LRU Is Better than FIFO

M. Chrobak and J. Noga

Abstract. In the paging problem we have to manage a two-level memory system, in which the first level has short access time but can hold only up to \( k \) pages, while the second level is very large but slow. We use competitive analysis to study the relative performance of the two best known algorithms for paging, LRU and FIFO. Sleator and Tarjan proved that the competitive ratio of LRU and FIFO is \( k \). In practice, however, LRU is known to perform much better than FIFO. It is believed that the superiority of LRU can be attributed to locality of reference exhibited in request sequences. In order to study this phenomenon, Borodin et al. [2] refined the competitive approach by introducing the concept of access graphs. They conjectured that the competitive ratio of LRU on each access graph is less than or equal to the competitive ratio of FIFO. We prove this conjecture in this paper.

Key Words. Paging, Caching, On-line algorithms, Competitive analysis.

1. Introduction. In the paging problem we have a two-level memory system, in which the first level is small and has short access time, while the second level is very large but slow. We refer to the first memory level as the cache, and we denote its size by \( k \). A request to a page \( p \) can result either in a hit, when \( p \) is in the cache, or a fault, when it is not. In response to a fault we need to bring \( p \) into the cache. If the cache is already full, some other page \( q \) currently in the cache needs to be evicted. The goal of a paging algorithm is to choose the evicted page for each fault so as to minimize the cost, defined as the number of faults.

The eviction decisions are made on-line, without the knowledge of future requests. It is not difficult to see that no such on-line algorithm can achieve optimal cost on all request sequences. The competitive ratio is a performance measure designed to quantify the ability of an on-line algorithm to minimize faults, and is defined as follows: Let \( \mathcal{A} \) be an on-line algorithm for paging. We say that \( \mathcal{A} \) is \( C \)-competitive if, for any request sequence \( \varrho \),

\[
\text{cost}_{\mathcal{A}}(\varrho) \leq C \cdot \text{opt}(\varrho) + b, \tag{1}
\]

where \( \text{cost}_{\mathcal{A}}(\varrho) \) is the cost of \( \mathcal{A} \) on \( \varrho \), \( \text{opt}(\varrho) \) is the optimal cost of serving \( \varrho \), and \( b \) is a constant independent of \( \varrho \). The competitive ratio of \( \mathcal{A} \) is the smallest \( C \) for which \( \mathcal{A} \) is \( C \)-competitive. An algorithm \( \mathcal{A} \) is called strongly competitive if its competitive ratio is within a constant factor of the best possible competitive ratio.

\[\text{1}\] Research by both authors was supported by NSF Grant CCR-9503498.

\[\text{2}\] Department of Computer Science, University of California, Riverside, CA 92521, USA. marek@cs.ucr.edu.

\[\text{3}\] Department of Mathematics, University of California, Riverside, CA 92521, USA. jnoga@cs.ucr.edu.

Received June 2, 1997; revised January 28, 1998. Communicated by A. Borodin.
Paging has been extensively studied in the literature on competitive on-line algorithms. Belady [1] gave an optimal off-line algorithm defined by

**BEL:** When a fault occurs, evict the page whose next request is furthest in the future.

The two best known on-line algorithms for paging are **Least-Recently-Used (LRU)** and **First-In-First-Out (FIFO)**, defined as follows:

**LRU:** When a fault occurs, evict the page that was accessed least recently.

**FIFO:** When a fault occurs, evict the page that was brought into the cache least recently.

Sleator and Tarjan [9] proved that the competitive ratio of LRU and FIFO is \( k \).

The competitive approach has been criticized by practitioners for being overly pessimistic. For example, the experiments reported in [10] indicate that the ratio of LRU’s cost to the optimal cost on some typical program traces is nearly independent of the cache size. Another problem with competitive analysis is that algorithms with different performance in practice may have the same competitive ratio. The best examples of this are LRU and FIFO. In practice, LRU performs much better than FIFO.

One reason for this discrepancy may be the fact that typical request sequences exhibit **locality of reference**: whenever a page \( p \) is requested, the next request is most likely to be chosen from a small set of pages “local” to \( p \) (see [1], [3], [7], and [8]). It appears that LRU takes advantage of the locality more effectively than FIFO, but competitive analysis does not reflect this superiority of LRU.

These concerns were addressed by Borodin et al. [2] who model the locality of reference by an **access graph** which is known by the on-line algorithm. The vertices of an access graph \( G \) are pages, and \( G \) has an edge from \( p \) to \( q \) if \( q \) can be requested immediately after \( p \). As in [2] and [5], we assume that \( G \) is undirected (or, equivalently, that the locality relation is symmetric). We define an algorithm \( A \) to be \( C \)-competitive on \( G \) if inequality (1) holds for each \( \varrho \) that is a walk in \( G \). The competitive ratio of \( A \) on \( G \), denoted by \( C_{A,G} \), is the smallest \( C \) for which \( A \) is \( C \)-competitive on \( G \). Under this model, LRU is not an optimally competitive algorithm for most graphs. Although it is computationally difficult, an optimally competitive algorithm can be found for any particular finite graph. Both [2] and [6] investigate the problem of finding a natural algorithm which has near optimal competitive ratio. Fiat and Karlin [4] give strongly competitive algorithms for both the deterministic and randomized cases. Since the access graph may not be known by the on-line algorithm or may require a prohibitive amount of memory to store, Fiat and Mendel [5] define an algorithm as **truly on-line** if its decisions are based solely on the request sequence. A strongly competitive algorithm for the deterministic, truly on-line case is given in [5]. Note that both LRU and FIFO are truly on-line paging algorithms.

One of the problems addressed in [2] is the relative performance of LRU and FIFO. They provide some estimates on the competitive ratios of LRU and FIFO, and they prove that, for each access graph \( G \), \( C_{\text{LRU},G} \leq 2C_{\text{FIFO},G} \). They also conjecture that the constant 2 can be removed. (This conjecture was also stated in [6].) In other words, the conjecture states that LRU is never worse than FIFO.

In this paper we prove that, for all \( G \), \( C_{\text{LRU},G} \leq C_{\text{FIFO},G} \), thus settling the conjecture