)\}$ and $u(l)^t > 0$. Suppose that $S_i(l)$, $1 \leq i \leq I$, are given. Let $x^*$ indicate the mobile chosen at epoch $l$ by ITEM and $x_i(l)$ be the rate assigned to it. Then, ITEM obtains $x^*$ and $x_i(l)$ as a solution to the following minimization problem:

$$f\left(u(l), (\tau j_1(l)x_1(l), \ldots, \tau j_I(l)x_I(l))^t\right)$$

subject to

$$x_i(l) \in \mathbb{S}, \quad i = 1, 2, \ldots, I$$

$$j_1(l) + j_2(l) + \ldots + j_I(l) \leq 1$$

$$j_i(l) \in \{0, 1\}, \quad i = 1, 2, \ldots, I$$

1For brevity, we omit the proof; we note in passing that $(I(m + 1))^k$ overestimates the number of points to be searched.

If $u(l)^t > 0$, $l = 0, 1, \ldots, (k - 1)$ and $(m + 1)$, the number of rates feasible for each user, then ITEM finds the solution after a search among at most $I(m + 1)k$ possibilities. We note that ITEM is a greedy algorithm. If some of the terms in $u(l)$ are negative, $x^*$ and $x_i(l)$ are determined differently.

1) General properties of ITEM: Consider the case where $r_i(l)$ is large for $1 \leq i \leq I$. By the term “large”, we mean that even if user $i$ is scheduled in all the frames with the maximum possible feasible rate, $(r_i(l) - S_i(l)) > 0$, for $1 \leq i \leq I$. Suppose that we have found, by Exhaustive Search, a sequence of actions $a^*(l)$, $0 \leq l \leq (k - 1)$, that optimizes the objective function at the end of the horizon of length $k$. By $a^*(l)$ we denote the mobile chosen at the $l^{th}$ epoch and its rate, i.e., the non-zero $j_i(l)$ and the corresponding $x_i(l)$. Now under the assumption that $r_i(l)$ is large for $1 \leq i \leq I$, the order of the actions $a^*(0), a^*(1), \ldots, a^*(k - 1)$ can be changed without changing the objective function value. This is because the objective function depends on the total amount of service received by a mobile in the $k$-step horizon, and not on how the total amount is broken up among the steps.

At epoch 0, let us choose that action from $\{a^*(0), a^*(1), \ldots, a^*(k - 1)\}$ which minimizes the distance between required and allotted service, as measured by the function $f(.)$. Let us denote this by $a^*(p^*(0))$. Having chosen this action, let us choose at epoch 1, that action from $\{a^*(0), a^*(1), \ldots, a^*(k - 1)\} \setminus \{a^*(p^*(0))\}$ which minimizes the distance at epoch 1. This process is continued for the $(k - 1)$ epochs. Clearly, $\{a^*(p^*(0)), a^*(p^*(1)), \ldots, a^*(p^*(k - 1))\}$ is a permutation of $\{a^*(0), a^*(1), \ldots, a^*(k - 1)\}$, and it minimizes, among all permutations, the distance at epochs 0, 1, $\ldots, (k - 1)$.

We claim that the actions $a^*(p^*(0)), a^*(p^*(1)), \ldots, a^*(p^*(k - 1))$ are exactly the actions that ITEM chooses. Let us recall that at each $l$, ITEM chooses, from all $I(m + 1)$ possible actions, the one that minimizes the distance at each epoch. Suppose, if possible, ITEM finds an action $\hat{a}(l)$ at epoch $l$ that strictly improves the distance at epoch $l$. Then, the sequence $a^*(p^*(0)), a^*(p^*(1)), \ldots, a^*(p^*(l - 1)), \hat{a}(l), a^*(p^*(l + 1)), \ldots, a^*(p^*(k - 1))$ would strictly improve the objective function at epoch $(k - 1)$ (again, we use the assumption that $r_i(l)$ is large for all $i$). This is a contradiction because we know already that $a^*(0), a^*(1), \ldots, a^*(k - 1)$ is the optimal sequence of actions found by Exhaustive Search.

The arguments above establish the following:

**Proposition 4.1.** When $r_i(l)$ is large for $1 \leq i \leq I$, the ITEM policy is a computationally efficient way of finding the optimal sequence of actions for the scheduling problem.

2) ITEM applied to DSCH in the UMTS Network: Here, we consider a special case where ITEM is applied to the DSCH rate set in the UMTS network. We show that in this case, ITEM always results in the optimal value of the discrepancy. We present a high level description of ITEM, provide the pseudo-code of ITEM and study its properties.

Consider the DSCH where the feasible rate set for each user has the following structure:

$S_i = \{0, x_1, 2x_1, x_2x_1, \ldots, 2^{(m_i - 1)}x_1\} \quad i = 1, 2, \ldots, I.$
The above procedure is repeated till the last epoch is to convert the negative updated requirements into updated requirements, the one with maximum magnitude is selected (say \( i^* \)). The epoch \( e \) is identified such that \( u_i(e)u_i(e + 1) < 0 \). \( x_i(e) \) is used to make \( u_i(l) \) positive. The basic idea is to convert the negative updated requirements into positive values so that solution of equation 4 gives \( i^* \) and \( x_i(l) \). If \( i^* \neq i \), then \( u_i(l) \) and \( S_i(l) \) are restored to their original values. The updated requirements and the service received of the mobiles are updated. The above procedure is repeated till the last epoch and the optimum value of the discrepancy is calculated.

The steps can be enumerated as follows:

1. At each epoch \( l, l \neq (k - 1) \), if \( u(l) \) is positive, then
   - Determine \( i^* \) and \( x_i(l) \), as in Equation 4.
   - Otherwise,
     - Determine \( i = \arg \max_x \{ u_i(l) : u_i(l) < 0 \} \).
     - Find the epoch \( e \), such that \( u_i(e + 1) < 0 \) and \( u_i(e) > 0 \) and assign \( x_i(e) \) to \( f_i(e) \).
     - If \( |u_i(l)| \) is not the maximum among all mobiles, then
       a) Modify \( x_i(e), u_i(l) \) and \( S_i(l) \) as
          \[
          x_i(e) = f_i(e)/2 \quad u_i(l) = u_i(l) + \tau x_i(e) \quad S_i(l) = S_i(l) - \tau x_i(e)
          \]
       b) Determine \( i^* \) and \( x_i(l) \) among the mobiles with positive updated requirements as in Equation 4.
      c) If \( i^* \neq i \), restore \( x_i(e), u_i(l) \) and \( S_i(l) \).
   - Otherwise
     a) Determine \( i^* \) at epoch \( l \) as \( i^* = i \).
     b) Modify \( x_i(e), u_i(l) \) and \( S_i(l) \) as before and determine \( x_i(l) \).

2. Update \( S_i(l + 1) \) and \( u_i(l + 1) \) as:
   \[
   S_i(l + 1) = S_i(l) + \tau x_i(l) \\
   S_i(l + 1) = S_i(l), i \neq i^* \\
   u_i(l + 1) = u_i(l) - \tau x_i(l) \\
   u_i(l + 1) = u_i(l), i \neq i^*
   \]

3. At epoch \((k - 1)\), determine \( i^* \) and \( x_i(k - 1) \) after converting all negative \( u_i(k - 1) \) into positive as mentioned in Step 1.

4. Update \( u(k) \) and \( S(k) \) and determine \( f(r(k), S(k)) \).

The pseudo-code is given in Algorithm 1.

We now proceed to state the observations from ITEM applied to the DSC1 in the UMTS network in the form of Lemmas 4.1-4.3.

**Lemma 4.1:** \( u_i(l) \), after step 10 in Algorithm 1 is guaranteed to be positive.
Lemma 4.2: The rate to be selected at epoch $c$ to convert a negative $u_i(l)$ into positive is half of the rate which was originally selected at epoch $c$.

Lemma 4.3: If at epoch $l$, $l \leq (k - 1)$, user $i^*$ is selected, such that $u_i(l) < 0$, then $x_i(e) = f_i(e)/2$ and $x_i(l) = f_i(e)/2$ or $f_i(e)/4$ and $S_i = [x_i^*, 2x_i^*, \ldots, x_i(l)/2]$. The proofs of Lemmas 4.1-4.3 are given in the Appendix.

Given the special structure of $S_i$, where the elements are in geometric progression with an integral common ratio, or in arithmetic progression with common difference greater than the first term, we can state the following proposition.

Proposition 4.2: For arbitrary requirements $r_1, r_2, \ldots, r_I$ and a subset of UMTS rate sets $S_i = [30, 60, 120, 240]$ (kbps) $i = 1, 2, \ldots, I$, ITEM results in the optimal value of the objective function $f(.)$. The proof is given in the Appendix.

3) ITEM execution time and its comparison with Exhaustive Search: We now present a numerical comparison of the performance of ITEM with that of Exhaustive Search in terms of the objective function value and the execution time. The DSCH rate sets are used for the comparison. Rayleigh fading is not considered linked with the assumption that Rayleigh fading is perfectly compensated by fast power control. However, the channel seen by each user is different (because of different loss factors). Therefore, the set of feasible rates for each user is different. We show that ITEM is optimal (in the mathematical sense, meaning that it gives the minimum possible value of the discrepancy) when applied to DSCH rate sets, while remaining computationally efficient.

Table I shows the performance comparison between ITEM and the Exhaustive Search method. The distance function chosen is the Euclidean norm, $f(x, y) = \sqrt{(x_1 - y_1)^2 + \ldots + (x_I - y_I)^2}$. The 1st column shows the control horizon $k$. The set of rates $S_i$ available to mobile $i$ (excluding rate 0) is shown next. Different $S_i$ reflect the different channel conditions seen by each user. The total number of rate sets is equal to the number of users $I$. This is followed by mobile $i$'s contracted rate $r_i$. The entry “Exhaust” refers to the Exhaustive Search method. In the next two columns, “Value” refers to the objective function value evaluated using Exhaustive Search and ITEM algorithms, respectively, and “Time” indicates the average execution time taken for computation. The remaining columns show the performance comparison for a different set of requirements and rate sets for the same value of $k$.

Both ITEM and Exhaustive Search method have been implemented in C++. Execution times are determined using the clock() function. The difference between the clock() function values at the end and the beginning of the program divided by $CLOCKS_PER_SEC$ gives the execution time of the program. The program is run repeatedly many times (10000 times) and the average execution time is calculated.

It is observed from the table that as $k$, $m_i$, and $I$ increase, the execution time of Exhaustive Search method increases enormously while that of ITEM remains nearly constant. For example, the comparisons for $k = 11, 17$ show very high execution times for Exhaustive Search method and negligible execution times for ITEM. This clearly shows the computational efficiency of ITEM. As $I$ increases, the execution time of Exhaustive Search method increases enormously for the same $k$, while the execution time for ITEM is hardly affected. This is observed from Table I for $I = 6$. Thus, ITEM results in the optimal value of the objective function in all the cases while remaining computationally efficient.

The second comparison with $k = 5$ is one of the cases where the allocated services for certain users (2 and 3 in this case) are more than their required services. The first comparison with $k = 6$ shows the perfect match where the value of the objective

<table>
<thead>
<tr>
<th>$k$</th>
<th>$S_i$ (kbps)</th>
<th>$r_i$ (kbps)</th>
<th>Value</th>
<th>Time (ms)</th>
<th>$S_i$ (kbps)</th>
<th>$r_i$ (kbps)</th>
<th>Value</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>[30, ..., 1920]</td>
<td>20.000</td>
<td>141.42</td>
<td>40</td>
<td>0.4</td>
<td>[30, ..., 480]</td>
<td>55.635</td>
<td>40.000</td>
</tr>
<tr>
<td></td>
<td>[30, ..., 240]</td>
<td>15.000</td>
<td>141.42</td>
<td>40</td>
<td>0.4</td>
<td>[30, ..., 240]</td>
<td>20.925</td>
<td>47.371</td>
</tr>
<tr>
<td></td>
<td>[30, ..., 240]</td>
<td>10.000</td>
<td>141.42</td>
<td>40</td>
<td>0.4</td>
<td>[30, ..., 240]</td>
<td>146.536</td>
<td>48.576</td>
</tr>
<tr>
<td></td>
<td>[30, ..., 240]</td>
<td>5.380</td>
<td>239.51</td>
<td>20</td>
<td>0.4</td>
<td>[30, ..., 480]</td>
<td>32.530</td>
<td>70.433</td>
</tr>
<tr>
<td></td>
<td>[30, ..., 480]</td>
<td>5.380</td>
<td>239.51</td>
<td>20</td>
<td>0.4</td>
<td>[30, ..., 480]</td>
<td>70.433</td>
<td>40.765</td>
</tr>
<tr>
<td>5</td>
<td>[30, 60]</td>
<td>20.000</td>
<td>206.16</td>
<td>20</td>
<td>0.4</td>
<td>[30, ..., 480]</td>
<td>52.028</td>
<td>42.140</td>
</tr>
<tr>
<td></td>
<td>[30, ..., 240]</td>
<td>15.000</td>
<td>206.16</td>
<td>20</td>
<td>0.4</td>
<td>[30, ..., 480]</td>
<td>42.140</td>
<td>41.239</td>
</tr>
<tr>
<td></td>
<td>[30, ..., 240]</td>
<td>10.000</td>
<td>206.16</td>
<td>20</td>
<td>0.4</td>
<td>[30, ..., 480]</td>
<td>357.57</td>
<td>118.69</td>
</tr>
<tr>
<td></td>
<td>[30, ..., 480]</td>
<td>0.0</td>
<td>0</td>
<td>50</td>
<td>0.4</td>
<td>[30, ..., 480]</td>
<td>81.814</td>
<td>24.231</td>
</tr>
<tr>
<td></td>
<td>[30, ..., 240]</td>
<td>0.0</td>
<td>0</td>
<td>50</td>
<td>0.4</td>
<td>[30, ..., 240]</td>
<td>24.231</td>
<td>24.823</td>
</tr>
<tr>
<td>6</td>
<td>[30, ..., 480]</td>
<td>460.150</td>
<td>29893</td>
<td>1300</td>
<td>0.7</td>
<td>[30, ..., 480]</td>
<td>388.154</td>
<td>80803</td>
</tr>
<tr>
<td></td>
<td>[30, 60]</td>
<td>675.364</td>
<td>29893</td>
<td>1300</td>
<td>0.7</td>
<td>[30, ..., 480]</td>
<td>914.050</td>
<td>80803</td>
</tr>
<tr>
<td></td>
<td>[30, ..., 240]</td>
<td>174.227</td>
<td>29893</td>
<td>1300</td>
<td>0.7</td>
<td>[30, ..., 480]</td>
<td>485.368</td>
<td>80803</td>
</tr>
<tr>
<td></td>
<td>[30, 60]</td>
<td>268.209</td>
<td>29893</td>
<td>1300</td>
<td>0.7</td>
<td>[30, ..., 240]</td>
<td>912.565</td>
<td>80803</td>
</tr>
<tr>
<td></td>
<td>[30, ..., 240]</td>
<td>29893</td>
<td>1300</td>
<td>0.7</td>
<td>[30, ..., 240]</td>
<td>508.080</td>
<td>80803</td>
<td>6290</td>
</tr>
<tr>
<td>11</td>
<td>[30, ..., 480]</td>
<td>212.328</td>
<td>114.56</td>
<td>166980</td>
<td>0.7</td>
<td>[30, ..., 480]</td>
<td>34.225</td>
<td>340.26</td>
</tr>
<tr>
<td></td>
<td>[30, 60]</td>
<td>89.668</td>
<td>114.56</td>
<td>166980</td>
<td>0.7</td>
<td>[30, ..., 480]</td>
<td>68.044</td>
<td>340.26</td>
</tr>
<tr>
<td></td>
<td>[30, ..., 240]</td>
<td>86.370</td>
<td>114.56</td>
<td>166980</td>
<td>0.7</td>
<td>[30, ..., 480]</td>
<td>58.060</td>
<td>340.26</td>
</tr>
<tr>
<td></td>
<td>[30, ..., 240]</td>
<td>210.116</td>
<td>38040</td>
<td>11960</td>
<td>0.7</td>
<td>[30, ..., 480]</td>
<td>146.536</td>
<td>2104.05</td>
</tr>
<tr>
<td></td>
<td>[30, ..., 480]</td>
<td>210.116</td>
<td>38040</td>
<td>11960</td>
<td>0.7</td>
<td>[30, ..., 240]</td>
<td>159.247</td>
<td>2104.05</td>
</tr>
</tbody>
</table>

TABLE I

Objective function values & execution times of Exhaustive Search & ITEM run on a P-IV machine at 1.51 GHz with 1 GB RAM.
function is zero. Here, the allotted service of each mobile is equal to its requirement. Therefore \( f(r(k), S(k)) = 0 \) because the two points \( r(k) \) and \( S(k) \) coincide.

If the objective function is too large, it implies that the Connection Admission Control (CAC) module is not doing a good job. However, this also suggests that the CAC module can benefit from using ITEM. If the computed objective is too large, then there is excessive demand and the set of I connections should not be accepted at the present \( r_i \); renegotiation of parameters may be required.

The computational complexity of the algorithms is measured by the number of points to be searched to find the optimal point. Fig. 1 shows the natural logarithm of the computational complexities of Exhaustive Search method and ITEM versus the number of users \( I \) for a fixed control horizon \( k \). We can clearly see that the computational complexity of Exhaustive Search method increases enormously as \( I \) increases, while that of ITEM is hardly affected.

Fig. 2 depicts the natural logarithms of the computational complexities of Exhaustive Search method and ITEM versus \( k \) for a fixed \( I \). The computational complexity of Exhaustive Search method increases with increasing \( k \), while there is a negligible increase in the complexity of ITEM.

Thus, we see that ITEM is an optimal computationally efficient method of determining the scheduling rule over the control horizon. It can also aid in Connection Admission Control so that the requirements of the admitted connections are met as far as possible.

V. PERFORMANCE COMPARISON

In this section, we compare ITEM with other scheduling algorithms. We present numerical evaluations which show how ITEM performs comparatively better both in terms of fairness and the aggregate throughput. The algorithms taken for comparison are PFS, APFS, CREDIT and WCFQ.

A. Proportional Fair Scheduler (PFS)

In [11], a PFS for the High Data Rate (HDR) system was proposed and it was shown in [16] to result in throughput optimization. At epoch \( l \), each mobile user \( i \) sends a data rate control (DRC) message to the base station indicating the maximum rate \( DRC_i(l) \) at which the base station can transmit to that user. It is assumed that \( DRC_i(l) \) is constant over the control horizon, independent of \( l \). At epoch \( l \), the scheduler picks the user that maximizes

\[
DRC_i(l)/R_i(l)
\]

where \( R_i(l) \) is the average achieved rate, and is equal to \( S_i(l)/l \).

When all users experience the same channel conditions and have identical “large” requirements, ITEM and PFS result in the same schedule. At epoch \( l \), PFS selects the user for which \( R_i(l) \) is the smallest. ITEM does the same since the user with smallest \( R_i(l) \) is the one for which \( u_i(l) \) is the largest.

PFS does not take the requirements of the users into account while making scheduling decisions and therefore, the performance suffers when the users have target data rates. For this reason, PFS is not considered for performance comparison in Table II. ITEM takes into account the target data rates and the control horizon while determining the scheduling rule and therefore has a definite edge over PFS.
B. Adaptive Proportional Fair Scheduler (APFS)

The Adaptive Proportional Fair Scheduler (APFS) proposed in [12] is an adaptation of PFS. \(DRC_i\) and \(R_i(l)\) have the same meaning as in PFS. At epoch \(l\), APFS selects user \(i^*\) as:

\[
i^* = \arg\max_{1 \leq i \leq I} \frac{DRC_i}{R_i(l)} \times r_i
\]

We see that ITEM and APFS select the same user at each epoch when the channel conditions are the same, the requirements of a certain user compared to other users is very high and also in certain other cases. APFS starts with an algorithm and then derives the performance measures while we consider a discrepancy criterion for the users and then obtain the algorithm which minimizes the discrepancy. The interpretation of fairness is also different. We also observe that ITEM generally results in a higher aggregate throughput than APFS.

C. Credit Based algorithm (CREDIT)

In [13], a credit based algorithm (CREDIT) for simultaneous transmissions was proposed and shown to guarantee throughput, bandwidth and fairness. We assume that CREDIT selects at most one user at each epoch. The bandwidth reservation scheme starts with the negotiation of \(r_i\) between mobile user \(i\) and the network management on connection setup. The scheduler keeps track of a variable called credit, \(C_i(l)\), for each connection \(i\) at epoch \(l\) and schedules the connection with maximum credit. \(C_i(l)\) is calculated as

\[
C_i(l) = r_i k r - S_i(l), \quad i = 1, 2, \ldots, I
\]

\(C_i(l)\) represents the difference between the guaranteed transmissions and actual transmissions of user \(i\) till epoch \(l\). With identical channel conditions for all users, CREDIT and ITEM result in the same schedule.

CREDIT does not consider the channel variations for determining the schedule. Therefore, CREDIT may select user \(i\) with poor channel condition for transmission, resulting in low aggregate throughput and a large discrepancy. ITEM performs comparatively better, achieving higher aggregate throughput and lower discrepancy, since channel conditions are considered in determining the user to be scheduled.

D. Wireless Credit-Based Fair Queuing (WCFQ)

In [3], the Wireless Credit Based Fair Queuing (WCFQ) algorithm with provable statistical temporal fairness properties over both short- and long-term horizons was proposed. \(DRC_i\) is the maximum feasible rate for user \(i\). The weight \(\phi_i\) of user \(i\) represents its targeted temporal channel of the equal to \(r_i k r / DRC_i\). \(L_i\), the HOL (Head Of Line) packet length for user \(i\), is equal to \(DRC_i \tau\). \(K_i(l)\) is the credit counter of user \(i\) at \(l\). Users accumulate credits when they are not scheduled and the credit of the user scheduled is decremented. \(E_i\) and \(U_i\) are the channel condition and the estimated cost for user \(i\), respectively. They are constant over the control horizon and related by \(U_i = -\beta \log(1 - E_i)\). \(\beta\) is a tunable parameter to trade off fairness and system gain. \(\beta = 0\) and \(\beta = \infty\) refer to perfect CBFQ fairness [17] and the best-channel-condition scheduling rules, respectively. \(f_i\) denotes the user selected at epoch \(l - 1\). The user \(f_{l+1}\) is selected according to the rule

\[
f_{l+1} = \arg\min_{1 \leq i \leq I} \frac{L_i - K_i(l) + U_i}{\phi_i}
\]

The credits of the users are updated accordingly.

When the requirement of one of the users is very large compared to those of the others, the channel conditions are the same and in certain other cases, WCFQ and ITEM select the same user. WCFQ starts with an algorithm and then the corresponding system performance metrics are derived. Our approach differs in the sense that a performance metric, the distance metric measuring the aggregate discrepancy between the required and actual service received is defined first, and then an algorithm is proposed to minimize it.

E. Numerical comparison

We compare the performance of ITEM with the other algorithms in terms of the discrepancy measure and the aggregate throughput provided to the users. Lower discrepancy and higher aggregate throughput imply better performance. We assume that Rayleigh fading is perfectly compensated by fast power control. Therefore we assume that the channel is not Rayleigh fading. However, the channel seen by each user is different depending on the loss factor between the BS and the user. Discrepancy measure and the aggregate throughput are chosen as the metrics for the comparison since they provide a measure of the fairness and the system efficiency. Let \((m+1)\) be the number of rates feasible for each user. In ITEM, APFS, CREDIT and WCFQ, the number of possibilities from which \(i^*\) must be found is \(I\) at any epoch. Selecting the best rate for \(i^*\) means a further search among \((m+1)\) possibilities; this results in a search among \(I(m+1)\) possibilities at epoch \(l\). Therefore, the complexity for the entire horizon is \(I(m+1)k\) for all the algorithms. If the number of feasible rates for each user is different and \(m_i\) is the number of rates available for user \(i\), then the computational complexity for the entire horizon is \(\sum_{i=1}^{I} m_i \times k\) for all the algorithms.

Fig. 3 shows the performance comparison (in terms of the objective function value and the aggregate throughput) of ITEM with the other algorithms for \(k = 17\) and \(I = 4\). It is clearly seen that ITEM outperforms the others on both counts. Fig. 4 shows the performance comparison under different channel conditions but with same requirements. Again, ITEM has a better performance. Moreover, it is observed that the performance of ITEM when the channel conditions are bad is better than that of WCFQ under good channel conditions. For bad channel conditions, the performance of CREDIT is better than that of APFS. However, as channel conditions improve, APFS performs better.

Table II shows the objective function value and the aggregate throughput achieved with ITEM and the other scheduling algorithms. The DSCH rate sets are used for comparison. The Euclidean norm is used as the discrepancy measure. A value of \(\beta = 10000\) is used for evaluation of \(U_i\) in WCFQ. The comparisons are done for \(I = 3\). The first column shows the control horizon \(k\). The set of rates \(S_i\) available to mobile \(i\) is shown next. This is followed by mobile \(i\)’s contracted
rate $r_i$. In the next two columns, “objective function value” refers to the value of the objective function evaluated using the algorithms. The term “aggregate throughput” refers to the sum of the throughputs achieved by all users. The algorithms have been implemented in C++.

<table>
<thead>
<tr>
<th>$k$</th>
<th>$S_i$ (kbps)</th>
<th>$r_i$ (kbps)</th>
<th>WCFQ</th>
<th>APFS</th>
<th>ITEM</th>
<th>CREDIT</th>
<th>WCFQ</th>
<th>APFS</th>
<th>ITEM</th>
<th>CREDIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>(30, 120)</td>
<td>(30)</td>
<td>1.25</td>
<td>832</td>
<td>140</td>
<td>50</td>
<td>2260.17</td>
<td>26950.50</td>
<td>84256.30</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>(30, ..., 240)</td>
<td>(30, 120)</td>
<td>0.75</td>
<td>832</td>
<td>140</td>
<td>50</td>
<td>2260.17</td>
<td>26950.50</td>
<td>84256.30</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>(30, ..., 480)</td>
<td>(30)</td>
<td>2.25</td>
<td>832</td>
<td>140</td>
<td>50</td>
<td>2260.17</td>
<td>26950.50</td>
<td>84256.30</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>(30, 60)</td>
<td>(30, ..., 480)</td>
<td>1.25</td>
<td>832</td>
<td>140</td>
<td>50</td>
<td>2260.17</td>
<td>26950.50</td>
<td>84256.30</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>(30, ..., 480)</td>
<td>(30, 60)</td>
<td>0.75</td>
<td>832</td>
<td>140</td>
<td>50</td>
<td>2260.17</td>
<td>26950.50</td>
<td>84256.30</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>(30, ..., 480)</td>
<td>(30, 60)</td>
<td>1.25</td>
<td>832</td>
<td>140</td>
<td>50</td>
<td>2260.17</td>
<td>26950.50</td>
<td>84256.30</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>(30, ..., 480)</td>
<td>(30, 60)</td>
<td>0.75</td>
<td>832</td>
<td>140</td>
<td>50</td>
<td>2260.17</td>
<td>26950.50</td>
<td>84256.30</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>(30, ..., 480)</td>
<td>(30, 60)</td>
<td>1.25</td>
<td>832</td>
<td>140</td>
<td>50</td>
<td>2260.17</td>
<td>26950.50</td>
<td>84256.30</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>(30, ..., 480)</td>
<td>(30, 60)</td>
<td>0.75</td>
<td>832</td>
<td>140</td>
<td>50</td>
<td>2260.17</td>
<td>26950.50</td>
<td>84256.30</td>
<td></td>
</tr>
</tbody>
</table>

Table II: Objective function value and aggregate throughput with WCFQ, APFS, ITEM & CREDIT.

Fig. 3. Performance comparison of ITEM with other algorithms. $k = 17$, $I = 4$, $r_1 = 146.536$, $r_2 = 450.247$, $r_3 = 24.550$, $r_4 = 580.600$ (in kbps), $S_1 = \{30, 60, 120, 240\}$, $S_2 = \{30, 60, 120, 240, 480\}$, $S_3 = \{30, 60, 120\}$, $S_4 = \{30, 60, 120, 240, 480, 960\}$ (in kbps).

Fig. 4. Performance comparison under different channel conditions but same contracted rates. $k = 13$, $I = 3$, $r_1 = 230.000$, $r_2 = 550.000$, $r_3 = 200.000$ (in kbps) (i) Better channel condition: $S_1 = \{30, 60, 120, 240, 480\}$, $S_2 = \{30, 60, 120, 240, 480, 960\}$, $S_3 = \{30, 60, 120, 240\}$ (in kbps). (ii) Worse channel condition: $S_1 = \{30, 60, 120, 240\}$, $S_2 = \{30, 60, 120, 240, 480\}$, $S_3 = \{30, 60, 120\}$ (in kbps).
From the table, it is clearly seen that ITEM performs comparatively better than the other algorithms both in terms of the objective function value and the aggregate throughput in all the cases. Since the objective function is a measure of fairness, ITEM achieves a balance between fairness and efficiency.

The case for \( k = 14 \) shows that when the channel condition experienced by each user is identical, ITEM and CREDIT have the same performance. When the channel conditions are poor for certain users, the performance of CREDIT suffers. ITEM has a definite edge over CREDIT under these conditions since channel conditions are taken into account for determining the scheduling rule. Therefore, ITEM has a better performance both in terms of fairness and the aggregate throughput.

The comparison for \( k = 10 \) shows that when the channel conditions are the same for all the users and the requirements do not vary widely, all the algorithms have the same performance. The comparison for \( k = 6 \) shows that when the channel conditions are very poor for some of the users, the performances of WCFQ and CREDIT in terms of the aggregate throughput degrade a lot, while that of ITEM and APFS do not suffer so much.

Fig. 5 compares the objective function value and the aggregate throughput of ITEM with CREDIT, APFS and WCFQ for fixed \( k \) and varying \( I \). We can see that the performance of ITEM, CREDIT and APFS improve as \( I \) decreases. The aggregate throughput of WCFQ increases as \( I \) increases, but the discrepancy also increases. For the other algorithms, the performance worsens as \( I \) increases. Fig. 6 compares the objective function value and the aggregate throughput of ITEM with the other algorithms for fixed \( I \) and varying \( k \). As we increase \( k \), we find that the aggregate throughput increases but the objective function value also increases.

Table III shows the discrepancy measure with different distance metrics. The distance metrics used for comparison are 1-norm, Euclidean norm, 4-norm and the infinity norm. The various columns carry the same meaning as in Table II. The evaluations are for various control horizons, different requirements and different rate sets. As the order of the norm increases, the user whose received service is farthest from its overall requirement gets more weightage in the selection. Under these conditions, minimizing the distance measure corresponds to choosing this user, thereby implying fairer operation. It is observed from the table that the discrepancy measure decreases as we proceed from 1-norm to the infinity norm and the aggregate throughput is non-increasing from left to right of the table (from 1-norm to infinity norm). Thus, depending on whether higher aggregate throughput or more fairness is required, one can decide on the distance metric to be used.

VI. CONCLUSION

We considered the problem of scheduling a set of \( I \) streaming connections over \( k \) frames of the DSCH in UMTS networks. We introduced the discrepancy criterion utilizing a distance measure \( f(.) \) in the \( I \)-dimensional space and gave the ITEM algorithm that was optimal. Its performance in terms of running time was far superior to that of Exhaustive Search. We
also showed that ITEM can aid in CAC. Next, we compared the performance of ITEM with that of other general scheduling algorithms and showed that it was superior to them in terms of both the fairness and the aggregate throughput. With the flexibility available in the choice of distance metric for the discrepancy measure, we can choose a distance metric which balances fairness and the aggregate throughput.

ACKNOWLEDGMENT

The contribution of M. V. Panduranga Rao, PhD student, Department of Computer Science and Automation, Indian Institute of Science is acknowledged.

REFERENCES

[2] “3GPP TS 25.211 V5.0.0, Technical specification, physical channels and mapping of transport channels onto physical channels (FDD).”

APPENDIX

A. Proof of Proposition 4.2:

Let \( S = \{ c, 2c, \ldots, 2^{(t-1)}c \} \) be the rate set available for a user. Let the requirement, \( n > c \times 2^{(t-1)} \) be partitioned as:

\[
 n = \left( n_1c + n_2c \times 2 + n_3c \times 2^2 + \ldots + n_tc \times 2^{(t-1)} \right) + \Delta 
\]

where \( \Delta \) is the remaining or exceeded requirement whose magnitude is less than \( c/2 \). This is one partition of \( n \) for the value of \( \Delta \). It is assumed that \( \sum_{v=1}^{t} T_v \) is given. There may exist different partitions characterized by different values of \( n_v, v = t, t - 1, \ldots, 1 \). ITEM determines \( n_v \), such that, if \( n_{v} \geq n_{v-1}, \text{ if } n_{v} = \text{ otherwise} \neq 0 \). There cannot be a situation, where \( n_{v} < n_{v-1}, \text{ if } n_{v} \neq 0 \).
Any higher rate can be written as an integer combination of lower rates. Let $S = \{3, 6, 12, 24\}$, $n = 37$. Here, $n_4 = 1$, $n_3 = 1$, $n_2 = n_1 = 0$ and $\Delta = 1$. Therefore, $37 = 1 \times 24 + 1 \times 12 + 1$. This is the unique partition of 37 for the given values of $n_i$. If, instead of using 24 and 12, we use 6, then $37 = 6 \times 6 + 1$. As we can see, using some lower rate (6) a higher number of times, the magnitude of $\Delta$ cannot be reduced further since $1 < 3/2$. If $|\Delta| < c/2$, then using the lower rate more number of times does not decrease the magnitude of $\Delta$ further, since it is equivalent to using the higher rate less number of times. If $|\Delta| > c/2$, then using values from the set $S$, we can reduce its magnitude to a point where it is less than $c/2$. For the given values of $n_i$-s, the partition of $n$ is unique. Moreover, ITEM results in the smallest value of $\sum_{i=1}^n n_i$ among all partitions for the same $\Delta$.

Whenever negative updated requirements are encountered, it implies that the allocation exceeds the requirement. The rate which caused the updated requirement to change from positive to negative is determined (say $2^{(v-1)}c$). The positive requirement before using $2^{(v-1)}c$ is determined. Using the rate, $2^{(v-2)}c$ twice, it is checked whether the updated requirement can be made positive. If the updated requirement is still negative, then rates lower than $2^{(v-2)}c$ are used to make it positive. If positivity of updated requirements is ensured at each point, then we can apply Proposition 4.1 to prove the optimality of ITEM. Extending the argument to multiple users, it can proved that ITEM is optimal.

A similar argument can be applied to prove that ITEM is optimal when values in rate sets are in geometric progression with other integral common ratios and when they are arithmetic progression with common difference greater than the first term.

\textbf{B. Proof of Lemma 4.1:}

Let $f_i(e) = x_i(e)$ and $z_i = u_i(l) + \tau f_i(e)$. Note that $z_i = u_i(e)$ and $u_i(l) < 0$. Suppose if $u_i(l) + \tau f_i(e)/2$ is negative. The magnitude of $z_i - \tau f_i(e)/2$ would have been less than that of $z_i - \tau f_i(e)$. So, $f_i(e)/2$ would have been selected at epoch $e$ instead of $f_i(e)$. Therefore $u_i(l) + \tau f_i(e)/2$ cannot be negative. So, it has to be positive.

\textbf{C. Proof of Lemma 4.2:}

Let $f_i(e) = x_i(e)$ and $z_i = u_i(l) + \tau f_i(e)$. Selection of $f_i(e)$ at epoch $e$ implies that $3\tau f_i(e)/4 < z_i < \tau f_i(e)$. If a rate other than $f_i(e)$ has to be selected at epoch $e$, the obvious choice is $f_i(e)/2$. This is due to the fact that when multiplied by $\tau$, it is the one which is closest to $z_i$. Therefore the rate to be selected at epoch $e$ is $f_i(e)/2$.

\textbf{D. Proof of Lemma 4.3:}

$f_i(e)$ and $z_i$ are as in Lemma 4.2. If $z_i > 7f_i(e)\tau/8$, it implies that $z_i - 3\tau f_i(e)/4 > \tau f_i(e) - z_i$. Using 2 rates we should get closest to $z_i$. Applying Lemma 4.2, $x_i(e) = f_i(e)/2$. Therefore $u_i(l) = z_i - \tau x_i(e)$. Since $\tau x_i(e) - u_i(l) < u_i(l) - \tau x_i(e)/2$, $x_i(l) = x_i(e)$. $u_i(l + 1) = z_i - 2\tau x_i(e)$ and $u_i(l + 1) < 0$. In all future scheduling decisions, only rates from the least up to $x_i(l)/2$ can reduce the magnitude of updated requirements of $i^*$ further. Therefore, $S_i = \{x_i, 2x_i, \ldots x_i(l)/2\}$.

If $z_i < 7f_i(e)\tau/8$, applying a similar argument as in the previous paragraph $x_i(e) = f_i(e)/2$, $x_i(l) = f_i(e)/4$, $u_i(l + 1) > 0$ and $S_i = \{x_i, 2x_i, \ldots x_i(l)/2\}$.