Efficient Handoff Rerouting Algorithms: A Competitive On-Line Algorithmic Approach

Yigal Bejerano, Israel Cidon, and Joseph (Seffi) Naor

***Abstract*—This paper considers the design of handoff rerouting algorithms for reducing the overall session cost in personal communication systems (PCS). Most modern communication systems that are used as an infrastructure for PCS networks are based on connection-based technologies. In these systems, the session cost is composed of two components. The setup cost represents the cost associated with the handoff operations, and the hold cost determines the expense related to the use of network resources held by the connection. This work introduces for the first time, rerouting algorithms for general graphs which are cost effective in terms of their worst-case analysis. The algorithms are analyzed using a competitive analysis approach, and it is proved that the competitive ratio of the proposed algorithms is a small constant of which the precise value depends on the ratio between the setup costs and the hold costs of the links. We also prove a lower bound of 2 on the competitive ratio of any online algorithm, which means that the proposed algorithms are close in terms of worst case behavior to the best possible rerouting algorithm. In addition, experimental results also show that the proposed algorithms indeed balance between the session setup cost and the hold cost, yielding overall lower cost when compared to other algorithms described in the literature.**

***Index Terms*—Competitive analysis, connection management, handoff rerouting algorithms, online algorithms, personal communication systems (PCS).**

I. Introduction

PERSONAL communication systems (PCS) enable people and devices to communicate independently of their location, and while they transit from place to place. Therefore, a PCS network employs a mobility management mechanism for locating mobile users and for maintaining their sessions while they change their attachment points to the system's infrastructure. Such changes are called *handoff* or *handover* operations [6]. Most modern communication systems that are used as infrastructure for PCS networks [3], [14], [15], [17], [18] are based on connection-based technologies such as telephone and ISDN technologies [19], Frame–Relay [13] and ATM networks [13]. Such networks require the establishment of a *virtual channel* (VC) between the session participants and maintaining it during the session. In PCS networks, each time a session participant performs a handoff operation, the session VC must be modified for maintaining end-to-end connectivity. Using efficient handoff rerouting algorithms is important for the efficient management of PCS networks. The criteria for evaluating such algorithms are generally classified to two components; The *setup cost* represents the cost associated with the handoff operations, in particular signaling cost and handoff latency. The *hold cost* determines the expense related to the use of network resources held by the VC. This paper introduces new handoff rerouting algorithms and demonstrates that they are quantitatively efficient for both analytical and experimental respects. The algorithms take into account both setup and hold costs. Other criteria, such as the quality-of-service capability of the VC (in terms of end-to-end delay, etc.), or the network utilization, are also considered.

The different existing algorithms can be broadly divided into four groups, 1) connection reestablishment, 2) path extension; 3) connection modification, and 4) handoff anticipation. The connection reestablishment algorithm [11] establishes a new VC between the users at each handoff operation and releases the previous VC. This algorithm optimizes the network utilization at the expense of high signaling cost and high handoff latency. The path extension algorithm [2], [3] allocates a new segment between the old and the new attachment points and connects it to the existing VC. It never tears off any established VC segment.[1] This results in a simple handoff algorithm with low handoff latency, at the expense of possibly highly stretched routes that reduce the network efficiency and the quality-of-service capability of the path. The connection modification approach uses part of the existing VC and establishes a new path between the user's new location and a *crossover switch* (COS). Algorithms using this approach differ in the way a COS is selected. In [4], [5], and [16], a single node in the VC is selected as an anchor and only the path to the anchor is modified. In [12], the selected COS is the first node on the shortest path between the users that is also included in the existing VC. In [12] and [20], the algorithm selects as a COS, the node in the existing VC that is the closest one to the user new location. The last algorithm is called *minimal path update* (MPU). The handoff anticipation approach [1], [9] establishes a multipoint VC to several adjacent nodes in anticipation of a possible handoff. This approach reduces the handoff latency at the expense of processing cost and network utilization. A summary of handoff rerouting algorithms and comparisons is given in [6]. The above algorithms attempt, in general, to optimize the session cost according to a given criterion at the expense of other criteria. Connection modification algorithms attempt to reduce the overall session cost. However, they are not analytically shown to guarantee this in all cases.

[1]In practical implementations, optimizations such as detection and releasing of routing loops are usually performed.

Manuscript received March 15, 2001; revised December 6, 2001; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor M. Zukerman. This work was supported by an Eshkol Fellowship from the Israeli Ministry of Science.

Y. Bejerano is with Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07974 USA (e-mail: bej@research.bell-labs.com).

I. Cidon is with the Electrical Engineering Department, Technion–Israel Institute of Technology, Haifa 32000, Israel (e-mail: cidon@ee.technion.ac.il).

J. (S.) Naor is with the Computer Science Department, Technion–Israel Institute of Technology, Haifa 32000, Israel (e-mail: naor@cs.technion.ac.il).

Digital Object Identifier 10.1109/TNET.2002.804833

This paper considers connection-management issues and addresses the problem of reducing the overall session cost as much as possible. Optimizing the VC route after each movement of a user is a complicated task, since the handoff algorithm does not know, either the session duration, or the future movements of the users. Thus, our problem is an on-line dynamic decision problem, where decisions are based on the current state of the network without knowledge of future events.

A common way for measuring the quality of an on-line algorithm is *competitive analysis*. Here, the costs associated with an on-line algorithm are compared with the costs expended by an optimal off-line algorithm that knows the sequence of events in advance. The maximum ratio between their respective costs, taken over all sequences, is called the *competitive ratio*. It guarantees an upper bound on the worst-case performance with respect to an optimal off-line algorithm. This study focuses on competitive analysis. In recent years, this technique was extensively used for analyzing the performance of various algorithms for different communication problems, such as call admission, circuit routing, scheduling and load balancing. Extensive surveys of this area are given in [8] and [10]. An alternative approach to measuring the quality of on-line algorithms is through *average-case* analysis which relies on some hypothesis on the distribution of the input. Each of the two approaches has clear advantages as well as limitations, and the reader is referred to [8] for a related discussion.

This paper is the first to provide a worst-case analysis of handoff rerouting problems for general communication networks. In particular, it is the first to use competitive analysis in this context. It deals with handoff rerouting algorithms that are not allowed to hold unused resources,[2] as it may hurt the network utilization and increase the call blocking probability. This paper provides lower bounds for handoff rerouting algorithms and presents two efficient algorithms which have small (and almost tight) competitive ratios.

We first consider a general model where each link e is associated with two independent weights, the setup cost, s_e, and the hold cost, h_e. We present a lower bound of $\Omega(\log n)$ on the competitive ratio for arbitrary graphs, where n is the number of nodes in the graph. The proof of the lower bound strongly uses the independence between the setup and hold costs of each edge. However, in most practical networks, there is usually a correlation between the setup cost and the hold cost of the edges. In this paper, we assume that the ratio between the hold cost and setup cost of each edge is bounded by two positive constants c_1 and c_2, such that for every link e, $c_1 \leq h_e/s_e \leq c_2$. We assume that, in practice, c_1 and c_2 are close. With this fairly realistic restriction, we prove a lower bound of 2 on the competitive ratio in the case of arbitrary graphs (where $c_1 = c_2$), and we present two simple handoff rerouting algorithms with a competitive ratio of $(2 + (c_2/c_1))$. If $c_1 = c_2$, then the competitive ratio is 3. The first algorithm combines ideas from the path extension and the connection reestablishment algorithms for balancing between the session setup and hold costs. The second algorithm is based on the connection modification approach and yields better experimental results on the average. The algorithm uses two main steps for selecting a COS. First, the new VC is calculated according to the MPU algorithm. Then, for some $\alpha > 1$, segments of the VC of total weight more then α times the weight of the shortest path between their end-nodes are replaced by the corresponding shortest paths. In addition to the analytical results, our simulations show that the proposed algorithms balance between the session setup and the hold costs and yield lower overall cost than other algorithms that are described in the literature.

This paper is organized as follows. Section II describes the network model. Section III presents the lower bounds on on-line algorithms in both the general model and the correlated model. Section IV presents two on-line handoff rerouting algorithms for the correlated model. Section V shows our simulation results and Section VI concludes the work.

II. Network Model

We assume an arbitrary connection-oriented network modeled by an undirected graph $G(V, E)$, where the nodes and edges represent communication switches and full duplex links respectively. Users are attached to the nodes and can be either static or mobile. A mobile user may move and change its attachment node. A session between two users requires the establishment of a VC between the corresponding nodes and holding it during the session. This means allocating resources at each edge over the VC path and holding them during the session. Each edge $e \in E$ is associated with a *linear cost function*, $f_e(\tau) = s_e + h_e \cdot \tau$, that defines the cost of using edge e for a duration of τ time units. The *setup cost*, $s_e \geq 0$, is the cost of allocating resources over edge e, and the *hold cost* $h_e \geq 0$, is the cost of holding these resources for a single time unit. The hold time, τ, is measured from the time the edge resources are allocated until they are released. Hence, the entire cost of a VC session of duration τ, which is routed over a path p, is given by $f_p(\tau) = \sum_{e \in p} f_e(\tau)$. We use cost as a general term, and it can capture delay, dollar cost, handoff latency, signaling cost, or an aggregation of several measures.

The graph is associated with two positive constants, c_1 and c_2, that bound the ratio h_e/s_e for every edge $e \in E$, such that $c_1 \leq h_e/s_e \leq c_2$. For a path p, let $h(p) = \sum_{e \in p} h_e$ and $s(p) = \sum_{e \in p} s_e$ be the *hold cost* and the *setup cost* of path p, respectively. For every pair of nodes u and v, let the *shortest path* between them be the path which has *minimum setup cost*. Denote this path by $p^*_{u,v}$ and its setup cost by $s^*(u, v)$. In addition, let $h^*(u, v)$ be the *minimum hold cost* between nodes u and v. Note that the hold cost of the shortest path, $p^*_{u,v}$, may be more than $h^*(u, v)$. However, if the constants c_1 and c_2 are close, then, the hold cost of the shortest path between a pair of nodes is close to the minimum hold cost.

Now, consider a session σ between two users that starts at time zero and terminates at time τ. The session is defined by a sequence of m triplets, $\sigma = \{(u_i, v_i, t_i)\}_{i=0}^{m}$, where the ith triplet represents a movement of a user from node u_i to node v_i at a time t_i. We also consider the session initialization and the termination as movements. We assume that before a session starts, both users are attached to node u_0 and at time zero one of them moves to node v_0. Similarly, the session terminates at time t_m, when one of the users moves to the node to which the other attaches, and they both do not change their attachment node anymore.

[2]In [7], we consider handoff algorithms that are allowed to hold unused resources, in particular for networks with specific topologies.

The session cost depends on the handoff rerouting algorithm used. This cost is composed of the overall setup cost and the overall hold cost. For a given handoff algorithm $\mathcal{A}$ and a session σ, the first term is denoted by Setup Cost$_{\mathcal{A}}(\sigma)$ and the second term by Hold Cost$_{\mathcal{A}}(\sigma)$. Thus

$$\text{Cost}_{\mathcal{A}}(\sigma) = \text{Setup Cost}_{\mathcal{A}}(\sigma) + \text{Hold Cost}_{\mathcal{A}}(\sigma).$$

In this paper, we consider only handoff algorithms that do not hold unused resources. Edge resources that are not included in the VC at use must be released.

III. Handoff Cost Lower Bounds

In this section, we prove lower bounds for on-line handoff rerouting algorithms in general graphs with linear hold cost functions and a positive setup cost. We prove that the competitive ratio of the best on-line algorithm is at least $\Omega(\log n)$, where n is the number of nodes in the graph. This bound holds even if all edges have the same setup cost and the mobile users are allowed to move only between adjacent nodes. We first examine some properties of optimal off-line algorithms. A handoff algorithm is called *lazy* [8], if it changes a VC route only as a response to a movement of a mobile user, and, it makes the change at the time of the movement. The following theorem can be easily verified.

Theorem 1: For general graphs and linear cost functions, there exists a lazy optimal off-line algorithm.

Theorem 2: Consider a general graph and linear cost functions. Then, the competitive ratio of the best on-line algorithm is at least $\Omega(\log n)$, where n is the number of nodes in the network.

Proof: Consider a graph with $n = 2^K$ nodes. The graph contains a full binary tree with K levels such that all of its leaves are connected to a single node v. The tree levels are numbered bottom up as depicted in Fig. 1. The edges are also assigned to levels, where the edges of level k, $1 \le k \le K$, are those that connect nodes of levels $k-1$ and k. All edges e have the same setup cost, $s_e = 1$, and all the edges of a given level k have the same hold cost h_k, where $h_1 = 1$ and $h_k = \log n \cdot \sum_{j=1}^{k-1} h_j$. Thus, the hold cost of any edge e at level k is $\log n$ times more expensive then the hold cost of the entire path from node v to a node in level $k-1$.

Suppose, in contrast, that there is an on-line algorithm $\mathcal{X}$ with competitive ratio $\gamma < \log n/2$ and consider the following session σ between a mobile user and a static user that are initially attached to nodes r and v, respectively. Since all paths between nodes r and v have the same setup and hold cost, algorithm $\mathcal{X}$ selects one of these paths and establishes a VC over this route. Without loss of generality, let this route pass through the right child of the root, r. Immediately, after the VC establishment, the mobile user moves from r to its left child u. Both the off-line and the on-line algorithms have to adapt their VC for maintaining the session. Consider first the response of OPT, the off-line algorithm. OPT routes its initial VC through node u. After the movement, OPT is only required to release its VC resources over edge (r, u) and then its VC remains optimal with respect to the shortest path between nodes u and v. However, algorithm $\mathcal{X}$ has to determine whether it extends its current VC, i.e., it allocates VC resources over the edge (r, w), or it establishes an entire new VC over one of the shortest paths that connects nodes w and v. Algorithm $\mathcal{X}$ cannot choose the VC extension option for long, otherwise the session hold cost will be more than $\log n$ times the cost of OPT. Hence, some finite time after the movement of the mobile user, algorithm $\mathcal{X}$ reroutes the session VC over one of the shortest paths between nodes u and v through one of the children of node u. Suppose, without loss of generality, that this route passes through the right child of node u. Again, immediately after the VC establishment, the mobile user moves from u to its left child w. The same process continues until the mobile user reaches one of the nodes at level 1.

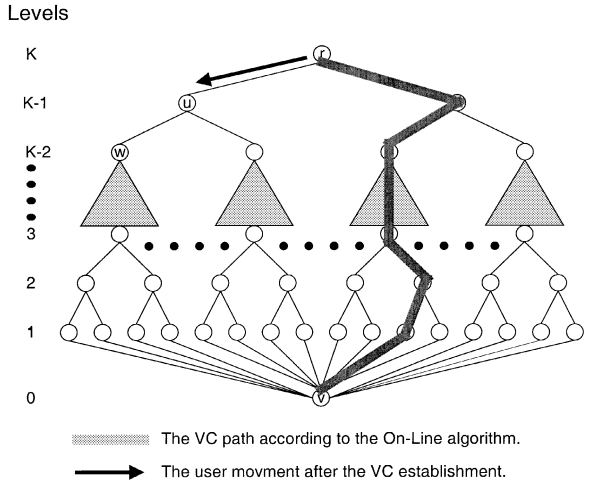

Fig. 1. The tree-like graph for proving the lower bound in the general model.

Now, compare the setup cost and the hold cost of the two algorithms. Regarding the hold cost, suppose that at time t the mobile user is attached to node x of level k. At that time, the VC route of OPT passes through one of the children of x. Therefore, its hold cost is Hold Cost$_{\text{OPT}}(t) = \sum_{j=1}^{k} h_j$. However, the VC route of algorithm $\mathcal{X}$ passes through the parent of node x at level $k+1$. Thus

$$\begin{aligned}\text{Hold Cost}_{\mathcal{X}}(t) &= \sum_{j=1}^{k} h_j + 2 \cdot h_{k+1} \\ &= (1 + 2 \cdot \log n) \cdot \sum_{j=1}^{k} h_j \\ &\ge \gamma \cdot \text{Hold Cost}_{\text{OPT}}(t).\end{aligned}$$

We now turn to calculate the setup cost. Since OPT knows the mobile user movements in advance, it establishes a single VC only at the session initialization between nodes r and v and after each movement it just releases unused edge resources. This VC contains $K = \log n$ edges, thus Setup Cost$_{\text{OPT}}(\sigma) = \log n$. In contrast, algorithm $\mathcal{X}$ is required to establish a new VC following each movement of the mobile user. Hence, its setup cost is

$$\begin{aligned}\text{Setup Cost}_{\mathcal{X}}(\sigma) &= \sum_{j=1}^{\log n} j = \frac{\log n \cdot (\log n + 1)}{2} \\ &\ge \frac{\log n}{2} \cdot \text{Setup Cost}_{\text{OPT}}(\sigma).\end{aligned}$$

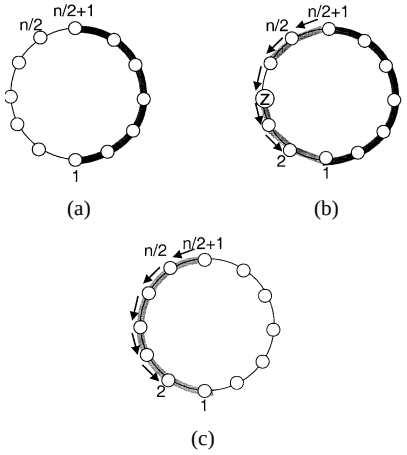

Fig. 2. The ring for the lower bound proof. (a) The initial VC of the on-line alg'X. (b) All the allocated edges according to the on-line alg' X. (c) The VC of the off-line alg' OPT.

Contradicting the assumption that the competitive ratio of algorithm $\mathcal{X}$ is $\gamma < \log n/2$. □

Theorem 2 strongly uses the independence between the setup and hold costs of each edge. As mentioned before, in this work we assume that there is a correlation between the setup cost and the hold cost of the edges which is bounded by two constants that are close to each other. We first prove a lower bound on the competitive ratio in this case.

Theorem 3: If the setup cost and hold cost are correlated, then, the competitive ratio of any on-line algorithm is at least 2.

Proof: We assume in this proof that $c_2 = c_1$. Suppose, in contrast, that there is an on-line algorithm $\mathcal{X}$ with a competitive ratio $\gamma < 2$. Consider a ring with n nodes, where n is an even number greater then $2/(2-\gamma)$, and both the setup and hold costs of each edge are 1. Clearly, $(2 \cdot (n-1))/n > \gamma$. Let σ be a session between a mobile and a static user which are initially at distance $n/2$ away from each other, e.g., at nodes $n/2+1$ and 1, respectively, as depicted in Fig. 2. During the session initialization, algorithm $\mathcal{X}$ establishes a VC connecting the two users, and suppose that the VC is routed through the right side of the ring. Immediately after the VC setup the mobile user moves to node $n/2$, and continues its movement until it reach node 2 through its left neighbor. We assume that the mobile user moves fast enough so that the session duration and hold cost are negligible.

Let us turn to calculate the setup cost of Algorithm $\mathcal{X}$. Initially, it uses the path-extension method until the mobile user reaches some node z, where it decides to reroute the session VC through the left side of the ring. Thus, its setup cost is at least $(n-1)$. Note that node z may be any node in the left side of the ring including nodes 2 and $n/2$. Concerning the off-line algorithm OPT, since it knows the mobile user movements in advance, it routes the session VC through the left side of the ring. After each movement it only releases unused VC resources without allocating new ones. Therefore, its setup cost is $n/2$. The competitive ratio of Algorithm $\mathcal{X}$ is

$$\frac{\text{Cost}_{\mathcal{X}}(\sigma)}{\text{Cost}_{\text{OPT}}(\sigma)} = \frac{n-1}{n/2} > \gamma$$

in contrast to the above assumption. □

Upon the session initialization between nodes u and v do:
Establish a VC over the path $p^*_{u,v}$

Upon a movement from node w to u when the second user is attached to node v do:
$p \leftarrow p_{w,v} \cup p^*_{w,u}$
If $(s(p) \le \alpha \cdot s^*(u,v))$ then
  Establish a VC over the path $p^*_{w,u}$
  Add $p^*_{w,u}$ to the VC path $p_{u,v}$
Else
  Release the current VC
  Establish a new VC over the path $p^*_{u,v}$

Fig. 3. A formal description of Algorithm $\mathcal{A}$.

IV. Competitive On-Line Algorithms

In this section we present two lazy on-line algorithms for arbitrary graphs which have a competitive ratio of $1+2\cdot(c_2/c_1)$.

A. Algorithm $\mathcal{A}$

1) A Description of the Algorithm:

The first algorithm, which we denote by $\mathcal{A}$, balances between the path extension and the connection reestablishment algorithms as follows. During the session initialization it establishes a VC over the shortest path between the users. Now, suppose that during a session one of the users moves from node w to node u, while the other user is attached to node v. The algorithm finds the path p which is obtained by concatenating the shortest path between nodes w and u, $p^*_{w,u}$, to the current VC. If the setup cost of p is not more than α times the setup cost of the shortest path between the two users, $s^*(u, v)$, then the path $p^*_{w,u}$ is established and it becomes part of the VC route. Otherwise, the current VC is released and a new VC over the shortest path $p^*_{u,v}$ is established. A formal description of the algorithm is given in Fig. 3, where $p_{w,v}$ is the VC path before the movement.

The algorithm uses the *credit principle*. It attempts to minimize the setup cost of each handoff operation under the constraint that the VC total setup cost is at most α times the setup cost of the shortest path between the users. We call α the *credit parameter*. The credit principle guarantees that the hold cost of $\mathcal{A}$ will not exceed $(c_2/c_1)\cdot\alpha$ times the hold cost of OPT. In the sequel we show that a proper selection of the parameter α yields a small competitive ratio.

2) The Competitive Ratio of the Algorithm:

We turn to prove that the competitive ratio of the algorithm is $1+2\cdot(c_2/c_1)$. Consider a session σ that starts at time zero and is defined by a sequence of m triplets, $\sigma = \{(w_i, u_i, t_i)\}_{i=0}^{m}$, where the ith triplet represents a movement of a mobile user from node w_i to node u_i at time t_i, as described in Section II.

Lemma 1: For every session σ,

$$\text{Hold Cost}_{\mathcal{A}}(\sigma) \le \frac{c_2}{c_1}\cdot\alpha\cdot\text{Hold Cost}_{\text{OPT}}(\sigma).$$

Proof: For every edge $e \in E$, $c_1 \le h_e/s_e \le c_2$. Therefore, for every pair u and v, $s^*(u, v) \le h^*(u, v)/c_1$. In addition, for every path p, $h(p) \le c_2 \cdot s(p)$. By the credit principle, at any time during the session, the VC route, p,

satisfies $s(p) \le \alpha \cdot s^*(u, v)$, where the users are attached to nodes u and v. Hence

$$h(p) \le c_2 \cdot s(p) \le c_2 \cdot \alpha \cdot s^*(u, v) \le \frac{c_2}{c_1} \cdot \alpha \cdot h^*(u, v)$$

proving the lemma. □

Consider the ith movement from node w_i to node u_i. Let $s(M_i) = s^*(w_i, u_i)$ be the setup cost of the shortest path between these nodes, called the *movement cost*, and let $s(M) = \sum_{i=0}^{m} s(M_i)$.

Lemma 2: For every session σ

$$\text{Setup Cost}_{\text{OPT}}(\sigma) \ge \frac{s(M)}{2}.$$

Proof: First, assume that OPT pays for allocating the resources of an edge in two installments: the first half is paid for at the time of allocation, and the second half is paid for when the edge resources are released. Now consider a user's movement from node w_i to node u_i, and let us bound its contribution to the total setup cost of OPT. As a result of this movement, part of the VC route between node w_i and some node x is released, and a new path is established between nodes x and u_i. Hence, the setup cost of this movement is at least $(s^*(w_i, x) + s^*(x, u_i))/2$. By the triangle inequality, we get that

$$s^*(w_i, x) + s^*(x, u_i) \ge s^*(w_i, u_i).$$

Therefore, the total cost is

$$\text{Setup Cost}_{\text{OPT}}(\sigma) \ge \sum_{i=0}^{m} \frac{s^*(w_i, u_i)}{2} = \frac{s(M)}{2}.$$ □

Lemma 3: For every session σ,

$$\text{Setup Cost}_{\mathcal{A}}(\sigma) \le \frac{\alpha}{\alpha - 1} \cdot s(M).$$

Proof: We partition the session into phases so as to calculate the total setup cost of the session. The first phase, called phase 0, begins at the session initialization. A new phase begins each time the algorithm decides to release the current VC and to establish a new one over the shortest path. Suppose that the session contains K connection reestablishment operations ($K + 1$ phases). Let $s(p_k)$ be the setup cost of the VC path that is established at the beginning of phase k, and let $s(M_k)$ be the sum of all the movement costs that are made during phase k. Note that $s(p_0) = 0$, since we consider the session initialization as a movement. According to the credit principle

$$\begin{aligned} s(p_k) &\le \frac{1}{\alpha} \cdot [s(M_{k-1}) + s(p_{k-1})] \\ &\le \frac{1}{\alpha} \cdot s(M_{k-1}) + \frac{1}{\alpha^2} \cdot [s(M_{k-2}) + s(p_{k-2})] \\ &\le \frac{1}{\alpha} \cdot s(M_{k-1}) + \frac{1}{\alpha^2} \cdot s(M_{k-2}) + \frac{1}{\alpha^3} \\ &\quad \cdot [s(M_{k-3}) + s(p_{k-3})] \\ &\le \frac{1}{\alpha} \cdot s(M_{k-1}) + \frac{1}{\alpha^2} \cdot s(M_{k-2}) \\ &\quad + \frac{1}{\alpha^3} s(M_{k-3}) + \cdots + \frac{1}{\alpha^k} \cdot s(M_0) \\ &\le \sum_{j=0}^{k-1} \frac{1}{\alpha^{k-j}} \cdot s(M_j). \end{aligned}$$

Thus, the total cost of the VCs that are established at the connection reestablishment operations is

$$\begin{aligned} \sum_{k=1}^{K} s(p_k) &= \sum_{k=1}^{K} \sum_{j=0}^{k-1} \frac{1}{\alpha^{k-j}} \cdot s(M_j) = \sum_{j=0}^{K-1} s(M_j) \cdot \sum_{k=1}^{K-j} \frac{1}{\alpha^k} \\ &\le \left(\sum_{j=0}^{K-1} s(M_j) \right) \cdot \left(\sum_{k=1}^{\infty} \frac{1}{\alpha^k} \right) \le \frac{1}{\alpha - 1} \cdot s(M). \end{aligned}$$

Hence, the total setup cost is

$$\begin{aligned} &\text{Setup Cost}_{\mathcal{A}}(\sigma) \\ &\le \sum_{k=0}^{K} [s(p_k) + s(M_k)] \le \sum_{k=0}^{K} s(p_k) + \sum_{k=0}^{K} s(M_k) \\ &\le \frac{1}{\alpha - 1} \cdot s(M) + s(M) \le \frac{\alpha}{\alpha - 1} \cdot s(M). \end{aligned}$$ □

Theorem 4: Algorithm $\mathcal{A}$ is $(2 + (c_2/c_1))$-competitive for

$$\alpha = 1 + 2 \cdot \frac{c_1}{c_2}.$$

Proof: The total cost of a session σ is the sum of two components, the setup cost and the hold cost. According to Lemma 1, the competitive ratio of the hold cost is

$$\frac{\text{Hold Cost}_{\mathcal{A}}(\sigma)}{\text{Hold Cost}_{\text{OPT}}(\sigma)} \le \frac{c_2}{c_1} \cdot \alpha.$$

According to Lemmas 2 and 3, the competitive ratio of the setup cost is

$$\frac{\text{Setup Cost}_{\mathcal{A}}(\sigma)}{\text{Setup Cost}_{\text{OPT}}(\sigma)} \le \frac{\frac{\alpha}{\alpha - 1} \cdot s(M)}{\frac{1}{2} \cdot s(M)} = \frac{2 \cdot \alpha}{\alpha - 1}.$$

The value of α that minimizes the competitive ratio of both components is obtained from the equation $(2 \cdot \alpha)/(\alpha - 1) = (c_2/c_1) \cdot \alpha$. Hence, the best competitive ratio is obtained by setting $\alpha = 1 + 2 \cdot (c_1/c_2)$, and its value is $2 + (c_2/c_1)$. □

Corollary 1: If $c_2 = c_1$, then Algorithm $\mathcal{A}$ is 3-competitive (for $\alpha = 3$) for general graphs.

B. Algorithm $\mathcal{B}$

1) A Description of the Algorithm: The second algorithm, which we denote by $\mathcal{B}$, uses the connection modification approach for improving the first algorithm in terms of the session cost. It is based on the following two improvements in the selection of the COS at each handoff operation.

The first improvement is achieved by selecting the COS according to *MPU*. The selected COS is the node on the existing VC which is the closest to the user new attachment node. This is the cheapest modification of the existing VC and Fig. 4 demonstrates that such a selection always yields lower setup and hold costs compared with the path extension algorithm.

The second improvement is achieved by removing exceptionally "heavy" segments from an MPU VC. A segment $p_{a,b}$ between nodes a and b is called an *exceptional segment* if its cost is at least α times the setup cost of the shortest path, i.e., $s(p_{a,b}) \ge \alpha \cdot s^*(a, b)$. If this happens, then the algorithm checks if there are exceptional segments that contain the selected COS, and replaces the most expensive exceptional

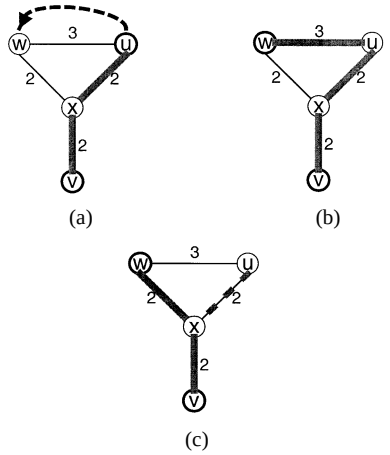

Fig. 4. Selecting the VC route according to the path extension and the MPU algorithms. (a) The initial VC. (b) The path extension routing decision. (c) The MPU routing decision.

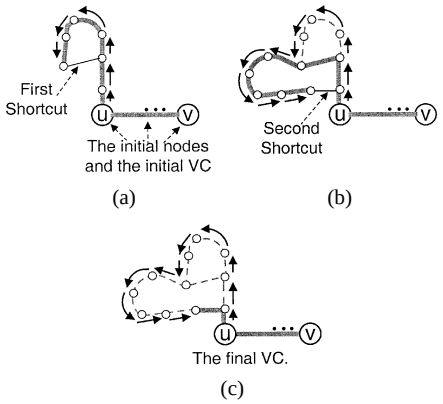

Fig. 5. An example of session with exceptional segment removal operations.

Upon a session initialization between nodes u and v do:
Establish a VC over the path $p^*_{u,v}$

Upon a movement from node w to u when the second user is attached to node v do:
$x \leftarrow arg\_min_{x \in p_{w,v}}\{s^*(u,x)\}$
$\tilde{p}_{u,v} \leftarrow p^*_{u,x} \cup p_{x,v}$.
$(a,b) \leftarrow arg\_max_{a \in p^*_{u,x}, b \in p_{x,v}, s(\tilde{p}_{a,b}) \geq \alpha \cdot s^*(a,b)}\{s(\tilde{p}_{a,b})\}$
If $(\tilde{p}_{a,b} \neq \emptyset)$ then
$B \leftarrow s^*(u,a) + s^*(a,b)$
$c \leftarrow arg\_min_{c \in p_{x,v}, s^*(u,c) \leq B}\{s^*(u,c) + s(p_{c,v})\}$
$\tilde{p}_{u,v} \leftarrow p^*_{u,c} \cup p_{c,v}$
Allocate the missing edge resources in $\tilde{p}_{u,v}$.
Release unused edge resources.
Route the VC over the path $\tilde{p}_{u,v}$.

Fig. 6. A formal description of Algorithm $\mathcal{B}$.

segment by a shortest path, called a *shortcut*. This improvement considerably reduces the VC cost by establishing low cost shortcuts and releasing unused resources. It is especially useful for users which have local movement patterns, as described by Fig. 5. In this figure the users are initially attached to the nodes u and v. After the session initialization, the mobile user from node u moves around node u, and creates exceptional segments along its way. Each time such a segment is detected it is replaced in the VC by a shortcut. Thus, the VC length during the session is kept close to optimal.

So far we have described the two improvements as two separate steps which are employed sequentially. However, a better COS that further reduces the VC cost is as follows. For each handoff operation we allocate a budget equal to the cost of the allocated paths according to both of the above improvements. This budget upper bounds the cost of the allocated path. It is used for finding the COS that further reduces the VC cost as much as possible while satisfying the budget constraint. Note that if an exceptional segment is not found, then the selected path is the same as the path selected by an MPU decision. Otherwise, if an exceptional segment exists after the MPU decision, then, it is removed at the end of this stage.

The final algorithm works as follows. At the session initialization, it establishes a VC over the shortest path between the session users. Now, suppose that one of the users moves from node w to node u, while the other user is attached to node v, and let $p_{w,v}$ be the VC path before the movement. The algorithm uses three steps for calculating the new path for the VC, $\tilde{p}_{u,v}$. In Step 1, it finds the node $x \in p_{w,v}$ which is the closest one to node u, $x = \arg\_\min_{x \in p_{w,v}} \{s^*(u, x)\}$. This function computes the value of x that minimizes $s^*(u, x)$. Let $\tilde{p}_{u,v}$ be the path that is obtained by concatenating the VC segment between nodes v and x with the shortest paths between nodes x and u, $\tilde{p}_{u,v} = p^*_{u,x} \cup p_{x,v}$. In Step 2, the algorithm finds the most expensive exceptional segment in the path $\tilde{p}_{u,v}$ that includes node x, denoted by $\tilde{p}_{a,b}$. In Step 3, the algorithm selects a COS. If such an exceptional segment was found, then let the handoff budget be $B = s^*(u, a) + s^*(a, b)$. The algorithm selects a COS, c, that minimizes the VC total weight, $s^*(u, c) + s(p_{c,v})$, under the budget constrain, $s^*(u, c) \leq B$. The received path is $\tilde{p}_{u,v} = p_{u,c} \cup p^*_{c,v}$. Otherwise, node x remains the COS. Finally, the algorithm allocates the missing edge resources in the received path $\tilde{p}_{u,v}$, routes the session VC over this path, and releases unused resources. A formal description of Algorithm $\mathcal{B}$ is given in Fig. 6.

Fig. 7 provides an example of handoff rerouting operation according to Algorithm $\mathcal{B}$ with $\alpha = 3$. Initially, the users are attached to nodes w and v. Then, the user from node w moves to node u [Fig. 7(a)]. In Step 1, the algorithm adds the path (x, z, a, u) to the existing VC and let $\tilde{p}$ be the resulting path [Fig. 7(b)]. This path contains two exceptional segments, $\tilde{p}_{y,z}$ and $\tilde{p}_{a,b}$, where $s(\tilde{p}_{y,z}) = 120$ and $s(\tilde{p}_{a,b}) = 200$ [Fig. 7(c)]. Note that node u by itself is *not* included in any legitimate exceptional segment. The most expensive segment is $\tilde{p}_{a,b}$. Therefore, the handoff operation budget is $B = s^*(u, a) + s^*(a, b) = 60 + 60 = 120$. The node that minimizes the VC cost under the budget constraint is node c and thus the final VC cost is 150 [Fig. 7(d)].

Theorem 5: Let $\alpha = 1 + 2 \cdot (c_1/c_2)$. Then, Algorithm $\mathcal{B}$ is $(2 + (c_2/c_1))$-competitive.

Corollary 2: For general graphs with $c_1 = c_2$, Algorithm $\mathcal{B}$ is 3-competitive for $\alpha = 3$.

We defer the proof of the above theorem to the Appendix. Moreover, our simulations show that Algorithm $\mathcal{B}$ achieves better results on the average than Algorithm $\mathcal{A}$.

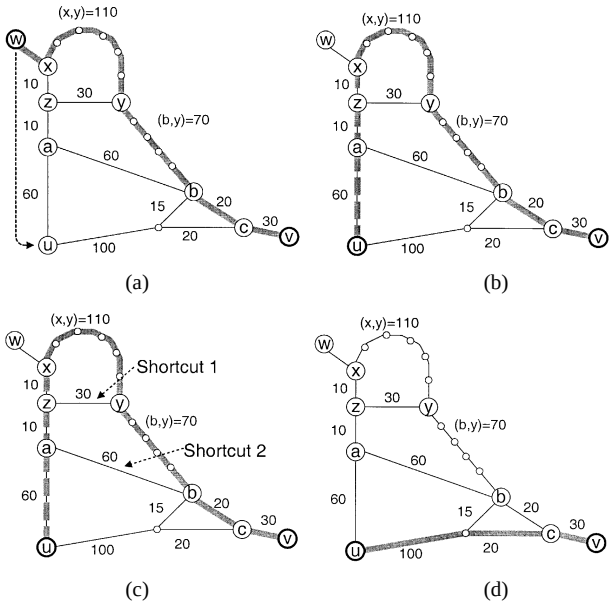

Fig. 7. An example of handoff operation according to Algorithm $\mathcal{B}$. (a) The initial VC. (b) The VC after the MPU decision. (c) Finding the longest exceptional segement (2). (d) The final VC.

V. Performance Comparison With Other Algorithms

We compared the competitive ratios as well as the performance of our handoff algorithms with four other handoff rerouting algorithms. The evaluated algorithms are the connection reestablishment algorithm [11], the path extension algorithm [2], [3] and two connection modification algorithms: the MPU algorithm [20], and the anchor rerouting algorithm [4]. For the latter, the initial location of the mobile user is selected as an anchor (a permanent COS) and only the path to the anchor is modified. We also evaluated the performance of Algorithms $\mathcal{A}$ and $\mathcal{B}$, that are described in Section IV.

In Section V-A, we show that the competitive ratios of all the other algorithms is at least $\Omega(n)$ even when the setup cost and hold cost are correlated. Where n is the number on nodes in the network graph. In Section V-B, we compare by simulations the performance of the considered algorithms in average sense.

A. Competitive Ratio Evaluation

In the following we divide the other handoff rerouting algorithms into two groups. i) *Hold-cost-minimization* algorithms that minimize the session hold cost, ignoring its setup cost. This group contains the connection reestablishment algorithm [11]. ii) *setup-cost-minimization* algorithms that minimize the session setup cost without considering its hold cost. Such algorithms are the path extension algorithm [2], [3], the MPU algorithm [20], and the anchor rerouting algorithm [4]. Recall that the proposed Algorithms $\mathcal{A}$ and $\mathcal{B}$ are not included in neither of these groups.

Theorem 6: The competitive ratio of any hold-cost-minimization algorithm is at least $n/2$, where n is the number of nodes in the network, even if the setup cost and hold cost are correlated.

Proof: Consider the network $G(V, E)$ that is described in Fig. 8, where for every edge $e \in E$, $s_e = h_e$. For every $i \in [2 \cdots n]$, let $S_{1,i} = (i-1)$ and for every $i \in [3 \cdots n]$, let $S_{i-1,i} = 1 + \epsilon$. We assume a session σ between two users

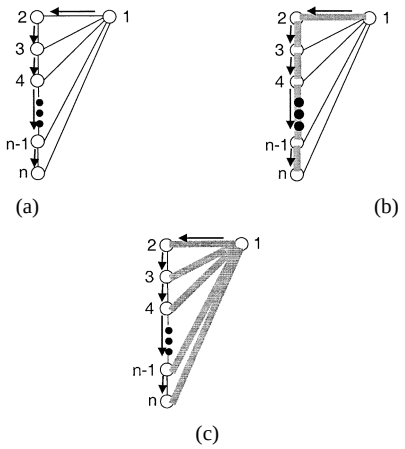

Fig. 8. The graph $G(V, E)$ used by Theorem 6. (a) The considered graph. (b) The VC of the off-line alg' OPT. (c) All the allocated VCs by the on-line alg' X.

that are located in node 1. Immediately after the session initialization, one of the users moves from node 1 to node n through the path $1, 2, 3, \ldots, n$. As soon as the user reaches node n, the session terminates. We assume that the mobile user moves fast enough so that the session duration and hold cost are negligible. Now, consider the best hold-cost-minimization algorithm, denoted by $\mathcal{X}$. After each movement from node $i-1$ to node i the algorithm releases the current VC and establishes a new one between the nodes 1 and i. The cost of this setup operation is $(i-1)$, thus the setup cost of the algorithm is Setup $\text{Cost}_{\mathcal{X}}(\sigma) = (n \cdot (n-1))/2$. However, the optimal off-line algorithmOPT extend the current VC after each movement. Thus, its setup cost is Setup $\text{Cost}_{\text{OPT}}(\sigma) = (n-1) \cdot (1+\epsilon)$. As a result

$$\frac{\text{Cost}_{\mathcal{X}}(\sigma)}{\text{Cost}_{\text{OPT}}(\sigma)} = \frac{n \cdot (n-1)/2}{(n-1) \cdot (1+\epsilon)} = \frac{n/2}{1+\epsilon}.$$

Since, ϵ may be any small positive value, the lower bound is at least $n/2$. □

Corollary 3: The competitive ratio of the connection reestablishment algorithm is at least $n/2$, where n is the number of nodes in the network.

Proof: The connection reestablishment algorithm will make the same routing decisions that the best setup-cost-minimization algorithm has made in the example given in the proof of Theorem 6. □

Theorem 7: The competitive ratio of any hold-cost-minimization algorithm is at least n, where n is the number of nodes in the network, even if the setup cost and hold cost are correlated.

Proof: Consider a ring network $G(V, E)$ as depicted in Fig. 2, with even number of nodes n and for every edge $e \in E$, $s_e = h_e$. Let $S_{1,n} = 1 + \epsilon$ and let the setup cost of the other edges $e \in E - \{(1, 2)\}$ be $S_e = 1$. We assume a session σ between two users that are located at the nodes 1 and $n/2 + 1$. Immediately after the session initialization, the user from node $n/2 + 1$ moves to node 2 through its lest neighbor. We assume that the mobile user moves fast enough so that movement time from node $n/2 + 1$ to node 2 is negligible and the user stays at node 2 for a very long time. Recall that the setup cost of the edges between nodes 1 and $n/2 + 1$ along the left side of the ring (the path $1, 2, \ldots, n/2 + 1$) is $n/2 + \epsilon$ while

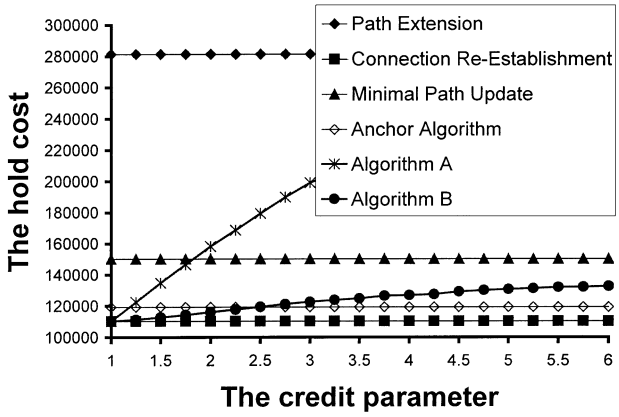

Fig. 9. The effect of the credit parameter, α, on the hold cost.

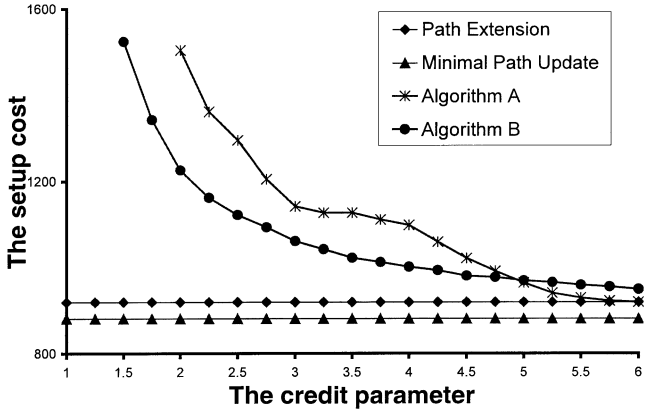

Fig. 10. The effect of the credit parameter, α, on the setup cost.

the cost of the path along the right side is $n/2$. Now, consider the best setup-cost-minimization algorithm, denoted by $\mathcal{X}$. At the beginning, it establish the VC a long the right side of the ring, and after each movement from node $i-1$ to node i, $i \in [n/2+2, n]$, the algorithm adds the edge $(i-1, i)$ to the existing VC. Thus, Setup $\text{Cost}_{\mathcal{X}}(\sigma, t) = n-1$ and the hold cost until time t is Hold $\text{Cost}_{\mathcal{X}}(\sigma) = (n-1)\cdot t$. Let us turn to describe the routing decisions of the off-line algorithm OPT. Algorithm OPT routes the VC along the left side of the ring, and after each movement it removed the unused edge. Thus, Setup $\text{Cost}_{\text{OPT}}(\sigma) = n/2+\epsilon$ and the hold cost until time t is Hold $\text{Cost}_{\text{OPT}}(\sigma, t) = (1+\epsilon)\cdot t$. Consequentially

$$\frac{\text{Cost}_{\mathcal{X}}(\sigma, t)}{\text{Cost}_{\text{OPT}}(\sigma, t)} = \frac{(n-1)+(n-1)\cdot t}{n/2+\epsilon+(1+\epsilon)\cdot t} \simeq n.$$

Thus, for large t and small ϵ the ratio is approximately n. □

Corollary 4: The competitive ratios of the path extension, the MPU the anchor rerouting algorithms are at least n, where n is the number of nodes in the network.

Proof: These algorithms will make the same routing decisions that the best setup-cost-minimization algorithm has made in the example given in the proof of Theorem 7. □

The above theorems show that all the other evaluated schemes, beside Algorithms $\mathcal{A}$ and $\mathcal{B}$, may make pure routing decisions that yield high session cost with respect to the cost of the optimal algorithm.

B. Simulation Results

We compare, using simulations, the performance of our handoff algorithms with respect to the other schemes in average sense. Our simulations considered different networks, different roaming distances and various initial distances between the users. For Algorithms $\mathcal{A}$ and $\mathcal{B}$ we also evaluated the affect of different credit parameter values, α, on their performance.

Selected typical results from our experiments are described in Figs. 9–12. In this example, the tested communication network is a grid graph, where both the setup cost s_e, and the hold cost, h_e, of each edge $e \in E$ are 1. We evaluated the average setup and hold costs of sessions with the following characteristics. The initial distance between the users is 200 edges, where one of them is static and the other is mobile. The mobile users

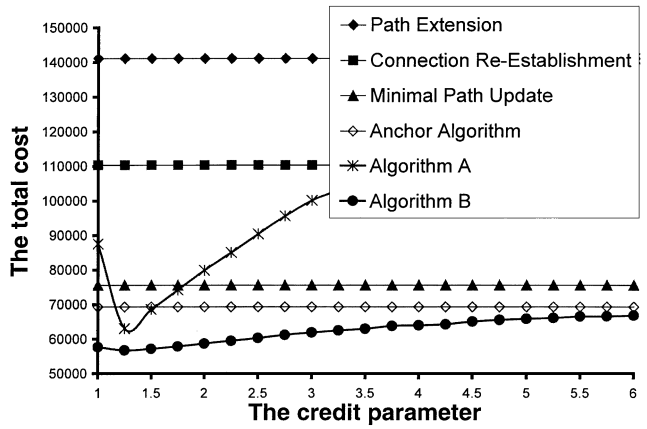

Fig. 11. The effect of the credit parameter, α, on the session total cost with $\beta = 0.5$.

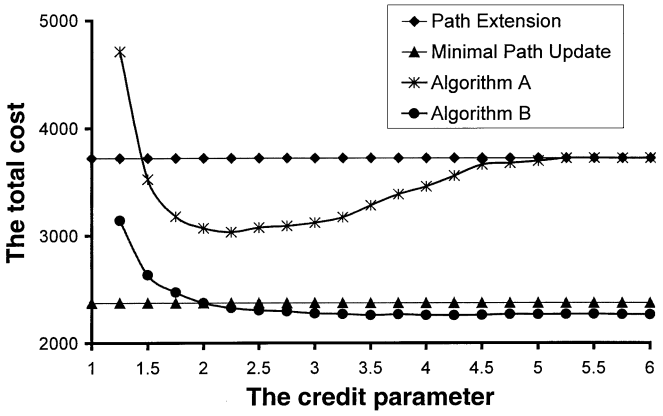

Fig. 12. The effect of the credit parameter, α, on the session total cost with $\beta = 0.99$.

move along a random path with 500 nodes. Its handoff rate is one handoff operation per time unit and the movement range is limited to a square of 100×100 nodes for achieving local movement effect. The session overall cost is calculated by the following equation:

$$\text{Total Cost} = \beta \cdot \text{Setup Cost} + (1-\beta)\cdot \text{Hold Cost}$$

where $0 \le \beta \le 1$ determines the effect of the setup and the hold costs on the session overall cost.

The experimental results show that the connection reestablishment algorithm yields the lowest hold cost with the expense of high setup cost, while the MPU algorithm achieved the lowest setup cost with high hold cost. The anchor rerouting algorithm and our proposed algorithms balance between the setup and the hold cost of the session, where Algorithm $\mathcal{B}$ yields a relatively low setup cost and hold cost. Fig. 9 describes the effect of α on the hold cost and Fig. 10 shows its effect on the setup cost. In the latter figure the results of the connection reestablishment and the anchor rerouting algorithms were omitted due to their high setup cost relative to the other algorithms (110 365, and 19 331, respectively). Figs. 11 and 12 demonstrate the effect of α over the session overall cost for the cases where β equal 0.5 and 0.99, respectively. In the first case, the hold cost is the dominant component of the session total cost. Therefore, $\alpha = 1.25$ yields the lowest total cost for both Algorithms $\mathcal{A}$ and $\mathcal{B}$. Moreover, Algorithm $\mathcal{B}$ produces the minimal session cost for every α.[3] In the second case, the setup and the hold cost have a similar effect over the session total cost. Here, for every $\alpha \geq 2.25$, Algorithm $\mathcal{B}$ produces the minimal cost. We see that the credit parameter, α, can be used for balancing between the setup cost and the hold costs. If the hold cost is high relative to the setup cost, then selecting a low value to α guarantees low overall session cost. If the setup cost is the dominant component, then, a high value to α is preferable.

VI. Summary

This work considers the general connection management problem of reducing the overall session cost of mobile users experiencing handoff. It presents a new model where the overall session cost is composed of two components, a setup cost and hold cost. Every edge e of the network graph is associated with both a setup cost s_e, and a hold cost h_e, and the ratio h_e/s_e is bounded by two positive constants c_1 and c_2, $c_1 \leq h_e/s_e \leq c_2$. It is also assumed that in practical networks there is a correlation between these two components, such that the ratio c_2/c_1 is close to one. Under this assumption, we present two new handoff rerouting algorithms with a competitive ratio of $(2 + (c_2/c_1))$. These algorithms balance between setup and hold costs by using a credit parameter α. Experimental results demonstrate that the proposed algorithms yield low overall cost also in the average sense relative to other algorithms described in the literature. These experiments show that selecting a proper value of α is essential for minimizing the session cost. This value depends on the network parameters as well as on the users movement characteristics. Finding the best α that optimizes the algorithms performance is still an open question.

Appendix
Competitive Ratio of Algorithm $\mathcal{B}$

We now turn to calculate the competitive ratio of Algorithm $\mathcal{B}$. Consider a session σ, defined by a sequence of m triplets, $\sigma = \{(w_k, u_k, \tau_k)\}_{k=0}^{m}$, which is managed by Algorithm $\mathcal{B}$.

[3]The evaluated α is in the range [1, 6].

Lemma 4: For every session σ

$$\text{Hold Cost}_{\mathcal{B}}(\sigma) \leq \frac{c_2}{c_1} \cdot \alpha \cdot \text{Hold Cost}_{\text{OPT}}(\sigma).$$

Proof: For every edge $e \in E$, $c_1 \leq h_e/s_e \leq c_2$. Therefore, for every pair u and v, $s^*(u, v) \leq h^*(u, v)/c_1$. In addition, for every path p, $h(p) \leq c_2 \cdot s(p)$. Step 2 of the algorithm guarantees that the VC setup cost does not exceed α times the setup cost of shortest path between the users. Let p be the VC route and suppose that users are attached to nodes u and v. Hence

$$h(p) \leq c_2 \cdot s(p) \leq c_2 \cdot \alpha \cdot s^*(u, v) \leq \frac{c_2}{c_1} \cdot \alpha \cdot h^*(u, v).$$

The VC hold cost according to OPT is at least $h^*(u, v)$. Therefore the lemma is satisfied. □

We turn to calculate the setup cost. During the session, VC resources of a given edge e may be allocated several times. For distinguishing between the different allocations, each allocation is denoted by a pair (e, k), where e is the edge identifier and k is the corresponding movement index. In the sequel, we distinguish between two types of handoff operations, the MPU *operation* where the allocated path is determined by an MPU decision at Step 1, and the *shortcut operation*, where an exceptional segment is removed from the VC path. The set $\mathcal{F}$ contains the indexes of all the shortcut operations, while the set $\overline{\mathcal{F}}$ contains the indexes of all the MPU operations.

Consider the kth movement from node w_k to node u_k. Let $s(M_k) = s^*(w_k, u_k)$ be the setup cost of the shortest path between these nodes, called the *movement cost*, and let $s(M) = \sum_{k=0}^{m} s(M_k)$. In addition, let U_k be the path between the user location and the COS at the kth handoff operation due to the MPU decision, and let $s(U_k)$ be its cost. Let $s(U) = \sum_{k=0}^{m} s(U_k)$.

Lemma 5: $s(U) \leq s(M)$.

Proof: Consider the kth movement, from node w_k to node u_k, and let x_k be the selected COS at Step 1. Hence, x_k is the closest node to u_k at the existing VC

$$s(U_k) = s(p^*_{u_k, x_k}) \leq s(p^*_{u_k, w_k}) = s(M_k).$$

Thus

$$s(U) = \sum_{i=0}^{m} s(U_i) \leq \sum_{i=0}^{m} s(M_i) = s(M).$$

□

Consider a shortcut operation with index k, in which the path R_k is allocated. Let $\tilde{p}_{k, a_k, b_k}$ be the most expensive exceptional segment that was found at this operation, where the segment end-nodes are a_k and b_k. Note that $\tilde{p}_{k, a_k, b_k}$ contains also part of U_k. Let $L_k = U_k \cap \tilde{p}_{k, a_k, b_k}$, the common resources to both paths, and let $G_k = U_k - L_k$, $G_k = p^*_{u_k, a_k}$ the shortest path between the new user location, u_k, and node a_k. In addition, we denote by $D_k \subseteq \tilde{p}_{k, a_k, b_k}$ all the edge resources that were actually allocated by the MPU operations, and let $C_k \subseteq \tilde{p}_{k, a_k, b_k}$ be all the edge resources that were actually allocated by shortcut operations. Note that L_k, G_k, D_k, and C_k are disjoint sets, and

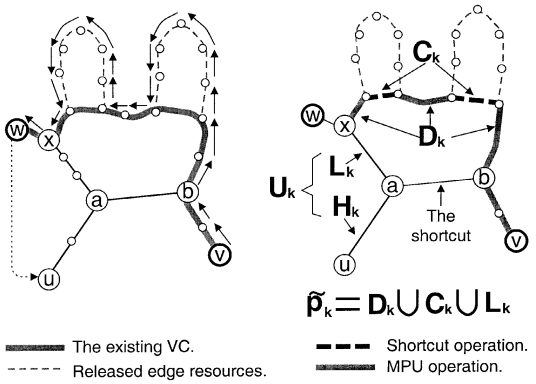

Fig. 13. An example of the notations used for proving the competitive ratio. (a) The VC before the movement. (b) Th notations for the proof.

the path $\tilde{p}_{k,\,a_k,\,b_k} = L_k \cup D_k \cup C_k$. An example of these notations is given in Fig. 13.

For simplicity and without lost of generality, we assume the following shortcut assumption. The resources R_k of every shortcut k may be included in at most a single exceptional segment $\tilde{p}_q$ that is removed by a single shortcut operation,[4] with index q. We say that a shortcut operation k has an *effect of order 1* on a shortcut operation q if its edge resources R_k are included in the exceptional segment $\tilde{p}_q$. A shortcut operation k has an *effect of order* r on a shortcut operation q, if its resources R_k are included in the exceptional segment $\tilde{p}_j$ of a shortcut operation j which has an effect of order $r-1$ on shortcut operation q. For completeness, we say that shortcut operation k has *effect of order 0* on itself. Let $\mathcal{I}_k^r$ be the set of indexes of all the shortcut operations that have effect of order r on shortcut operation k. $\mathcal{I}_k^r$ is defined in a recursive manner

$$\mathcal{I}_k^0 = \{k\}$$
$$\mathcal{I}_k^r = \left\{q \,|\, R_q \subset \tilde{p}_j,\, j \in \mathcal{I}_k^{r-1}\right\}.$$

Lemma 6: For every k, $q \in \mathcal{F}$, $k \neq q$ and positive integer r, the sets $\mathcal{I}_k^r$ and $\mathcal{I}_q^r$ are disjoint.

Proof: We prove by induction on the order of the effect r. For $r = 0$, for every $k \in \mathcal{F}$, $\mathcal{I}_k^0 = \{k\}$ and the lemma is satisfied.

Now, suppose that the lemma is satisfied for every $r' < r$. Consider an index set $\mathcal{I}_k^r$. According to the shortcut assumption, for every shortcut operation $j \in \mathcal{I}_k^r$, its resources R_j are included only in a single exceptional segment $\tilde{p}_z$ of a given shortcut operation z, where $z \in \mathcal{I}_k^{r-1}$. By the induction assumption, there is no another shortcut operation q such that $z \in \mathcal{I}_q^{r-1}$. Therefore, shortcut j is included only in the sets $\mathcal{I}_k^r$, and the lemma is also satisfied for r. □

Lemma 7: $\sum_{k\in\mathcal{F}} [s(U_k) + s(D_k)] \leq s(M)$.

Proof: For every shortcut operation k, D_k are the edge resources allocated by the MPU operations. Moreover, for every pair of q, $k \in \mathcal{F}$, $q \neq k$, the sets D_k and D_q are disjoint. Thus, $\sum_{k\in\mathcal{F}} s(D_k) \leq \sum_{k\in\overline{\mathcal{F}}} s(U_k)$. From Lemma 5

$$\sum_{k\in\mathcal{F}} [s(U_k) + s(D_k)] \leq \sum_{k\in\mathcal{F}} s(U_k) + \sum_{k\in\overline{\mathcal{F}}} s(U_k) = \sum_{k=1}^{m} s(U_k) \leq s(M).$$

□

Lemma 8: For every shortcut operation k and its allocated resources R_k

$$s(R_k) \leq \sum_{r=0}^{\infty} \frac{1}{\alpha^r} \cdot \sum_{j\in\mathcal{I}_k^r} \left[s(G_j) + \frac{1}{\alpha} \cdot (s(L_j) + s(D_j))\right].$$

Proof: For every shortcut operation k and its allocated resources R_k

$$s(R_k) \leq s(G_k) + \frac{1}{\alpha} \cdot s(\tilde{p}_{k,\,a_k,\,b_k}) \leq s(G_k) + \frac{1}{\alpha} \cdot [s(L_k) + s(D_k) + s(C_k)].$$

Since, $s(C_k) = \sum_{j\in\mathcal{I}_k^1} s(R_j)$, the cost of R_k is

$$\begin{aligned}
s(R_k) &\leq s(G_k) + \frac{1}{\alpha} \cdot [s(L_k) + s(D_k) + s(C_k)] \\
&\leq s(G_k) + \frac{1}{\alpha} \cdot [s(L_k) + s(D_k)] + \frac{1}{\alpha} \cdot \sum_{j\in\mathcal{I}_k^1} s(R_j) \\
&\leq s(G_k) + \frac{1}{\alpha} \cdot [s(L_k) + s(D_k)] + \frac{1}{\alpha} \\
&\quad \cdot \sum_{j\in\mathcal{I}_k^1} \left[s(G_j) + \frac{1}{\alpha} \cdot (s(L_j) + s(D_j) + s(C_j))\right] \\
&\leq s(G_k) + \frac{1}{\alpha} \cdot [s(L_k) + s(D_k)] + \frac{1}{\alpha} \\
&\quad \cdot \sum_{j\in\mathcal{I}_k^1} \left[s(G_j) + \frac{1}{\alpha} \cdot (s(L_j) + s(D_j))\right] + \frac{1}{\alpha^2} \\
&\quad \cdot \sum_{j\in\mathcal{I}_k^2} s(R_j) \\
&\leq s(G_k) + \frac{1}{\alpha} \cdot [s(L_k) + s(D_k)] + \frac{1}{\alpha} \\
&\quad \cdot \sum_{j\in\mathcal{I}_k^1} \left[s(G_j) + \frac{1}{\alpha} \cdot (s(L_j) + s(D_j))\right] + \frac{1}{\alpha^2} \\
&\quad \cdot \sum_{j\in\mathcal{I}_k^2} \left[s(G_j) + \frac{1}{\alpha} \cdot (s(L_j) + s(D_j))\right] + \frac{1}{\alpha^2} \\
&\quad \cdot \sum_{j\in\mathcal{I}_k^3} s(R_j) \\
&\leq \sum_{r=0}^{\infty} \frac{1}{\alpha} \sum_{j\in\mathcal{I}_k^r} \left[s(G_j) + \frac{1}{\alpha} \cdot (s(L_j) + s(D_k))\right].
\end{aligned}$$

□

Lemma 9: Let R_k be the allocated resources at the kth handoff operation and let $s(R) = \sum_{k=1}^{m} s(R_k)$, then $s(R) \leq (\alpha/(\alpha-1)) \cdot s(M)$.

[4]In the case where the resources R_k are included in several exceptional segments then R_k is divided to parts such that each part is included in a single exceptional segment. Each part j of R_k is considered to be a separate shortcut with a unique index and it is associated with a proportional part of the segments $\tilde{p}_k$ and U_k.

Proof: According to Lemma 8

$$\sum_{k\in\mathcal{F}} s(R_k) \le \sum_{k\in\mathcal{F}}\sum_{r=0}^{\infty}\frac{1}{\alpha^r}\sum_{j\in\mathcal{I}_k^r}\left[s(G_j)+\frac{1}{\alpha}\cdot(s(L_j)+s(D_j))\right] \le \sum_{r=0}^{\infty}\frac{1}{\alpha^r}\sum_{k\in\mathcal{F}}\sum_{j\in\mathcal{I}_k^r}\left[s(G_j)+\frac{1}{\alpha}\cdot(s(L_j)+s(D_j)\right].$$

According to Lemma 6, for every $k, q \in \mathcal{F}$, $k \neq q$, and positive integer r, the sets $\mathcal{I}_k^r$ and $\mathcal{I}_q^r$ are disjoint. Therefore

$$\sum_{k\in\mathcal{F}}\sum_{j\in\mathcal{I}_k^r}\left[s(G_j)+\frac{1}{\alpha}\cdot(s(L_j)+s(D_j)\right] \le \sum_{k\in\mathcal{F}}\left[s(G_k)+\frac{1}{\alpha}\cdot(s(L_k)+s(D_k)\right].$$

From Lemma 5, $\sum_{k=1}^{m} s(U_k) \le s(M)$, from Lemma 7, $\sum_{k\in\mathcal{F}}[s(U_k)+s(D_k)] \le s(M)$, and at MPU operations, $R_k = U_k$. Hence

$$\begin{aligned} s(R) &= \sum_{k\in\overline{\mathcal{F}}} s(U_k) + \sum_{k\in\mathcal{F}} s(R_k) \\ &\le \sum_{k\in\overline{\mathcal{F}}} s(U_k) + \sum_{r=0}^{\infty}\frac{1}{\alpha^r} \cdot \sum_{k\in\mathcal{F}}\left[s(G_k)+\frac{1}{\alpha}\cdot(s(L_k)+s(D_k))\right] \\ &\le \sum_{k\in\overline{\mathcal{F}}} s(U_k) + \sum_{k\in\mathcal{F}} s(G_k) + \sum_{r=1}^{\infty}\frac{1}{\alpha^r} \cdot \sum_{k\in\mathcal{F}}[s(G_k)+s(L_k)+s(D_k)] \\ &\le \sum_{k=1}^{m} s(U_k) + \left(\sum_{r=1}^{\infty}\frac{1}{\alpha^r}\right)\cdot\left(\sum_{k\in\mathcal{F}}[s(U_k)+s(D_k)]\right) \\ &\le s(M) + \frac{1}{\alpha-1}\cdot s(M) \le \frac{\alpha}{\alpha-1}\cdot s(M). \end{aligned}$$

□

Corollary 5: For every session σ

$$\text{Setup Cost}_{\mathcal{B}}(\sigma) \le \frac{\alpha}{\alpha-1}\cdot s(M).$$

Theorem 8: Algorithm $\mathcal{B}$ is $(2+(c_2/c_1))$-competitive for

$$\alpha = 1 + 2\cdot\frac{c_1}{c_2}.$$

Proof: Similar to the proof of Theorem 4. □

Corollary 6: For general graphs with $c_1 = c_2$, Algorithm $\mathcal{B}$ is 3-competitive for $\alpha = 3$.

REFERENCES

[1] A. Acampora and M. Naghshineh, "An architecture and methodology for mobile executed handoff in cellular ATM networks," *IEEE J. Select. Areas Commun.*, vol. 12, pp. 1365–1375, Oct. 1994.

[2] A. Acharya, S. Biswas, L. French, and D. Raychaudhuri, "Handoff and location management in mobile ATM networks," in *Proc. 3rd Int. Conf. Mobile Multimedia Communication (MoMu-C3)*, Princeton, NJ, Sept. 1996.

[3] P. Agrawal, E. Hyden, P. Krzyzanowski, P. Mishra, M. B. Srivastava, and J. A. Trotter, "SWAN: A mobile multimedia wireless network," *IEEE Pers. Commun.*, vol. 3, pp. 18–33, Apr. 1996.

[4] I. F. Akyildiz, J. S. M. Ho, and M. Ulema, "Performance analysis of the anchor radio system handover method for personal access communications system," in *Proc. IEEE INFOCOM'96*, vol. 3, 1996, pp. 1397–1404.

[5] B. Akyol and D. Cox, "Rerouting for handoff in a wireless ATM network," in *Rec. IEEE Conf. Universal Personal Communication*, vol. 1, Oct. 1996, pp. 374–379.

[6] B. A. J. Banh, G. J. Anido, and E. Dutkiewicz, "Handover re-routing schemes for connection oriented services in mobile ATM networks," in *Proc. IEEE INFOCOM'98*, vol. 3, 1998, pp. 1139–1146.

[7] Y. Bejerano, I. Cidon, and J. Naor, "Dynamic session management: A competitive on-line algorithmic approach," in *Proc. Dial-M for Mobility Workshop*, Boston, MA, Aug. 2000.

[8] A. Borodin and R. El-Yaniv, *Online Computation and Competitive Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[9] R. Ghai and S. Singh, "Maintaining seamless communication between mobile users: An architecture and communication protocol for picocellular networks," *IEEE Pers. Commun.*, vol. 1, pp. 29–36, July 1994.

[10] S. Irani and A. R. Karlin, "On online computation," in *Approximation Algorithms for NP-Hard Problems*, D. S. Hochbaum, Ed. Boston, MA: PWS, 1996, ch. 13.

[11] K. Keeton, B. A. Mah, S. Seshan, R. H. Katz, and D. Ferrari, "Providing connection-oriented network services to mobile hosts," in *Proc. USENIX Symp. Mobile and Location-Independent Computing*, Cambridge, MA, Aug. 1993.

[12] J. Li, R. Yates, and D. Raychaudhuri, "Performance analysis on path rerouting algorithms for handoff control in mobile ATM networks," in *Proc. IEEE INFOCOM'99*, vol. 3, 1999, pp. 1195–1203.

[13] D. E. McDysan and D. L. Spohn, *ATM Theory and Application*. New York: McGraw-Hill, 1994.

[14] J. Mikkonen, C. Corrado, C. Evci, and M. Progler, "Emerging wireless broadband networks," *IEEE Commun. Mag.*, vol. 36, pp. 112–117, Feb. 1998.

[15] M. Mouly and M. B. Pautet, *The GSM System for Mobile Communication*, M. Mouly, Ed. Palaiseau, France: Telecom Pub., 1992.

[16] J. Naylon, D. Gimurray, J. Porter, and A. Hopper, "Low-latency handover in a wireless ATM LAN," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 909–921, Aug. 1998.

[17] T. Ojanpera and R. Prasad, "An overview of third-generation wireless personal communication: A European perspective," *IEEE Pers. Commun.*, vol. 5, pp. 59–65, Dec. 1998.

[18] D. Raychaudhuri, "Wireless ATM: An enabling technology for multimedia personal communication," *Wireless Netw.*, vol. 2, pp. 163–171, 1996.

[19] W. Stallings, *ISDN and Broadband ISDN*, 2nd ed. New York: Macmillan , 1992.

[20] C. K. Toh, "Performance evaluation of crossover switch discovery algorithms for wireless ATM LANs," in *Proc. IEEE INFOCOM'96*, vol. 3, 1996, pp. 1380–1387.

Yigal Bejerano received the B.Sc.degree (*summa cum laude*) in computer engineering, the M.Sc. degree in computer science, and the Ph.D. degree in electrical engineering from the Technion–Israel Institute of Technology, Haifa, Israel, in 1991, 1995, and 2000, respectively.

He is currently a Member of the Technical Staff (MTS) at Bell Laboratories, Lucent Technologies, Murray Hill, NJ. His research interests are mainly management aspects of high-speed and wireless networks, including the areas of mobility management, network monitoring, topology discovery and QoS routing.

Dr. Bejerano is on the technical program committee (TPC) of the INFOCOM–2002 and INFOCOM–2003 conferences.

Israel Cidon received the B.Se. and D.Sc. degrees in electrical engineering from the Technion-Israel Institute of Technology, Haifa, Israel.

He is a Professor in the Faculty of Electrical Engineering at the Technion. During 1994–1995, he was Manager of High-Speed Networking at Sun Microsystems Labs, Mountain View, CA. Between 1985 and 1994, he was with IBM Thomas J. Watson Research Center, White Plains, NY, where he served as a Research Staff Member and a Manager of the Network Architectures and Algorithms Group.

He was an Editor for the IEEE/ACM TRANSACTIONS ON NETWORKING and for the IEEE TRANSACTIONS ON COMMUNICATIONS. Dr. Cidon is a recipient of the IBM Outstanding Innovation Award for his work on the PARIS project and topology update algorithms (1989 and 1993, respectively).

Joseph (Seffi) Naor received the B.Sc. degree in computer science (cum laude) from the Technion–Israel Institute of Technology, Haifa, Israel, in 1981, and the M.Sc. (*cum laude*) and Ph.D. degrees in computer science, from the Hebrew University of Jerusalem, Jerusalem, Israel, in 1983 and 1987, respectively.

He is currently an Associate Professor of Computer Science at the Technion, where he has been on the faculty since 1991. During 1987–1988 he was a Postdoctoral Research Associate at the University of Southern California, Los Angeles, and during 1988–1991 he was a Postdoctoral Research Associate at Stanford University, Stanford, CA. During 1998–2000, he was a Member of the Technical Staff at Bell Laboratories, Lucent Technologies, Murray Hill, NJ. He was a Visiting Scientist at the IBM T. J. Watson Research Center, White Plains, NY, and at IBM, Research Laboratory, Tokyo, Japan. His research interests are mainly in the design and analysis of efficient algorithms, in particular, approximation algorithms for NP-hard algorithms and on-line algorithms. Much of the focus of his research is in the area of communication networks, in scheduling, load balancing, multimedia, quality of service, and high-speed networks. He has more than 50 published papers in top professional journals and conferences.