Online Perfect Matching and Mobile Computing

Edward F. Grove[*], Ming-Yang Kao[**], P. Krishnan[***], and Jeffrey Scott Vitter[†]

Department of Computer Science, Duke University, Durham NC 27708, USA
Email:{efg,kao,pk,jsv}@cs.duke.edu

Abstract. We present a natural online perfect matching problem motivated by problems in mobile computing. A total of n customers connect and disconnect sequentially, and each customer has an associated set of stations to which it may connect. Each station has a capacity limit. We allow the network to preemptively switch a customer between allowed stations to make room for a new arrival. We wish to minimize the total number of switches required to provide service to every customer. Equivalently, we wish to maintain a perfect matching between customers and stations and minimize the lengths of the augmenting paths. We measure performance by the worst case ratio of the number of switches made to the minimum number required.

When each customer can be connected to at most two stations:
- Some intuitive algorithms have lower bounds of $\Omega(n)$ and $\Omega(n/\log n)$.
- When the station capacities are 1, there is an upper bound of $O(\sqrt{n})$.
- When customers do not disconnect and the station capacity is 1, we achieve a competitive ratio of $O(\log n)$.
- There is a lower bound of $\Omega(\sqrt{n})$ when the station capacities are 2.
- We present optimal algorithms when the station capacity is arbitrary in special cases.

1 Introduction

We present an online problem related to the emerging field of *mobile computing* [AwP, DKM, IDJ, USE, Wei]. Current trends suggest that in the near future there will be many customers with portable computing boxes, trying to connect to a huge network of services. The most probable way this will happen will be for the customers to connect to a local "station" through a cellular connection or a wireless LAN, using infrared technology or radio frequency [Dav], and then access the sites of their choice via wired links, e.g., the Internet.

The bottleneck in the mobile interconnection process is the wireless link connecting a customer to a *mobile support station* (MSS). As customers come into the system, the decision of which station they should connect to must be made online. A customer who wants to leave the system *disconnects* from the system. The physical moving of a customer in the world can be modeled by a disconnection at its present site followed by a connection at its new location. As more customers get connected to a station, the response-time performance of the system degrades.

In this paper, we use the standard definition of *competitiveness* to analyze online algorithms. An algorithm A is said to be C-competitive if there exists a constant b such that for every sequence σ of customer connects and disconnects,

$$Cost_A(\sigma) \leq C \cdot Cost_{OPT}(\sigma) + b,$$

where OPT is the optimal offline algorithm, and $Cost_X(\sigma)$ is the expected cost of running X on σ.

Azar et al. [ABK, AKP, ANR] studied the problem of load balancing, motivated by the cellular phone system. They place no limit on the maximum number of customers that can be connected to a station and try to minimize the maximum number of customers connected to any station. In [ANR], the authors assume that customers do not disconnect, and show that the greedy algorithm is strongly competitive with a competitive ratio of $\Theta(\log n)$. In [ABK], customers are allowed to disconnect, and the greedy algorithm is shown to be $\Theta(n^{2/3})$ competitive, with a lowerbound of $\Omega(\sqrt{n})$ on the competitive ratio for the problem. This gap is closed in [AKP] by an algorithm that is $O(\sqrt{n})$ competitive. In [ABK, AKP, ANR], a customer, once connected to a station, cannot be preempted.

Should preemptive scheduling be allowed? It is clear that in some cases it would be advantageous for the system to move a customer from a heavily loaded MSS to another MSS with a lighter load. Preemption adds overhead and makes the system more complicated. We would like to know whether the gains of preemptive scheduling are substantial enough to make it worthwhile. In this paper we fix the maximum number of customers that can be connected to a station. We focus our attention on the problem of online perfect matching, where we maintain a perfect matching of customers to stations, and the cost of connecting a customer is the number of customers that are switched to make room for the new customer (in essence, the length of the augmenting path). In order to get results, we make strong simplifying assumptions. We hope to generate interest

in this version of online matching. Deep results will be required to answer the basic questions of mobile computing that motivate the problems.

Consider the case in which each customer can be connected to at most two stations. When the station capacities are 1, we achieve a competitive ratio of $O(\sqrt{n})$. We show that intuitive algorithms have lower bounds of $\Omega(n)$ and $\Omega(n/\log n)$. If, in addition, we do not allow customers to disconnect, we achieve a competitive ratio of $O(\log n)$. We can derive a lower bound of $\Omega(\sqrt{n})$ when the station capacities are 2. We also present algorithms with optimal competitive factors when station capacity is arbitrary in some special cases.

2 Preemptive Scheduling: Model and Algorithms

Model: Each station has a *capacity*, which is the maximum number of customers that can be connected to the station. Customers arrive and depart sequentially. When a customer enters the system, it announces a set of stations to which it may be connected. While the customer remains in the system, it must be connected to one of those stations, but the system has the power to *switch* the customer from of these stations to another. A station is called *full* if the number of customers connected to it is equal to its capacity. A customer is denied service if and only if the stations to which it may connect remain full no matter how customers presently in the system are switched around. Connected customers cannot be disconnected to make room for new customers. A connection costs 1, a disconnection costs 1, a switch costs 1, and there is no cost for denying service.

Our goal is to develop algorithms which do not need to do too much switching in order to connect the incoming customers.

This paper concentrates on the case when each customer can be connected to at most two stations. For this case, there is a graph-based representation that is simpler than the definition above.

Simplified Model: The stations and customers are represented by a graph. We denote by $G = (S, E)$ the graph on the set of stations S. Let $n = |S|$ be the number of stations. The customers appear on the edges or on the nodes of G. A customer appearing on edge $(v_i, v_j) \in E$ can be connected either to station v_i or to station v_j. A customer appearing at node v_i can be connected only to station v_i. A customer on edge (v_i, v_j) who is connected to station v_i can be *switched* at a cost of 1 to be connected to station v_j.

If the capacity of every station is 1, there can be at most two customers on each edge (v_i, v_j). Let the capacity of each station be 1. An edge (v_i, v_j) is said to *point towards* station v_i if there is one customer on edge (v_i, v_j) and it is connected to station v_i. We say that an edge (v_i, v_j) is *unaligned* for algorithms A and B, if edge (v_i, v_j) points towards station v_i for algorithm A, and towards station v_j for algorithm B. An edge (v_i, v_j) is *aligned* for algorithms A and B, if edge (v_i, v_j) points towards the same station for both A and B. An edge that

is neither aligned nor unaligned is *irrelevant*. A maximal path $(v_{i_1}, v_{i_2}, \ldots, v_{i_k})$ in G is called a directed *chain* if for all $1 \leq j \leq k - 1$, edge $(v_{i_j}, v_{i_{j+1}})$ points towards station $v_{i_{j+1}}$; station v_{i_k} is called the *head* of the chain, and station v_{i_1} is the *tail* of the chain. Two chains are said to be (un)aligned for algorithms A and B if all the edges on the chain are (un)aligned for algorithms A and B. A directed chain $(v_{i_1}, v_{i_2}, \ldots, v_{i_k})$ can be *switched* to get the directed chain $(v_{i_k}, v_{i_{k-1}}, \ldots, v_{i_1})$. A switch that is not required to provide connection to a new customer is called a *useless* switch.

Algorithm GREEDY: Assigns a new customer to a station that minimizes the number of switches, choosing arbitrarily if there is a tie.

Algorithm ASSIGNLEFT: (valid for trees, and circles of stations) Define a uniform preferred direction "left" on the edges of the graph of stations (e.g., anti-clockwise on a circle, and towards the root for a tree). Assign the new customer on edge (v_1, v_2) to the station along the preferred direction (switching existing customers, if necessary) unless that is impossible.

Algorithm RAND: When switches have to be made to connect an incoming customer, switch a chain with a probability inversely proportional to the length of the chain.

3   When Customers Never Disconnect

When the station capacities are all 1, the "pointing" of edges essentially defines a matching of edges to vertices. Also, each edge is in at most one chain. If a new customer appears on edge (v_i, v_j) and stations v_i, v_j are full, there are at most two possible chains (the chains with v_i and v_j as their heads) that can be switched to accommodate the new customer, assuming no useless switches are made. Irrelevant edges are those with 0 or 2 customers.

Theorem 1. *There is a a lower bound of $\Omega(\log n)$ on the competitive ratio of any deterministic algorithm when there are no disconnections. A graph with $O(n \log n)$ edges achieves this lower bound.*

Proof. (Sketch) The lower bound is achieved on a complete graph where each station has a capacity of 1. The lower bound holds when stations have finite capacities greater than 1, since for each station v of capacity $cap(v)$ we can force $cap(v) - 1$ customers to connect to station v, leaving a graph where each station has capacity 1 to play our game.

Given two complete subgraphs on 2^k stations each, we can "combine" these two subgraphs while maintaining the following invariants: On each subgraph with 2^k stations:

1. The adversary forces the online configuration to be a chain.
2. The adversary can have its chain either aligned or unaligned with the online's.

3. The adversary incurs a cost of at most $2^k - 1$, while the online incurs a cost of at least $k2^{k-2}$.

Condition 3 of the invariant implies that a complete graph on n nodes will force a competitive ratio of $\Omega(\log n)$ between the costs of the online algorithm and the adversary. The construction can be improved to use only $O(n \log n)$ edges (details omitted from this abstract). $\qquad\square$

Theorem 2. *When the capacity of every station is 1 and there are no disconnections,* GREEDY *is $O(\log n)$-competitive.*

Proof. (Sketch) Let us define a *component* to be the set of nodes connected by edges on which there are customers. Initially, each node is a component by itself. As connection requests arrive, two things can happen:

1. A component becomes "dead" when a cycle is formed (i.e., a chain $(v_{i_1}, \dots, v_{i_1})$), or when a customer appears at a station (rather than an edge adjacent to the station). Once this happens, the nodes of the component don't affect anything in the future.
2. Two components C_1 and C_2 join. In this case, there are two chains $c_1 \in C_1$ and $c_2 \in C_2$ that can be switched. GREEDY switches the smaller of the two chains. We charge the cost of switching the smaller chain uniformly to the edges of the chain of the smaller component. (This implies that each edge is assigned a cost of at most 1.)

Any edge (v_i, v_j) is charged a total cost of at most $O(\log n)$, since the size of the component to which (v_i, v_j) belongs at least doubles each time it is charged. The adversary incurs $\Omega(n)$ cost for the connections. GREEDY is therefore $O(\log n)$ competitive. $\qquad\square$

The proof of Theorem 2 does not hold when the capacities of the stations are arbitrary since Item 1 above is not true when station capacities are greater than 1. However, we show that ASSIGNLEFT is 2-competitive when the graph is a tree and or a cycle of stations even when the capacities on the nodes are arbitrary.

Theorem 3. *When disconnections are not allowed,* ASSIGNLEFT *is 2-competitive against trees and circles of stations with arbitrary capacities.*

Proof. (Sketch) If a customer is assigned or switched to a station against the preferred direction, it will not be switched again. Assigning or switching to a station against a preferred direction can happen only because the station in the preferred direction is full. Since there are no disconnections, full stations remain full. The algorithm first tries to assign a customer to the preferred direction. It follows that any customer can be switched at most once. Notice that the proof is valid for any graph with exactly one path between any two nodes.

The lower bound of 2 is obtained easily when the capacity on every station is 1 by forcing a chain for the online algorithm and forcing a switch of the chain by placing a connect request at the head of the chain. $\qquad\square$
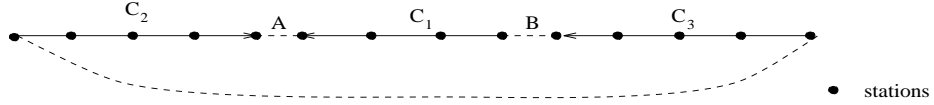
Fig. 1. Proof of Theorem 6. Lower bound of GREEDY.

4 When Customers Disconnect

The lower bounds are achieved by a circle of stations with capacity 1. Theorem 1 implies the following lemma.

Lemma 4. *There is a lower bound of $\Omega(\log n)$ on the competitive ratio of any deterministic algorithm even when there are disconnections.*

A *lazy* algorithm does not perform switches if it can connect the customer without making any switches. For the lower bounds we will discuss in this section, we need to let the adversary set a specific initial configuration for use against lazy online algorithms. The following simple observation says that any desired configuration can be obtained at nominal costs.

Lemma 5. *Given a lazy algorithm, an adversary can achieve any desired legal assignment of customers to stations with $O(n)$ cost, when the capacity of each station is constant.*

We now show that some intuitive algorithms have high lower bounds on the competitive ratio. It is interesting that we do not need complicated graphs to prove these lower bounds. All of the lower bounds can be obtained using circles of stations or path-graphs with capacity 1 on the stations.

Theorem 6. GREEDY *has a lower bound of $\Omega(n)$ on its competitive ratio. This lower bound is achieved on a circle of stations with capacity 1 each.*

Proof. We first force the configuration shown in Figure 1 for both the online and the adversary. Let $|C_1| < |C_2| = |C_3|$. We repeat the following sequence: connect at A, disconnect at A, connect at B, disconnect at B. On the first connect at A, the online switches chain C_1 but the adversary switches chain C_2. On the first request at B, the online switches chain C_1 while the adversary switches chain C_3. In every future request, the online switches chain C_2 back and forth, while the adversary satisfies the requests without any switches. The competitive ratio is asymptotically $|C_1| \approx n/3$. □

Theorem 7. ASSIGNLEFT *has a lower bound of $\Omega(n)$ on its competitive ratio. This lower bound is achieved on a circle of stations with capacity 1 each.*

Notice that Theorem 7 taken in conjunction with Theorem 3 gives a clear indication of the power of allowing disconnections in the model. The ASSIGNLEFT
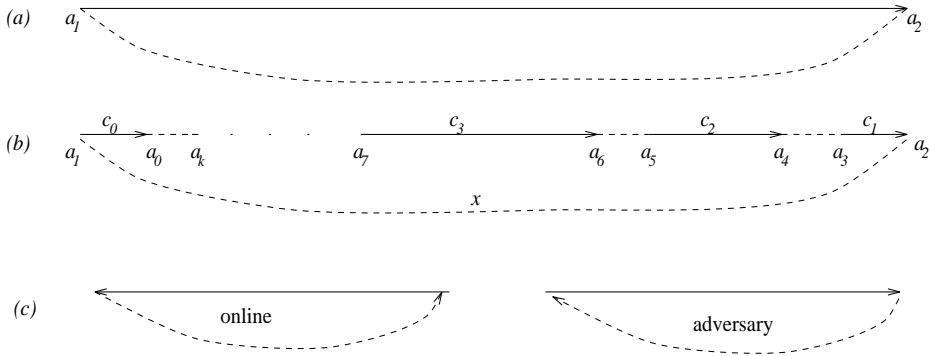
Fig. 2. Proof of Lemma 8. Lower bound of the weighted greedy algorithm.

algorithm, which is 2-competitive without disconnections on a circle of stations, has a lower bound of $\Omega(n)$ on a circle of stations when deletions are allowed.

GREEDY and ASSIGNLEFT have bad competitive ratios because the adversary makes them repeatedly switch the same chain. Does it help to try to avoid this behavior?

Algorithm WEIGHTEDGREEDY**:** Amongst all chains along which to switch customers to accommodate a new request, choose the one along which the total number of switches already made is minimum.

Unfortunately, WEIGHTEDGREEDY also has a high lower bound on its competitive ratio.

Lemma 8. *There is a lower bound of $\Omega(n/\log n)$ on the competitive ratio for* WEIGHTEDGREEDY. *This lower bound is achieved on a circle of stations with capacity 1 each.*

Proof. We first force the configuration shown in Figure 2a for the online and the adversary (without switching any customers). By Lemma 5, this set-up phase costs $O(n)$ for both the online and the adversary. Now repeat the following "routine:" disconnect the customer between a_0 and a_k, a_3 and a_4, a_5 and a_6, ..., where, $|c_1| = |c_0|, |c_2| = 2|c_1| - 1, |c_3| = 2|c_2| - 1, \ldots$. The situation is as shown in Figure 2b. Make a request at a_0. Both the online and the adversary switch chain c_0. Delete the customer at a_0 and make a request at x. The weighted greedy algorithm will switch chain c_1 (since the customers on chain c_0 have been switched more times than the ones on chain c_1), while the adversary switches chain c_0. For every subsequent request at (a_{2i-1}, a_{2i}) the online switches chain c_i, while the offline does $O(1)$ work. (The length of the chains have been adjusted to make this happen.) Now the chains look as in Figure 2c. Disconnect the customer at x and force the online to switch and get aligned on all edges with the offline.

For the requests made in the "routine," the adversary does $O(\log n)$ work while the online does $O(n)$ work. Asymptotically, this implies a lower bound of $\Omega(n/\log n)$ on the competitive ratio. $\qquad\square$
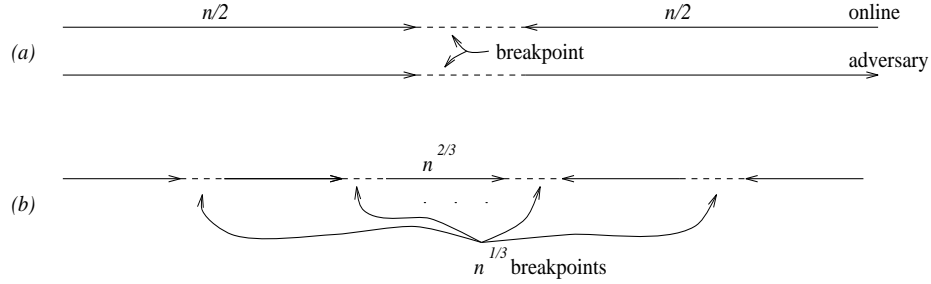
Fig. 3. Proof of Theorem 9. Lower bound of RAND.

We have seen that intuitive deterministic algorithms have a bad competitive ratio. We can do better if we use randomization. We will now analyze RAND.

Theorem 9. *There is a lower bound of $\Omega(\sqrt{n})$ on the competitive ratio of RAND against an adaptive adversary, and a lower bound of $\Omega(n^{1/3})$ on the competitive ratio of RAND against an oblivious adversary. These lower bounds are achieved on a circle of stations of capacity 1.*

Proof. (Sketch) We first get the configuration for the online and the adaptive adversary as in Figure 3a with $\Theta(n)$ cost (see Lemma 5). Delete the customers that are at distance $\sqrt{n}$ to the left and right of the "break point" and make a request at the break point. The adversary incurs a cost of 1 and the online incurs a cost of $\sqrt{n}$ to connect this new customer, and the break point moves. Repeat this process until the break point reaches one of the ends. By the theory of random walks, the break point takes an expected $\Theta(n)$ times before it reaches one of the ends.

The expected cost of the online algorithm is $\Theta(n)$ (for the setup) $+ \Theta(n\sqrt{n})$, while the adversary's cost is $O(n)$ (for the setup) $+ \Theta(n)$, giving a lower bound of $\Omega(\sqrt{n})$ on the competitive ratio.

Against an oblivious adversary, we do not know where the break point is. We divide the chain into $n^{1/3}$ sub-chains of length $n^{2/3}$ each thereby defining $n^{1/3}$ breakpoints as shown in Figure 3b. We place a request at each of these $n^{1/3}$ breakpoints and repeat this $n^{2/3}$ times (the expected time for RAND to align with the adversary). The online does a total of $\Omega(n^{4/3})$ work, while the adversary does $O(n)$ work, giving a lower bound of $\Omega(n^{1/3})$ for the competitive ratio. □

We are now ready to upper bound RAND. The proof uses an interesting potential function derived from the random walk idea used in the lower bound proof of Theorem 9.

Theorem 10. *For any graph with station capacity 1, RAND is $O(\sqrt{n})$-competitive.*

Proof. (Sketch) A customer can appear either on an edge between two stations, or at a station. An adversary can generate a customer connect request at a
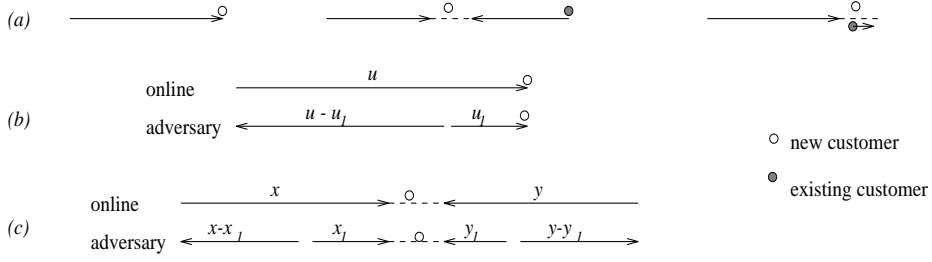
Fig. 4. Proof of competitiveness of RAND. (a) Ways in which forced moves happen. (b) Forced move scenario. (c) Unforced move scenario.

station which forces a lot of switches and then disconnect the customer. However, intuitively, such forced switches make the online paths more aligned with the adversary's. We use a potential function that accurately captures the gain from such switches for the future.

Let D be the number of unaligned edges between RAND and OPT, and let S be the number of aligned edges between RAND and OPT. We define our potential function as

$$\Phi = k_1 D \sqrt{n} + \frac{k_2 D S}{\sqrt{n}}, \tag{1}$$

where k_1 and k_2 are constants to be determined. Intuitively, the first part of the potential function, $\Phi_1 = k_1 D \sqrt{n}$, accounts for the situation when switches are forced. The second part, $\Phi_2 = k_2 D S / \sqrt{n}$ accounts for the cost of doing a random walk before aligning with the adversary. Let W_{on} be the cost of the online algorithm to service a connection or disconnection request, and let W_{adv} be the cost for the adversary. To show an $O(\sqrt{n})$ competitive ratio, we need to show that $\sum W_{on} \leq c\sqrt{n} \sum W_{adv}$. It suffices to show that for each (connection or disconnection) request,

$$W_{on} + \Delta\Phi \leq c\sqrt{n} \times W_{adv}, \tag{2}$$

since summing (2) over all requests will give us our result.

It is easy to see that for a disconnection, the potential always drops (i.e., $\Delta\Phi < 0$) and so (2) is satisfied. The case when a connection can be satisfied without any switches is also easily verified. The hard part is when a connection forces switches. We divide our analysis into two cases.

1. In *forced moves*, the online algorithm does not have a choice of which chain of customers to switch. Forced moves happen because of the situations given in Figure 4a. The forced move scenario is given in Figure 4b. In this case, $W_{on} = u$, and u_1 edges along the chain are aligned between the online algorithm and the adversary, $0 \leq u_1 \leq u$. We have $W_{adv} \geq u_1$. Clearly, $\Delta D = -(u - u_1)$, and $\Delta S = u - u_1$. Hence, $\Delta\Phi = -k_1\sqrt{n}(u - u_1) + k_2(u - u_1)(D - S - u + u_1)/\sqrt{n}$. Since $(D - S - u + u_1) \leq n$, it can be verified that (2) holds as long as $k_1 \geq k_2$.

2. In *unforced moves*, the online algorithm has a choice between two chains of customers to switch. The situation is as depicted in Figure 4c, where $0 \leq y_1 \leq y$, and $0 \leq x_1 \leq x$. In this case, the online algorithm switches the chain of length x with a probability of $y/(x+y)$, and the chain of length y with a probability of $x/(x+y)$ incurring an expected cost of $2xy/(x+y)$. Without loss of generality, assume that the adversary switches the chain of length y_1; hence $W_{adv} = y_1 + 1$. We need to verify that (2) holds.

If the online algorithm switches the chain of length y, $\Delta D = -(y - y_1)$, and $\Delta S = y - y_1 + 1$. If the online algorithm switches the chain of length x, $\Delta D = 2x_1 - x + y_1 + 1$, and $\Delta S = x - 2x_1 - y_1$. Substituting and simplifying, we get

$$\Delta \Phi_1 \approx \frac{2k_1\sqrt{n}y(x_1 - x)}{x + y}; \tag{3}$$

$$\Delta \Phi_2 \approx \frac{k_2 y (x - x_1)(2D + 4x_1)}{\sqrt{n}(x + y)} + \frac{2k_2 y S (x_1 - x)}{(x + y)\sqrt{n}} - \frac{k_2 xy}{\sqrt{n}}. \tag{4}$$

In simplifying to get the above expressions, we ignored terms of value less than or equal to $c_3\sqrt{n}(y_1 + 1)$. For an appropriately large c, terms of value $\leq c_3\sqrt{n}(y_1 + 1)$ appearing on the lhs of (2) will be "paid for" by W_{adv} on the rhs of (2). The constant c_3 is independent of c. Note that $\Delta \Phi_1 \leq 0$. Since $2D + 4x_1 = O(n)$ and $x \geq 0$, the first term of $\Delta \Phi_2$ is non-negative but is "paid for" by $\Delta \Phi_1$, if $k_1 \geq 3k_2$. The second term of $\Delta \Phi_2$ is negative and can be ignored. The third term of $\Delta \Phi_2$ (i.e., $-k_2 xy/\sqrt{n}$) "pays" for the online cost of $2xy/(x + y)$, if $k_2 > 2$ and $x + y \geq \sqrt{n}$. If $x + y < \sqrt{n}$, the adversary's cost pays for the online cost.

□

Theorem 11. *There is a lower bound of $\Omega(\sqrt{n})$ on the competitive ratio of any algorithm (if randomized, against an adaptive adversary) when the capacities of the stations are 2. This lower bound is achieved by a tree of stations.*

Proof. (Sketch) Consider the tree of stations in Figure 5. The capacity on each station is 2. The idea of the proof is to force an initial configuration with all chains pointing towards the root. The adversary then places a connect request at the root, and the online switches a chain C_i with some leaf l_i as the tail of the chain. It then disconnects the new customer and forces the online to switch the chain C_i again by placing a request at l_i. Divide the request sequence into minimal blocks, such that for every block the online switches all the $\sqrt{n}$ chains. For the block B_j of requests, let $C_{j,1}$ be the first chain switched by the online algorithm and let $C_{j,m}$ be the last chain switched by the online algorithm, $m \geq \sqrt{n}$. For the first request in block B_j, the adversary switches chain $C_{j,m}$ incurring a cost of $O(\sqrt{n})$. For every connect request except the first in the block, the online incurs a cost of $\Omega(\sqrt{n})$, while the adversary incurs a cost of 1. The net ratio of the cost of the online to the adversary for any block of connect requests is $\Omega(\sqrt{n})$. □
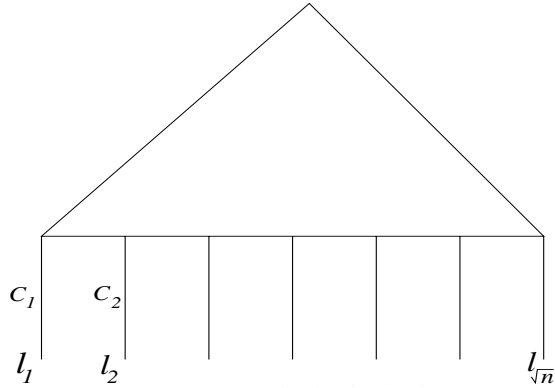
Fig. 5. Lower bound of $\Omega(\sqrt{n})$ for a tree.

Theorem 12. RAND *is $O(\sqrt{n})$-competitive on a circle of stations with arbitrary capacities.*

Like the proof of Theorem 10, this proof is based on a potential function that measures a random walk, but it is of a rather different form. For brevity, the proof is omitted from this abstract.

5 Conclusions

We have presented in this paper a model of mobile connectivity. There are some very challenging questions that arise from our model. The most obvious open question is: Is there an $O(\sqrt{n})$-competitive algorithm for general graphs when disconnections are allowed, and the capacities are arbitrary? We have a generalization of RAND that we believe is $O(\sqrt{n})$-competitive. When the capacities are arbitrary, the main difficulty is that there can be many non-disjoint augmenting paths for a new connect request. Our generalization of RAND defines a resistive network [CDR] of the augmenting paths and switches a path proportional to the current that would flow through it when a unit voltage is placed at the new connect request point. We expect that the online algorithms discussed in this paper will do better when the capacities of the stations are larger, and the degree of each customer is greater than 2, because intuitively, more paths help the algorithm more than adversary.

References

[AwP] B. Awerbuch and D. Peleg, "Concurrent Online Tracking of Mobile Users," *Proceedings of SIGCOMM 1991*.

[ABK] Y. Azar, A. Y. Broder, and A. R. Karlin, "On-line Load Balancing," *Proceedings of the 33rd Symposium on Foundations of Computer Science* (October 1993), 218–225.

[AKP] Y. Azar, B. Kalyanasundaram, S. Plotkin, K. R. Pruhs, and O. Waarts, "Online Load Balancing of Temporary Tasks," *Proceedings of the 1993 Workshop on Algorithms and Data Structures* (August 1993).

[ANR] Y. Azar, J. Naor, and R. Rom, "The Competitiveness of On-Line Assignments," *Proceedings of the 3rd Annual ACM-SIAM Symposium on Discrete Algorithms* (January 1992).

[CDR] D. Coppersmith, P. Doyle, P. Raghavan, M. Snir, "Random Walks on Weighted Graphs and Applications to On-line Algorithms," *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing* (May 1990, and 369–378), Also appears as IBM Research Report RC 15840.

[Dav] D. W. Davis, "Wirless LANs broadcast their benefits over cable," Electronic Business, May 1991.

[DKM] F. Douglis, P. Krishnan, and B. Marsh, "Thwarting the Power Hungry Disk," *Proceedings of the 1994 Winter USENIX Conference* (January 1994).

[IDJ] J. Ioannidis, D. Duchamp, and G. Macquire, Jr., "IP-based Protocols for Mobile Internetworking," *Proceedings of SIGCOMM '91* (September 1991), 235–245.

[USE] USENIX, "Proceedings of the USENIX Mobile and Location-Independent Computing Symposium," Cambridge, MA, August 1993.

[Wei] M. Weiser, "The Computer for the 21st Century," *Scientific American* (September 1991), 94–104.