Multi-Constraint Dynamic Access Selection in Always Best Connected Networks

Bo Xing and Nalini Venkatasubramanian
Donald Bren School of Information and Computer Sciences
University of California, Irvine CA 92697-3425, USA
{bxing, nalini}@ics.uci.edu

***Abstract*— In future generation networks, various access technologies, such as Wi-Fi, Bluetooth, GPRS and UMTS, etc., are simultaneously available to mobile devices. They vary in characteristics (communication range, power consumption, security, etc.) and QoS parameters (bandwidth, delay, etc.). The notion of Always Best Connected (ABC) enables people to run applications over the most efficient combination of access technologies with continuous connectivity. Access selection is the key functional block in ABC solutions, as it chooses the most suitable access networks for application traffic flows. However, it is important that access selection decisions be dynamically made, minimizing the power consumption on mobile devices while satisfying QoS requirements and user/application preferences. In this paper, we model the problem of multi-constraint dynamic access selection (MCDAS) as a variant of bin packing problem. A series of approximation algorithms derived from the First Fit Decreasing (FFD) algorithm are proposed for finding near-optimal solutions. Simulation studies show that the algorithms we propose gradually improve performance towards quasi-optimal solutions in terms of power consumption and preference satisfaction.**

I. Introduction

Traditional mobile networking applications focus on ensuring continuity of services under the assumption that devices connect to the infrastructure using a single access technology. In future generation mobile communication systems, heterogeneous access networks (Wi-Fi, Bluetooth, cellular) with different characteristics coexist in a complementary manner [1]. These access technologies vary in bandwidth, delay, communication range, power consumption, security, reliability, implementation complexity, end-user cost and several other aspects. The existence of multiple access networks opens up the possibilities for selecting appropriate access networks or sharing access networks to ensure that all applications receive acceptable Quality of Service (QoS). For example, when network condition changes, applications may be vertically handed off to other better accesses transparently. When the bandwidth in one access network is constrained, the bandwidth offered by multiple access networks can be aggregated to support broadband services. This notion of ”Always Best Connected (ABC)” enables users to remain seamlessly connected to the network in a way that best suits their application needs. The notion of ”best” is often based on a number of user and application dependent factors - such as personal preferences, device capabilities, application QoS needs, security, available network resources, and network coverage [2].

The concept of ABC is enabling a new paradigm in fourth generation mobile communication (4G) systems [3]. Recent efforts [2], [4] describe the basic components in an QoS-capable ABC solution as comprising of the following functional blocks: Access Discovery, Access Selection, AAA (Authentication, Authorization and Accounting) Support, Mobility management, Profile Handling, and Content Adaptation. Access discovery serves to discover access networks available to a terminal and to determine parameters describing the characteristics of the discovered access networks [5]. Middleware support for profile management aids in access discovery and maintenance; such profiles in addition to dynamically collected information such as residual bandwidth, delay etc. are passed to and used by an access selection algorithm. Access selection mechanisms determine which access network is best suited for a given application traffic flow [6]. Fodor et al. combine information about user required QoS and the status of the candidate accesses to perform access selection from the service providers’ point of view, maximizing overall network capacity [7]. Chebrolu and Rao address access selection from users’ perspective and describe an approach to handle the jitters incurred by transmission on multiple interfaces [8]. Given a set of predetermined flows, [9] models static access selection as a knapsack problem; whereas [10] develops policies for deciding which wireless interface to employ for a given application based on power and performance needs.

In this paper, we address the problem of dynamic access selection, wherein a sequence of un-predetermined flows start at discrete time points and an access network is selected for each of them upon arrival. Access selection decisions are based on various aspects, including application QoS requirements, access preferences (where the user/application explicitly indicates preference for a specific access network), system resource utilization bounds and device constraints. The selections are expected to reserve power on a mobile device, guarantee application QoS needs and increase request acceptance rates while accommodating user access preferences.

We model the Multi-Constraint Dynamic Access Selection (MCDAS) problem as a variant of the bin packing problem, which is NP-hard. A series of approximation algorithms derived from the well-known First Fit Decreasing (FFD) algorithm are proposed for finding near-optimal solutions. The optimizations incorporate the ability to adapt to varying load conditions as well as dynamic network parameter changes caused by device mobility. Simulation results show that the algorithms we propose effectively enhance the desired performance metrics; an analysis of the results reveals the tradeoffs that exist between the performance metrics. We also compare the proposed algorithms with a quasi-optimal off-line solution (obtained assuming full knowledge of all traffic flows for a set of inputs).

The rest of the paper is organized as follows. Part II describes the MCDAS problem and its mathematical model. We introduce a family of algorithms to address the needs of the MCDAS problem in Part III. We evaluate the performance of our proposed techniques via detailed simulation in Part IV. Part V contains a brief description of the implementation challenges and our middleware based implementation approach. We conclude with future work in Part VI.

II. Multi-Constraint Dynamic Access Selection

Mobile networking applications of the future (e.g. multiparty video games) often consist of one or more traffic flows, e.g., a video conferencing application consists of a control signaling flow and a video data flow. These flows have different requirements in terms of reliability (tolerable loss rates), QoS (end-to-end delay) and capacity needs (transmission bandwidth). The different application flows may have varying access network preferences (signalling and paging are optimal on Bluetooth while video delivery requires the bandwidth of 802.11b networks). Access preferences may be indicated explicitly by the user or implicitly by the application characteristics.

The problem of multi-constraint dynamic access selection can be stated as follows. Given a set of access networks and dynamically arriving traffic flows from mobile devices, determine a mapping of traffic flows to access networks to (a) accommodate access preferences from users/applications, (b) maximize the number of traffic flows admitted into the system, (c) minimize the power consumption cost on mobile devices while satisfying application QoS requirements. In this section, we first define the MCDAS problem, and then model it as a multi-constraint variable-sized bin packing problem, known to be NP-hard.

A. Mathematical Model

To develop a mathematical formulation for the access selection problem at hand, we use the following notational conventions and definitions. Let $F = f_1, ..., f_n$ be the set of flows for which access networks have to be determined. Let $A = A_1, ..., A_m$ be the set of access networks available.

A **flow** $f_i, (1 < i < n)$ is described using a 4-tuple (ap_i, p_i, bw_i, d_i) where ap_i represents the access preference (defined below) of flow f_i, p_i is the partitionability (defined below) of f_i, and bw_i and d_i are the bandwidth and delay requirements of f_i.

An **access network** $A_j (1 < j < m)$ is described using a 5-tuple $(BW_j, D_j, PB_j, PT_j, PR_j)$ where BW_j is the maximum bandwidth capacity of access network A_j, D_j is the maximum communication delay of A_j and PB_j, PT_j, PR_j are power consumption cost parameters defined below.

The assignment function **Asgn** maps a flow f_i to access network A_j, i.e. $Asgn(f_i) = A_j$ if f_i is assigned to A_j.

The **access preference** ap_i of a flow f_i describes which access network is preferred by a traffic flow and to what extent. A no-preference flow can be assigned to any access network and has its access preference value set to zero ($ap_i = 0$). A strong-preference flow has an integer value j assigned to its access preference parameter ($ap_i = j, 1 \leq j \leq m$), and cannot be assigned to any access network other than the preferred one A_j. A flow with medium preference for an access network k has $ap_i = -k$ ($1 \leq k \leq m$); in this case access network A_k is preferred, but other access networks are acceptable.

The **partitionability** p_i of a flow f_i is a boolean value which denotes whether a traffic flow is partitionable. If partitionable, a traffic flow can be run over multiple access networks. Partitionability only applies to medium-preference flows and no-preference flows since strong-preference flows can be assigned to only one access network. For simplicity, we assume that only medium-preference traffic flows are candidates for partitioning; furthermore, we assume that a flow can be partitioned into only two parts.

The **power consumption cost** model we use comes from [11]. It calculates the power consumption of flow f_i using access network A_j, depicted by the following equation.

$$c_i = PB_j + PT_j \times bwt_i + PR_j \times bwr_i, \qquad (1)$$

where PB_j denotes background consumption, PT_j and PR_j represent power consumed per transmission and reception unit (1 kbps) respectively, bwt_i and bwr_i are bandwidth consumed for transmission and reception by flow f_i respectively. Background consumption is the minimal power consumption caused by network interface card without data traffic of any kind. It is accounted for only once as long as the interface is active, e.g. in the traffic flow that brings up the interface (if multiple flows use the same interface) or in the only flow using the interface (if there is only one flow in the access). We assume that an interface goes down when it does not hold any traffic flow, and that interface activation and interface switch does not consume additional energy. Hence, whether an access network is considered cheaper for a specific traffic flow depends not only on its power consumption per unit bandwidth and background consumption, but also on whether it is currently active.

The **average power consumption cost** (average cost, for short) of a given configuration (set of flows assigned to access networks) is the power cost per unit bandwidth of the configuration and is calculated as follows: $\sum_1^n (c_i \times t_i)/(T \times \sum_1^n bw_i)$, where t_n is flow f_n's running time, T is the total time.

The **dissatisfaction** value for each traffic flow assignment describes the degree to which the assignment does not match the flow's access preference. The dissatisfaction value ds_i of flow f_i assigned to access network A_j is defined as follows:

$$ds_i = \begin{cases} 0, & ap_i = 0 \& Asgn(f_i) \neq \phi \\ 0, & 1 \leq ap_i \leq M \& Asgn(f_i) = A_j, ap_i = j \\ 0, & -M \leq ap_i \leq -1 \& Asgn(f_i) = A_j, ap_i = -j \\ 1, & -M \leq ap_i \leq -1 \& Asgn(f_i) = A_j, ap_i \neq -j \\ 2, & Asgn(f_i) = \phi \end{cases}$$

The **average preference dissatisfaction** (average dissatisfaction, for short) of a given configuration (set of flows assigned to access networks) is calculated as $\sum_{1}^{n} ds_i/n$. Note that a partitioned flow's dissatisfaction is obtained by averaging dissatisfaction values for the two sub-flows.

We now formally define the MCDAS problem.

MCDAS Problem Statement: Given a set of traffic flows $F = f_1, ..., f_n$ of the form $f_i(ap_i, p_i, bw_i, d_i)$ and a fixed set of access networks $A = A_1, ..., A_m$ of the form $A_j(BW_j, D_j, PB_j, PT_j, PR_j)$, find the assignments of the traffic flows F to the access networks A that minimize

(1) average cost $\sum_{1}^{n}(c_i \times t_i)/(T \times \sum_{1}^{n} bw_i)$

(2) average dissatisfaction $\sum_{1}^{n} ds_i/n$

subject to the following constraints

(1) $\sum_{i} bw_i \leq BW_j, Asgn(f_i) = A_j$

(2) $d_i \geq D_j, Asgn(f_i) = A_j$

B. A Bin Packing Formulation

The basic bin packing problem is as follows: given a list of items with sizes less than 1 and a set of bins with capacity 1, find the packing of these items into the bins without violating the capacity limit, so that the minimum number of bins are used. A bin packing problem is called on-line if every item is packed without information on subsequent items, while an off-line problem allows decisions to be made with full knowledge of all the items [12].

In our scenario, the maximum bandwidth of an access is analogous to the capacity of a bin, which varies from one access to another. The bandwidth requirement of a traffic flow is mapped to the size of an item. Other constraints are added to the basic problem, such as delay constraints and access preferences. If the accesses are sorted in increasing power cost order before allocation, seeking minimum total power is similar to finding minimum number of bins. So the MCDAS problem is modeled as a multi-constraint variable-sized bin packing problem. At a certain instance of time, it is off-line bin packing, since the traffic flows within the same application start simultaneously. Over a period of time, it is on-line bin packing, since applications start sequentially. It has been proved that bin packing problem is NP-hard, which means that the problem in question is also NP-hard [13]. Only approximation algorithms are available to find near-optimal solutions [14].

III. Proposed Algorithms

In this section, we develop a family of heuristics to address the MCDAS problem. We begin with extensions to a well-known heuristic for the off-line bin-packing problem, i.e., the First Fit Decreasing technique. We gradually introduce additional mechanisms to further optimize the access selection process and illustrate how added optimizations address specific characteristics of the ABC networking paradigm.

A. FFD (First Fit Decreasing)

First Fit Decreasing (FFD) is a well-known algorithm for solving the off-line bin packing problem. It always packs the largest item in the item list to the lowest-index bin that has enough remaining room [12]. First Fit (FF) is a simplified version of FFD for handling on-line bin packing and has been shown to have good performance [15].

We extend the basic FFD algorithm to deal with the added constraints in the MCDAS problem. The basic idea is as follows. When a selection process is triggered by the arrival of one or more traffic flows, the flows in the wait list are first sorted in decreasing preference and bandwidth order. Consequently, strong-preference flows are ahead of medium-preference flows, which in turn are ahead of no-preference ones. Within these three groups, flows with larger bandwidth needs are ahead of others, thus will be assigned earlier. As a selection starts, the sorted flows are assigned one by one depending on their access preferences as well as the conditions of the access networks. A strong-preference flow is assigned to its preferred access, as long as the access satisfies bandwidth and delay requirements. For a medium-preference flow, it is assigned to the lowest-cost access if the preferred access is not available. A no-preference flow is assigned to the lowest-cost access that has adequate residue bandwidth and satisfying delay. In short, the preferred access is always the first choice for a flow with access preference, whereas the access with lowest cost is the second choice. (The lowest-cost access for a specific flow is found by sorting the access networks in increasing estimated cost order according to the power consumption cost calculated using Equation (1)).

If a flow cannot find any access to fit in, it has to be rejected. Rejected flows with loose time constraints can be put back to the wait list for a retry, with the expectation that some flows currently running will leave shortly. However, since the possible increase in acceptance rate is not due to the access selection algorithm, retries are not used in our simulations when we evaluate the performance of the algorithms in Part IV.

The pseudo-code for our modified FFD is shown in Fig.1 (the unshaded code portion). Note that before this function is called, the flows in the wait list have been sorted in decreasing preference and bandwidth order.

B. FFDwS (FFD with Substitution)

The general strategy that FFD employs is to greedily fill the power-efficient access networks with bandwidth-hungry flows. However, it has the drawback that no-preference flows might occupy the bandwidth of low-power accesses, leading to the rejection of subsequent flows which strongly prefer these accesses. To overcome this pitfall and further reduce dissatisfaction, we optimize the selection algorithm by relocating some ongoing flows so that the ones with stronger preference can be accommodated. We call this substitution.

As illustrated by the pseudo-code in Fig.2, substitution takes place when the preferred access, m, does not have sufficient residue bandwidth to admit a strong-preference or medium-preference flow, f. For the purpose of finding a candidate for substitution, each no-preference or medium-preference flow f_{in} in m is checked to see whether f can be supported if

```
FUNCTION Select($f$) //select an access for flow $f$
SWITCH $f$'s access preference
CASE no preference:
    FOR accesses sorted in increasing cost order
        IF access $a$ meets BW and delay requirements THEN
            assign $f$ to $a$; BREAK;
CASE strong preference of access $m$:
    IF $m$ meets BW and delay requirements THEN assign $f$ to $m$;
    ELSE IF $m$ meets delay requirement THEN
        IF Substitute($m$, $f$) == true THEN assign $f$ to $m$;
CASE medium preference of access $m$:
    IF $m$ meets BW and delay requirements THEN assign $f$ to $m$;
    ELSE IF $m$ meets delay requirement THEN
        IF Substitute($m$, $f$) == true THEN assign $f$ to $m$;
        ELSE IF Partition($f$) == true THEN assign $f$ to $m$;
        ELSE
            FOR accesses sorted in increasing cost order
                IF access $a$ meets BW and delay requirements THEN
                    assign $f$ to $a$; BREAK;
    ELSE
        FOR accesses sorted in increasing cost order
            IF access $a$ meets BW and delay requirements THEN
                assign $f$ to $a$; BREAK;
IF no access network assigned THEN RETURN false;
RETURN true;
```

Fig. 1. Pseudo-Code for FFD, FFDwS and FFDwSP. 1) FFD: the part without shade; 2) FFDwS: the part without strong shade; 3) FFDwSP: all above

f_{in} leaves and whether another access can be found to hold f_{in}. The flows that satisfy both the above conditions form a candidate list; and one of them is picked to be substituted. The following list enumerates in decreasing priority order the rules we use to choose the flow to be substituted. Only if no such flow as described in a rule exists can the next rule be applied.

(i) the flow with largest bandwidth requirement among those no-preference flows that can be relocated to less power-consuming accesses.

(ii) the flow with largest bandwidth requirement among those medium-preference flows that are not in but can be relocated to their preferred accesses.

(iii) the flow with largest bandwidth requirement among those medium-preference flows that are not in their preferred accesses and can be relocated to less power-consuming accesses.

(iv) the flow with smallest bandwidth requirement among those no-preference flows that can be relocated to more power-consuming accesses.

(v) the flow with smallest bandwidth requirement among those medium-preference flows that are not in their preferred accesses and can be relocated to more power-consuming accesses.

(vi) (if f is with strong preference) the flow with largest bandwidth requirement among those medium-preference flows that are in their preferred accesses and can be relocated to less power-consuming accesses.

```
FUNCTION Substitute($m$, $f$) //find a substitution for flow $f$ in access $m$
$T$ = list of NP/MP flows in $m$ that can make $f$ supported if substituted;
$C$ = list of NP flows in $T$ that can move to lower-cost accesses;
IF $C$ == empty THEN
    $C$ += MP- flows in $T$ that can move to preferred accesses;
IF $C$ == empty THEN
    $C$ += MP- flows in $T$ that can move to lower-cost accesses;
IF $C$ != empty THEN $q$ = flow in $C$ with maximum BW;
ELSE
    $C$ += NP flows in $T$ that can move to higher-cost accesses;
    IF $C$ == empty THEN
        $C$ += MP- flows in $T$ that can move to higher-cost accesses;
    IF $C$ != empty THEN $q$ = flow in $C$ with minimum BW;
    ELSE IF $f$ is with strong preference THEN
        $C$ += MP+ flows in $T$ that can move to lower-cost accesses;
        IF $C$ != empty THEN $q$ = flow in $C$ with maximum BW;
        ELSE
            $C$ += MP+ flows in $T$ that can move to higher-cost accesses;
            IF $C$ != empty THEN $q$ = flow in $C$ with minimum BW;
IF $C$ == empty THEN RETURN false;
ELSE Remove $q$ from $m$; Select($q$);
RETURN true;
```

Fig. 2. Pseudo-Code for Substitution (NP: no-preference, MP: medium-preference, MP+: MP in preferred access, MP-: MP in non-preferred access)

(vii) (if f is with strong preference) the flow with largest bandwidth requirement among those medium-preference flows that are in their preferred accesses and can be relocated to more power-consuming accesses

These rules are designed as such based on the consideration of (a) reducing power cost as much as possible if it can be reduced, (b) increasing power cost as little as possible if it cannot be reduced while dissatisfaction is reduced. Obviously, substitution probably causes average power cost to increase; nevertheless, it helps reduce average dissatisfaction.

FFD plus substitution becomes the algorithm FFDwS. The part without strong shade in Fig.1 is the pseudo-code for FFDwS.

C. FFDwSP (FFD with Substitution and Partitioning)

When an access is not able to fully accommodate a traffic flow, partitioning the flow and supporting it using multiple accesses is an alternative of further optimization. Imagine a flow with a large bandwidth requirement, which is even larger than the maximum bandwidth that can be offered by any of the available accesses. Without partitioning, the flow has to be rejected. Partitioning helps aggregate the bandwidth provided by multiple access networks and overcome resource shortage. The problem of how to efficiently achieve bandwidth aggregation when a flow uses multiple interfaces simultaneously has been addressed by Hsieh and Sivakumar via a transport layer approach in [16]. Partitioning is also helpful when a flow can be categorized into sub-flows with varying priority, e.g.,

```
FUNCTION Partition($f$) //check if partitioning is possible for flow f
IF $f$ is partitionable
AND preferred access $m$ meets delay requirement THEN
    choose the reproduced flow's BW;
    IF an access $a$ can be found for the reproduced flow THEN
        IF cost for partitioning $<$ cost for no partitioning THEN
            partition $f$, reproducing a new flow $g$;
            assign $g$ to $a$; assign $f$ to $m$;
            RETURN true;
RETURN false;
```

Fig. 3. Pseudo-Code for Partitioning

a video encoded using a multi-layered encoding format can be split onto a reliable access for the base layer and a less reliable access for enhancement layers.

Partitioning works as shown in Fig.3. When the preferred access, m, does not have sufficient residue bandwidth to accommodate a medium-preference flow, f, and substitution is not possible for it, if f is partitionable, the possibility of performing partitioning on f is checked. For simplicity and ease of handling multi-access transfer, we only examine the possibility of running f's uplink (sub-flow for transmitting data to the other end) and downlink (sub-flow for receiving data from the other end) on two different access networks. A flow (either the uplink or the downlink), f_r, is reproduced and assigned to the available access with lowest power dissipation, m', if m' can accommodate f_r while m is able to support the remaining part of f. In this way, f with a truncated bandwidth requirement fits in its preferred access m. Whether the uplink or the downlink should be assigned to m depends on the characteristic of the flow and the bandwidth needs of the two sub-flows. It might be preferred that the more bandwidth-consuming sub-flow leaves the preferred access if it can go to a less power-consuming access.

Inherently, partitioning reduces dissatisfaction because part of the flow is assigned as preferred. However, it does not necessarily reduce power consumption cost, since otherwise the partitioned flow could be assigned to a power-efficient access as a whole. Based on this observation, we employ partitioning only when it is estimated to help consume less energy. To achieve this, we compare the estimated power cost of performing partitioning and that of not performing partitioning using Equation (1). Partitioning is performed only if the former is smaller.

Fig.1 shows the pseudo-code for FFDwSP, the algorithm with partitioning added.

D. FFDwSPL (FFD with Substitution, Partitioning and Load-Awareness)

All the algorithms described above take the risk of blocking some of the accesses, because power-efficient accesses always have higher priority in being selected, thus serving a higher load. However, in some emergency situations such as crisis rescue, it is important that certain access networks are always available. For example, it is expected that the access network for contacting rescue agencies is never blocked. Load-awareness is added to our algorithm with the intention to balance the loads on the accesses and prevent them from being blocked.

```
FUNCTION Reallocate($m$) //reallocate the flows in access $m$
FOR flows in $m$
    IF flow $f$'s delay requirement no longer met THEN
        remove $f$ from $m$ and put it to the wait list;
IF $m$'s BW not sufficient THEN
    remove all the flows in $m$ and put them to the wait list;
FOR flows $f$ in the wait list sorted in decreasing preference/BW order
    IF select($f$) == false THEN dump $f$;
```

Fig. 4. Pseudo-Code for Reallocation

In FFDwSPL, a load-aware threshold th limits the loads on the accesses to a certain extent. When f, a no-preference flow or a medium-preference flow that cannot get into its preferred access, is searching for a low-cost access in the sorted access list, if the bandwidth utilization percentage of an access would exceed th after this assignment, then f cannot be assigned to this access. However, if no access can be found for f after a round of searching, th is incremented and the search runs again till f is assigned or th reaches one hundred percent. th is reset to the initial value when the next selection starts. Note that the load-aware threshold th does not apply to a strong-preference flow or a medium-preference flow assigned to its preferred access, for the sake of respecting access preferences. So an assignment of such a flow may increase the bandwidth utilization percentage over the threshold.

The pseudo-code of FFDwSPL is similar to that of FFDwSP shown in Fig.1, except that the underlined IF statements are added with the condition that bandwidth utilization percentage would not exceed th after assignment.

Load-awareness brings advantages other than balanced loads and blocking prevention. As no access is filled up by flows with no preference, the subsequent flows with access preference are less likely to be rejected or assigned to non-preferred accesses. Also, substitution and partitioning, which might incur extra overhead, are performed less frequently. On the other hand, the downside of this mechanism is that power consumption definitely increases.

E. FFDwSPLR (FFD with Substitution, Partitioning, Load-Awareness and Reallocation)

Recall that we are dealing with a scenario in which multiple access networks are available to mobile devices. Due to the mobility of the devices, the bandwidth and delay of the access networks might be changing over time. For instance, the delay of the cellular network increases when a mobile device moves away from the base station. Here we consider the case where bandwidth shrink or delay increase occurs as well as how to adapt to the fluctuations. Responding to the changes by simply dumping the flows that do not get their requirements satisfied is undesirable, because there might be other access networks that are able to accommodate these flows.

TABLE I

SIMULATION INPUTS: POWER CONSUMPTION PARAMETERS

Access Network	PB (μW)	PT (μW/kbps)	PR (μW/kbps)
Bluetooth	118.50	18.40	9.36
IEEE802.11b	262.74	1.22	1.22
IEEE802.11a	368.38	0.32	0.14
UMTS	107.20	8.28	4.92
GPRS	313.47	0.76	0.36

Reallocation is an adaptation mechanism aimed at reducing the number of flows being dumped due to network parameter changes. It works as follows. If the delay of an access increases, all the traffic flows that are currently using this access but have more stringent delay constraints are put to the head of the wait list and reallocated. If bandwidth decrease makes an access not able to accommodate all of the flows it supports any more, access selection is re-performed on all the flows. As a consequence, some of the flows need to handoff to a different access network while some do not. If no access is available for a reallocated flow, the flow has to be dumped. This is how FFDwSPLR works to dynamically adapt to the changes of network characteristics. Fig.4 describes the function invoked when network parameter changes occur to a certain access m.

IV. PERFORMANCE EVALUATION

In order to evaluate the performance of the proposed algorithms, we conducted extensive simulations that capture the process and parameters of dynamic access selection using different algorithms. We begin by describing how access networks and traffic flows are modeled.

A. Simulation Setup

We assume that five access networks are available. Accesses 1 to 3 represent Bluetooth, IEEE 802.11b and 802.11a, while access networks 4 and 5 are intended to mimic cellular voice (UMTS) and data (GPRS) networks. The power consumption parameters we use for these 5 networks are listed in TABLE I, in which PB denotes background consumption, PT represents power consumed per unit of transmission (1 kbps), and PR represents the power consumed per unit of reception (1 kbps). The parameters of the first three accesses come from the simulation results in [11] corresponding to Bluetooth, IEEE802.11b and IEEE802.11a, respectively. The values for access networks 4 and 5 are derived from relative characteristics of UMTS and GPRS in comparison to the other three accesses. While more detailed profiling will allow us to obtain accurate measurements for the cellular networks, we believe that the parameters derived can be used without loss of generality to illustrate the impact of our heuristics. An interesting observation from TABLE I is that the power consumption behavior of access networks vary significantly - for instance, Bluetooth technology has lower background consumption values, but incur higher costs per unit of transmission/reception.

We model the arrival and departure of applications (composite traffic flows) as a Poisson Process with a mean arrival interval ($1/\lambda$) of 4 seconds and mean service time ($1/\mu$) of 4 seconds in the basic case. λ and μ are varied for different workloads using a factor we call *service intensity*, defined as the ratio of mean service time and mean arrival interval. By definition, with higher service intensity, flows arrive faster than they leave, causing the access networks to be more crowded. Service intensity has a strong impact on the performance of access selection algorithms, as will be seen in the simulation results. In our experiments, nine service intensity values were simulated. The values used for mean arrival interval and mean service time are listed in TABLE II.

TABLE II

SIMULATION INPUTS: POISSON PROCESS PARAMETERS

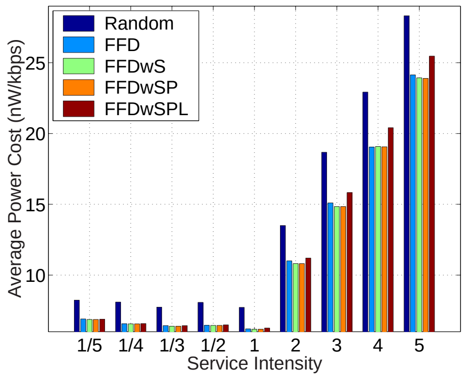
Service Intensity	1/5	1/4	1/3	1/2	1	2	3	4	5
arrival interval (s)	20	16	12	8	4	4	4	4	4
Service Time (s)	4	4	4	4	4	8	12	16	20

The number of traffic flows within a particular application is a randomly generated value between 1 and 3. Access preferences for each flow are randomly generated values between -5 and 5. As per the problem specification, a positive (negative) value indicates a strong (medium) preference for that specific access network while a value of 0 indicates no preference. Bandwidth and delay constraints for each flow are randomly generated values that are modeled in correspondence with the access preferences (e.g. a flow that strongly prefers Bluetooth has a bandwidth requirement that is smaller than the maximum bandwidth supported by Bluetooth). Partitionability (inapplicable to strong-preference and no-preference flows) values of 0 and 1 are randomly allocated for flows with medium preference. The basic simulation results do not account for dynamic network parameter changes; the impact of network variations and the parameters used therein are described in a separate subsection.

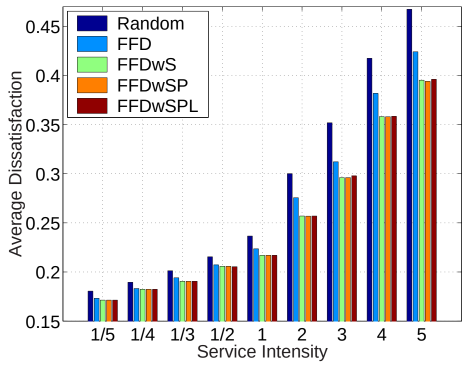
The main metrics used to evaluate the basic performance of the different heuristics are *Average Power Consumption Cost* and *Average Preference Dissatisfaction* as defined in Section II. A partitioned flow's dissatisfaction is obtained by averaging dissatisfaction values for the two sub-flows. Likewise, the dissatisfaction value for a substituted flow is calculated by averaging the dissatisfactions of the assignments for that flow. Other evaluation metrics include *Rejection rate*(the percentage of rejected incoming flows), *Dumping rate* (the percentage of admitted flows dumped during execution due to network parameter changes) and *Access bandwidth utilization standard deviation* (which depicts the extent to which loads on different access networks are balanced). Note that rejected flows are not included in the calculation of average power cost. This implies that an algorithm consuming less power might actually admit fewer flows, which is not necessarily preferred. Hence, *rejection rate* should also be an important performance metric when the effectiveness of an algorithm is measured.

B. Simulation Results

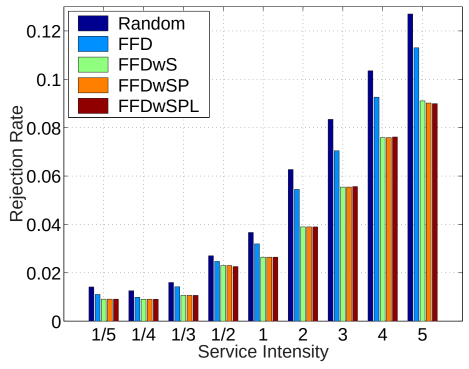
We first illustrate the basic simulation results of the MCDAS heuristics that demonstrates the impact of substitution and partitioning on performance. We then conduct experiments to



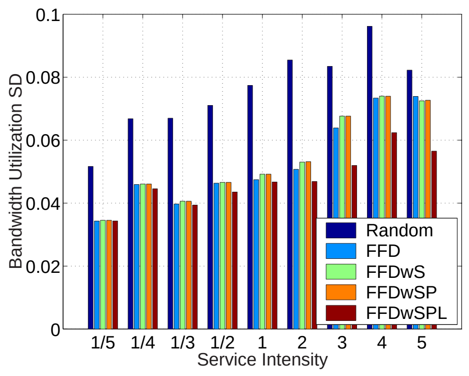
(a) Average Power Cost Vs Service Intensity



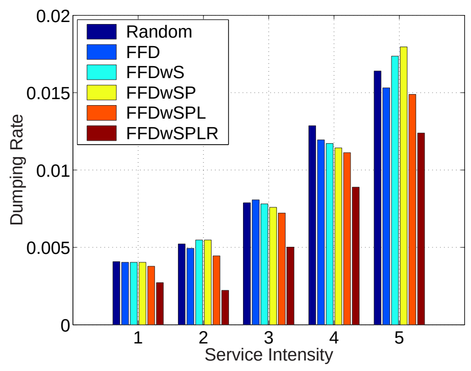
(b) Average Dissatisfaction Vs Service Intensity



(c) Rejection Rate Vs Service Intensity



(d) Bandwidth Utilization SD Vs Service Intensity



(e) Dumping Rate Vs Service Intensity

Fig. 5. Simulation Results

illustrate the value of load-awareness in FFDwSPL and the impact of the load threshold parameter th on performance. We further experiment with dynamic network parameter changes, and compare FFDwSPL and FFDwSPLR to see the effect of reallocation. A random selection algorithm is devised for comparison purposes. The randomized technique works similar to FFD except that it randomly selects an access instead of selecting the lowest-cost one when no preference is specified or the preferred access is unavailable. Random selection is expected to consume a higher power cost than the FFD variants, because FFD always puts a flow to the access with lowest power cost that is available. The simulation results are shown via a group of bar charts in Fig.5. They represent the average values of the performance metrics obtained from 10 consecutive executions of a 1000 second simulation.

1) Basic Performance of MCDAS Algorithms: Fig.5(a), 5(b) and 5(c) plot the algorithms' average power cost, average dissatisfaction and rejection rate at different service intensities, respectively, with no network parameter changes. Several observations are derived from these graphs. First of all, when service intensity is greater than one, traffic flows arrive faster than they are served, making access networks become more and more crowded. Meanwhile, average power cost, average dissatisfaction and rejection rates increase with rise in service intensity. This is because when the accesses are crowded, some of the flows have to be assigned to high-cost accesses or non-preferred accesses, and some are even rejected. When service intensity is less than 1, this tendency is not observed, simply because no access network is busy, and the mechanisms are not even performed. Second, random selection has the worst performance in all aspects at all times, while FFD, FFDwS and FFDwSP gradually decrease average power cost, average dissatisfaction and rejection rate with the best performance achieved by FFDwSP. Third, substitution and partitioning effectively improve performance, while load-awareness lowers rejection rate as a by-product because it always tries not to block the accesses. However, it can be seen that substitution has a more significant impact on performance than partitioning. There are two reasons for this behavior. Since we do partitioning conditionally, it does not take place as frequently as substitution. Moreover, partitioning usually causes accesses to be filled up, thus influencing the assignments of subsequent flows, so that the performance enhancement it brings in short term is amortized over long run.

To show how optimized the solutions found by our heuristics are, we come up with quasi-optimal solutions, which are obtained by assuming full knowledge of all traffic flows as if they were predetermined. In other words, we compare our on-line techniques to the best possible off-line technique. The quasi-optimal solution first assigns all strong-preference flows to their preferred accesses, then assigns medium-preference flows to their preferred accesses if possible, and finally assigns the no-preference flows to lowest-cost accesses. Note that the

TABLE III
NORMALIZED PERFORMANCES

	Avg Power Cost	Dissatisfaction	Rejection Rate
Random	20.78%	13.65%	43.07%
FFD	-0.90%	6.18%	22.71%
FFDwS	-1.53%	2.32%	0
FFDwSP	-1.56%	2.28%	-0.11%
FFDwSPL	1.36%	2.39%	-0.27%

quasi-optimal solution guarantees minimal average dissatisfaction, because access preference is always fulfilled in the first place.

A normalized performance describes how close the performance of an MCDAS heuristic is to the quasi-optimal solution. It is defined as in Equation (2).

$$per_{norm} = (per_{a\lg orithm} - per_{optimal})/per_{optimal} \qquad (2)$$

TABLE III shows the normalized performances for our algorithms, which are averaged over all service intensities. For average dissatisfaction, FFDwSP gets the closest performance to quasi-optimal solutions: only 2.28% larger than the optimal case, in contrast to 13.65% by random selection. We observe in Table III that some normalized performances for average power cost and rejection rate are negative. FFD incurs less power cost than quasi-optimal solutions because it denies services to more flows. Other negative normalized performances are due to the fact that substitution and partitioning are not used in quasi-optimal solutions and they typically reduce power dissipation and rejections. Substitution and partitioning provide more flexibility by running applications over different accesses at different time or at the same time. In this way, less power is consumed and more flows are admitted. However, the lower cost and better acceptance are achieved at the cost of lower preference satisfaction; the tradeoff between these metrics represents a choice for mobile applications.

2) Impact of Load-Awareness: Fig.5(d) illustrates the bandwidth utilization standard deviation for the different algorithms. Obviously FFDwSPL exhibits the best performance in terms of maintaining balanced loads on various accesses, hence reducing the probability of accesses being blocked. This is especially true when service intensity is high.

A key issue to study in our load-aware heuristic, FFDwSPL, is the impact of the threshold value th on performance. In previous experiments, FFDwSPL sets th to 50% by default, unless otherwise stated. Upon further experimentation (results not shown here due to space limitations), we notice that the value of th does not have an observable impact on average dissatisfaction and rejection rate since it only applies to no-preference flows. However, th significantly affects average power cost and bandwidth utilization standard deviation. Lower th gets more balanced loads but also higher average power cost; this effect is amplified when service intensity is greater than one. As an extreme case, when th equals 100%, it is equivalent to FFDwSP, with lowest power cost but least balanced loads. Therefore, there exists a tradeoff between low average power cost and balanced loads when FFDwSPL is employed. If power is more important, then a large th should be selected; in the extreme case one should use FFDwSP. When balanced loads and non-blocked accesses outweigh low power cost, th should be set to a small value.

3) Impact of Dynamic Network Parameter Changes: To show the performance enhancement brought by the introduction of reallocation, we evaluate the various policies under dynamic network parameter changes. The network variations modeled include the number of times changes occur, when they occur to which accesses and how much bandwidth and delay parameters changed. The key metric of interest here is the dumping rate that measures the degree to which admitted flows can be reaccommodated under changing conditions.

The simulation result is described by Fig.5(e). It is shown that reallocation effectively reduces dumping rate, i.e. FFDwSPLR always has the lowest dumping rate. However, reallocation does not completely avoid dumping. For instance, if a flow strongly preferring an access has to be reallocated due to unavailability of the access network, it will be dumped.

Fig.5(e) also illustrates another advantage of load-awareness: FFDwSPL on average has a lower dumping rate than other algorithms without the reallocation mechanism. This is because load-awareness distributes flows evenly among access networks. When a certain access's condition changes, the flows in the access that have to be dumped are fewer.

Summary of Performance Evaluation: In this section, we described the performance of the MCDAS algorithms we proposed in section III. It was shown that each of the algorithms improves performance in a certain aspect, and that tradeoffs exist between the performance metrics. In general, if balanced loads is not a major concern, FFDwSPR (FFDwSP plus Reallocation) is determined to be the best choice. From our experiments, we realized that a generally better approximation algorithm is not necessarily better at all times (the results depict the average case). After all, the scenario in question is a combination of off-line and on-line NP-hard problems, and the strategies we use are greedy. A good decision at a specific point of time may be proved a bad one in the future, while a bad decision perhaps sets a foundation for more future good decisions to be made. This is an inherent property due to the nature of the problem at hand.

V. IMPLEMENTATION ISSUES

The access selection techniques we propose can be implemented in the Dynamo middleware [17] (developed at the University of California, Irvine) on a mobile device. Dynamo is a distributed middleware framework which adaptively coordinates power management at all levels from hardware layer to user application layer so as to gain maximum power savings. As described in [18], we use multimedia applications such as video streaming to explore the performance of the cross-layer approach. Access discovery and access selection are added as adaptation measures for energy/user perception optimization in the context of multiple wireless interfaces. Assigning flows according to network parameters and interface activities aids in saving energy; however, user perception may be significantly impaired if video is transcoded and streamed via non-preferred access networks.

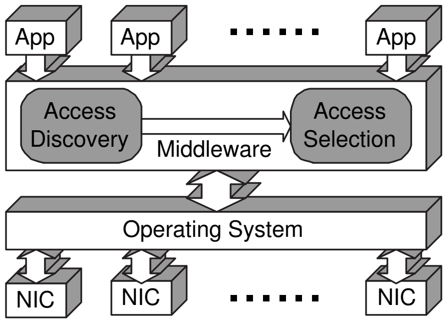


Fig. 6. Implementation System Model

As shown in the system model in Fig.6, parameters of available access networks are sensed by Network Interface Cards (NIC), and passed via operating system to the access discovery module, which in turn generates inputs for access selection. The access selection optimization mechanisms (substitution, partitioning, etc) are selectively performed at runtime, leveraging energy consumption and user experience. The selections are triggered by either the arrival of an application or changes of network conditions reported by the access discovery module. The access selection decisions are sent to the operating system, which manipulates the NICs to support traffic flows.

VI. Conclusions And Future Work

A wide variety of wireless access technologies will be simultaneously available to mobile devices in the future. In this paper, we consider the problem of multi-constraint dynamic access selection, i.e. assigning multiple traffic flows to multiple interfaces. We model it as a variant of bin packing problem, which is NP-hard. We propose a family of approximation algorithms for finding near-optimal solutions to the problem, which are intended to minimize power consumption and maximize user satisfaction. Simulation results show that each mechanism employed in the algorithms effectively contributes to performance enhancement. These mechanisms can be used in different combinations to achieve specific objectives and find the optimal operating point for both power and user experience.

In this paper, we did not consider the power consumption of vertical handoff caused by substitution as well as the handling overhead incurred by partitioning. Actually substitution and partitioning may not always be helpful in reducing the total energy consumed, however they definitely helps meet user/application preferences. Another issue is that, we only addressed the case in which access networks sporadically has their bandwidth-delay products decreased, whereas in reality, they may continuously experience dynamic changes in both directions. It would be interesting to analyze the impact of bandwidth fluctuations on the choices of access networks. The more complicated and unpredictable environment also introduces the necessity of switching ongoing traffic flows to the access networks which become more suitable after changes. Further study of these issues would be a good extension of this work.

VII. Acknowledgement

The authors gratefully acknowledge the work from the ONR (Office of Naval Research) - MURI (DoD Multidisciplinary University Research Initiative) project and the RESCUE (Responding to Crises and Unexpected Events) project, which inspire the idea of this work.

References

[1] M. Frodigh, S. Parkvall, C. Roobol, P.Johansson, and P. Larsson, “Future-generation wireless networks,” *Personal Communications, IEEE*, vol. 8, no. 5, pp. 10 – 17, Oct. 2001.

[2] E. Gustafsson and A. Jonsson, “Always best connected,” *IEEE Wireless Communications*, vol. 10, pp. 49 – 55, Feb. 2003.

[3] M. O’Droma, I. Ganchev, G. Morabito, R. Narcisi, N. Passas, S.Paskalis, V. Friderikos, A. S. Jahan, E. Tsontsis, C. F. Bader, J.Rotrou, and H. Chaouchi, “’always best connected’ enabled 4g wireless world,” in *IST Mobile and Wireless Communications Summit 2003*, June 2003.

[4] G. Fodor, A. Eriksson, and A. Tuoriniemi, “Providing quality of service in always best connected networks,” *Communications Magazine, IEEE*, vol. 41, no. 7, pp. 154 – 163, July 2003.

[5] S. Luck and A. Gutscher, “Improving access discovery by analyzing world-model information,” in *Ortsbezogene Anwendungen und Dienste, Fachgesprach der GI-Fachgruppe KuVS*, 2004.

[6] L. Suciu, J.-M. Bonnin, K. Guillouard, and B. Stvant, “’always best connected’ enabled 4g wireless world,” in *PWC*, June 2004, pp. 421 – 430.

[7] G. Fodor, A. Furuskar, and J. Lundsjo, “On access selection techniques in always best connected networks,” in *ITC Specialist Seminar on Performance Evaluation of Wireless and Mobile Systems*, Aug. 2004.

[8] K. Chebrolu and R. Rao, “Communication using multiple wireless interfaces,” in *Proceeding of the IEEE WCNC*, Mar. 2002.

[9] V. Gazis, N. Houssos, N. Alonistioti, and L. Merakos, “On the complexity of ’always best connected’ in 4g mobile networks,” in *Vehicular Technology Conference*, vol. 4, Oct. 2003.

[10] W. Qadeer, T. S. Rosing, J. Ankcorn, V. Krishnan, and G. D. Micheli, “Heterogeneous wireless network management,” in *PACS*, Dec. 2003, pp. 86 – 100.

[11] J. Lorchat and T. Noel, “Power performance comparison of heterogeneous wireless network interfaces,” in *Vehicular Technology Conference*, vol. 4, Oct. 2003.

[12] E. Coffman, J. Csirik, and G. Woeginger, “Approximate solutions to bin packing problems,” *Technical Report Woe-29, Institut fr Mathematik B, TU Graz, Steyrergasse 30, A-8010 Graz, Austria*, 1999.

[13] C. Tovey, “Tutorial on computational complexity,” *Interfaces*, vol. 32, no. 3, June 2002.

[14] E. Coffman, M. Garey, and D. Johnson, “Approximation algorithms for bin packing: A survey,” *Approximation Algorithms for NP-Hard Problems*, pp. 46 – 93, 1996.

[15] X. Gu and e. a. G. Chen, “Performance analysis and improvement for some linear on-line bin-packing algorithms,” *Journal of Combinatorial Optimization*, vol. 4, no. 6, Dec. 2002.

[16] H.-Y. Hsieh and R. Sivakumar, “A transport layer approach for achieving aggregate bandwidths on multi-homed mobile hosts,” in *MOBICOM’02*, Sept. 2002, pp. 83 – 94.

[17] S. Mohapatra. Dynamo homepage. [Online]. Available: http://www.ics.uci.edu/ mopy/research/index.html

[18] S. Mohapatra, R. Cornea, N. Dutt, A. Nicolau, and N. Venkatasubramanian, “Integrated power management for video streaming to mobile handheld devices,” in *ACM Multimedia 2003*, Nov. 2003, pp. 582 – 591.

[19] L. Feeney and M. Nilsson, “Investigating the energy consumption of a wireless network interface in an ad hoc networking environment,” in *INFOCOM*, vol. 3, Apr. 2001, pp. 1548 – 1557.