Relative Interval Analysis of Paging Algorithms on Access Graphs *

Joan Boyar, Sushmita Gupta, and Kim S. Larsen

University of Southern Denmark, Odense, Denmark
{joan,sgupta,kslarsen}@imada.sdu.dk

Abstract. Access graphs, which have been used previously in connection with competitive analysis and relative worst order analysis to model locality of reference in paging, are considered in connection with relative interval analysis. The algorithms LRU, FIFO, FWF, and FAR are compared using the path, star, and cycle access graphs. In this model, some of the expected results are obtained. However, although LRU is found to be strictly better than FIFO on paths, it has worse performance on stars, cycles, and complete graphs, in this model. We solve an open question from [Dorrigiv, López-Ortiz, Munro, 2009], obtaining tight bounds on the relationship between LRU and FIFO with relative interval analysis.

1 Introduction

The paging problem is the problem of maintaining a subset of a potentially very large set of pages from memory in a significantly smaller cache. When a page is requested, it may already be in cache (called a "hit"), or it must be brought into cache (called a "fault"). The algorithmic problem is the one of choosing an eviction strategy, i.e., which page to evict from cache in the case of a fault, with the objective of minimizing the total number of faults.

Many different paging algorithms have been considered in the literature, many of which can be found in [3, 13]. Among the best known are LRU (least-recently-used), which always evicts the least recently used page, and FIFO (first-in-first-out), which evicts pages in the order they entered the cache. We also consider a known bad algorithm, FWF (flush-when-full), which is often used for reference, since quality measures ought to be able to determine at the very least that it is worse than the other algorithms. If FWF encounters a fault with a full cache, it empties its cache, and brings the new page in. Finally, we consider a more involved algorithm, FAR, which works with respect to a known access graph. Whenever a page is requested, it is marked. When it is necessary to evict a page, it always evicts an unmarked page. If all pages are marked in such a situation, FAR first unmarks all pages. The unmarked page it chooses to evict is

* Supported in part by the Danish Council for Independent Research. Part of this work was carried out while the first and third authors were visiting the University of Waterloo, Ontario, Canada.

the one farthest from any marked page in the access graph. For breaking possible ties, we assume the LRU strategy in this paper.

Understanding differences in paging algorithms' behavior under various circumstances has been a topic for much research. The most standard measure of quality of an online algorithm, competitive analysis [18, 15], cannot directly distinguish between most of them. It deems LRU, FIFO, and FWF equivalent, with a competitive ratio of k, where k denotes the size of the cache. Other measures, such as relative worst order analysis [5, 6], can be used to obtain more separations, including that LRU and FIFO are better than FWF and that look-ahead helps. No techniques have been able to separate LRU and FIFO, without adding some modelling of locality of reference.

Although LRU performs better than FIFO in some practical situations [19], if one considers all sequences of length n for any n, bijective/average analysis shows that their average number of faults on these sequences is identical [2], which basically follows from LRU and FIFO being demand paging algorithms. Thus, it is not surprising that some assumptions involving locality of reference are necessary to separate them.

A separation between FIFO and LRU was established quite early using access graphs for modelling locality of reference [10], showing that under competitive analysis, no matter which access graph one restricts to, LRU always does at least as well as FIFO. This proved a conjecture in [4], where the access graph model was introduced. Another way to restrict the input sequences was investigated in [1]. Using Denning's working set model [11, 12] as an inspiration, sequences were limited with regards to the number of distinct pages in a sliding window of size k. This also favors LRU, as does bijective analysis [2], using the same locality of reference definition as [1]. There has also been work in the direction of probabilistic models, including the diffuse adversary model [17] and Markov chain based models [16].

The earlier successes and the generality of access graphs, together with the possibilities the model offers with regards to investigating specific access patterns, makes it an interesting object for further studies. In the light of the recent focus on development of new performance measures, together with the comparative studies initiated in [9], exploring access graphs results in the context of new performance measures seems like a promising direction for expanding our understanding of performance measures as well as concrete algorithms.

One step in that direction was carried out in [7], where more nuanced results were demonstrated, showing that restricting input sequences using the access graph model, while applying relative worst order analysis, LRU is strictly better than FIFO on paths and cycles. The question as to whether or not LRU is at least as good as FIFO on all finite graphs was left as an open problem, but it was shown that there exists a family of graphs which grows with the length of the corresponding request sequence, where LRU and FIFO are incomparable. Since LRU is optimal on paths, it is not surprising that both competitive analysis and relative worst order analysis find that LRU is better than FIFO on paths. Any "reasonable" analysis technique should give this result. Under competitive

analysis, LRU and FIFO are equivalent on cycles. The separation by relative worst order analysis occurs because cycles contain paths, LRU is better on paths, and relative worst order analysis can reflect this. The fact that there exists an infinite family of graphs which grows with the length of the sequence where LRU and FIFO are incomparable may or may not be interesting. There are many sequences were FIFO is better than LRU; they just seem to occur less often in real applications.

Comparing two algorithms under almost any analysis technique is generally equivalent to considering them with the complete graph as an access graph, since the complete graph does not restrict the request sequence in any way. Thus, LRU and FIFO are equivalent on complete graphs under both competitive analysis and relative worst order analysis, since they are equivalent without considering access graphs.

In this paper, we consider relative interval analysis [14]. In some ways relative interval analysis is between competitive analysis and relative worst order analysis. As with relative worst order analysis, two algorithms are compared directly to each other, rather than compared to OPT. This gives the advantage that, when one algorithm dominates another in the sense that it is at least as good as the other on every request sequence and better on some, the analysis will reflect this. However, it is similar to competitive analysis in that the two algorithms are always compared on exactly the same sequence. To compare two algorithms, LRU and FIFO for example, one considers the difference between LRU's and FIFO's performance on any sequence, divided by the length of that sequence. The range that these ratios can take is the "interval" for that pair of algorithms. For FIFO and LRU, [14] found two families of sequences I_n and J_n such that $\lim_{n\to\infty} \frac{\mathrm{FIFO}(I_n)-\mathrm{LRU}(I_n)}{n} = -1 + \frac{1}{k}$ and $\lim_{n\to\infty} \frac{\mathrm{FIFO}(J_n)-\mathrm{LRU}(J_n)}{n} = \frac{1}{2} - \frac{1}{4k-2}$. They left it as an open problem to determine if worse sequences exist, making the interval even larger. In their notation, they proved: $[-1+\frac{1}{k}, \frac{1}{2} - \frac{1}{4k-2}] \subseteq \mathcal{I}(\mathrm{FIFO}, \mathrm{LRU})$. We start by proving that this is tight: $\mathcal{I}(\mathrm{FIFO}, \mathrm{LRU}) = [-1+\frac{1}{k}, \frac{1}{2} - \frac{1}{4k-2}]$. These results would be interpreted as saying that FIFO has *better performance* than LRU, since the absolute value of the minimum value in the interval is larger than the maximum, but also that they have different strengths, since zero is contained in the interior of the interval. We obtain more nuanced results by considering various types of access graphs: complete graphs (K_N), paths (P_N), stars (S_N), and cycles (C_N). This splits the interval of $[-1+\frac{1}{k}, \frac{1}{2} - \frac{1}{4k-2}]$ into subintervals for the respective graph classes. Table 1 shows our results.

Comparing these results with the results from competitive analysis and relative worst order analysis, both with respect to access graphs, it becomes clear that different measures highlight different aspects of the algorithms. Both measures show that LRU is strictly better than FIFO on paths, which is not surprising since it is in fact optimal on paths and FIFO is not. On the other access graphs considered here, relative interval analysis gives results which can be interpreted as incomparability, but leaning towards deeming FIFO the better algorithm. Relative worst order analysis, on the other hand, shows that on cycles,

Table 1. Summary of Results: $\mathcal{A} \in \{\text{FAR}, \text{LRU}\}$, $\mathcal{B} \in \{\text{FAR}, \text{FIFO}, \text{LRU}\}$, $N = k + r$, with $1 \leq r \leq k-1$, $X_r = r(x-1) + \lceil \frac{N}{2^x} \rceil$ with $x = \lfloor \log \frac{N}{r} \rfloor$, and $\tilde{N}$ denotes N if N is even, and $N-1$ otherwise.

Lower Bound	Relative Interval	Upper Bound	Theorem
	$\mathcal{I}^{K_N}[\text{FIFO}, \text{LRU}] =$	$\left[-1+\frac{1}{k},\ \frac{1}{2}-\frac{1}{4k-2}\right]$	1
	$\mathcal{I}^{K_N}[\text{FWF}, \mathcal{A}] =$	$\left[0,\ 1-\frac{1}{k}\right]$	2
$\left[0,\ 1-\frac{k+1}{k^2}\right]$	$\subseteq \mathcal{I}^{K_N}[\text{FWF}, \text{FIFO}] \subseteq$	$\left[0,\ 1-\frac{1}{k}\right]$	3
	$\mathcal{I}^{P_N}[\text{FIFO}, \mathcal{A}] =$	$\left[0,\ \frac{1}{2}-\frac{1}{2k}\right]$	4
	$\mathcal{I}^{P_N}[\text{FWF}, \mathcal{A}] =$	$\left[0,\ 1-\frac{1}{k}\right]$	2
$\left[0,\ 1-\frac{k+1}{k^2}\right]$	$\subseteq \mathcal{I}^{P_N}[\text{FWF}, \text{FIFO}] \subseteq$	$\left[0,\ 1-\frac{1}{k}\right]$	3
$\left[-\frac{1}{2}+\Theta(\frac{1}{k}),\ \frac{1}{4}+\Theta(\frac{1}{k})\right]$	$\subseteq \mathcal{I}^{S_N}[\text{FIFO}, \mathcal{A}] \subseteq$	$\left[-\frac{1}{2}+\Theta(\frac{1}{k}),\ \frac{1}{4}+\Theta(\frac{1}{k})\right]$	5
	$\mathcal{I}^{S_N}[\text{FWF}, \mathcal{B}] =$	$\left[0,\ \frac{1}{2}\right]$	6
$\left[-1+\frac{r}{k},\ \frac{1}{2}-\frac{1}{4k-2}\right]$	$\subseteq \mathcal{I}^{C_N}[\text{FIFO}, \text{LRU}] \subseteq$	$\left[-1+\frac{1}{k},\ \frac{1}{2}-\frac{1}{4k-2}\right]$	7
	$\mathcal{I}^{C_N}[\text{FWF}, \text{LRU}] =$	$\left[0,\ 1-\frac{1}{k}\right]$	8
$\left[-\frac{r\left(\lfloor \log \frac{\tilde{N}}{r} \rfloor - 1\right)}{N-1}, 1-\frac{X_r}{k}\right]$	$\subseteq \mathcal{I}^{C_N}[\text{LRU}, \text{FAR}] \subseteq$	$\left[-\frac{X_r-1}{k},\ 1-\frac{1}{k}\right]$	9
$\left[-\frac{X_r-r}{k},\ 1-\frac{X_r}{k}\right]$	$\subseteq \mathcal{I}^{C_N}[\text{FIFO}, \text{FAR}] \subseteq$	$\left[-\frac{X_r-1}{k},\ 1-\frac{1}{k}\right]$	9
$\left[0,\ 1-\frac{X_r}{k}\right]$	$\subseteq \mathcal{I}^{C_N}[\text{FWF}, \text{FAR}] \subseteq$	$\left[0,\ 1-\frac{1}{k}\right]$	9
$\left[0,\ 1-\frac{k+1}{k^2}\right]$	$\subseteq \mathcal{I}^{C_N}[\text{FWF}, \text{FIFO}] \subseteq$	$\left[0,\ 1-\frac{1}{k}\right]$	3

LRU is strictly better than FIFO, and on complete graphs, they are equivalent. It has not yet been studied on stars, but an incomparability result for LRU and FIFO has been found for a family of graphs growing with the length of the input.

2 Preliminaries

We have defined the paging algorithms in the introduction. If more detail is desired, the algorithms are described in [3].

An *access graph* for paging models the access patterns, i.e., which pages can be requested after a given page. Thus, the vertices are pages, and after a page p has been requested, the next request is to p or one of its neighbors in the access graph. We let N denote the number of vertices of the access graph under consideration at a given time. This is the same as the number of different pages we consider. We will always assume that $N > k$, since otherwise the problem is trivial, and let $r = N - k$. A requests sequence is a sequence of pages and the sequence *respects* a given access graph if any two consecutive requests are either identical or neighbors in the access graph. We let $\mathcal{L}(G)$ denote the set of all request sequences respecting G.

We use the definition of k-phases from [3]:

Definition 1. *A request sequence can be divided recursively into a number of* k-phases *as follows: Phase* 0 *is the empty sequence. For every* $i \geq 1$*, Phase* i *is a maximal sequence following Phase* $i-1$ *containing at most* k *distinct requests.*

Thus, Phase i begins on the $(k+1)$st distinct page requested since the start of Phase $i-1$, and the last phase may contain fewer than k different pages. We generally want to ignore Phase 0, and refer to Phase 1 as the first phase.

Similarly, we can define x-blocks, for some integer x, focusing on when a given algorithm $\mathcal{A}$ has faulted x times.

Definition 2. *A request sequence can be divided recursively into a number of* x-blocks *with respect to an algorithm* $\mathcal{A}$ *as follows: The* 0*th* x*-block is the empty sequence. For every* $i \geq 1$*, the* i*th* x*-block is a maximal sequence following the* $(i-1)$*st* x*-block for which* $\mathcal{A}$ *faults at most* x *times.*

The complete *blocks are defined to be the ones with* x *faults, i.e., excluding the* 0*th block and possibly the last.*

There are some well-known and important classifications of paging algorithms, which are used here and in most other papers on paging [3]: An paging algorithm is called *conservative* if it incurs at most k page faults on any consecutive subsequence of the input containing k or fewer distinct page references. LRU and FIFO belong to this class. Similarly, a paging algorithm is called a *marking* algorithm if for any k-phase, once a page has been requested in that phase, it is not evicted for the duration of that phase. LRU, FAR, and FWF are marking algorithms.

If $\mathcal{A}$ is a paging algorithm, we let $\mathcal{A}(I)$ denote $\mathcal{A}$'s cost (number of faults) on the input (request) sequence I. We adapt relative interval analysis from [14] to access graphs. Let $\mathcal{A}$ and $\mathcal{B}$ be two algorithms. We define the following notation:

$$\mathrm{Min}^G(\mathcal{A},\mathcal{B}) = \lim_{n\to\infty} \inf \frac{\min_{|I|=n, I\in\mathcal{L}(G)}\{\mathcal{A}(I)-\mathcal{B}(I)\}}{n} \text{ and}$$

$$\mathrm{Max}^G(\mathcal{A},\mathcal{B}) = \lim_{n\to\infty} \sup \frac{\max_{|I|=n, I\in\mathcal{L}(G)}\{\mathcal{A}(I)-\mathcal{B}(I)\}}{n}$$

Definition 3. *The relative interval of two algorithms* $\mathcal{A}$ *and* $\mathcal{B}$ *with respect to the access graph,* G*, is*

$$\mathcal{I}^G(\mathcal{A},\mathcal{B}) = [\mathrm{Min}^G(\mathcal{A},\mathcal{B}), \mathrm{Max}^G(\mathcal{A},\mathcal{B})]$$

$\mathcal{B}$ has better performance than $\mathcal{A}$ *if* $\mathrm{Max}^G(\mathcal{A},\mathcal{B}) > |\mathrm{Min}^G(\mathcal{A},\mathcal{B})|$. $\mathcal{B}$ dominates $\mathcal{A}$ *if* $\mathcal{I}^G(\mathcal{A},\mathcal{B}) = [0,\beta]$ *for some* $\beta > 0$*. Note that* $\mathrm{Max}^G(\mathcal{A},\mathcal{B}) = -\mathrm{Min}^G(\mathcal{B},\mathcal{A})$.

This definition generalizes the one from [14] in that the original definition is the special case where G is the complete graph, which is the same as saying that there are no restrictions on the sequences.

Note that if $\mathcal{B}$ *dominates* $\mathcal{A}$, this means that $\mathcal{A}$ does not outperform $\mathcal{B}$ on any sequence (asymptotically), while there are sequences on which $\mathcal{B}$ outperforms $\mathcal{A}$. Also, when $\mathrm{Max}^G(\mathcal{A},\mathcal{B})$ is close to 0, this indicates that $\mathcal{A}$'s performance is not much worse than that of $\mathcal{B}$'s.

Due to space limitations, most proofs and the statements of most lemmas have been omitted. Refer to [8] for all the details.

3 Complete Graphs

As remarked earlier, if the access graph is complete, it incurs no restrictions, so the result of this section is in the same model as [14]. In [14], it is shown that $[-\frac{k-1}{k}, \frac{k-1}{2k-1}] \subseteq \mathcal{I}(\text{FIFO}, \text{LRU})$. Below, we answer an open question from [14], proving that this is tight. The full version of the paper [8] contains a more detailed proof.

Lemma 1. *For any access graph G,*

$$-1 + \frac{1}{k} \leq \text{Min}^G(\text{FIFO}, \text{LRU}) \text{ and } \text{Max}^G(\text{FIFO}, \text{LRU}) \leq \frac{1}{2} - \frac{1}{4k-2}.$$

Proof. We first consider the Min value. Suppose that a sequence I has b complete k-phases. Since LRU is conservative and a complete k-phase contains k distinct pages, it cannot fault more than $bk + k - 1$ times [3]. With b complete k-phases, $\text{FIFO}(I) \geq k + b - 1$, so $\text{FIFO}(I) - \text{LRU}(I) \geq k + b - 1 - (bk + k - 1) = -b(k-1)$. Each k-phase must have length at least k, so $|I| \geq bk$. Thus, $\text{Min}^G(\text{FIFO}, \text{LRU}) \geq -\frac{b(k-1)}{bk} = -1 + \frac{1}{k}$.

We now consider the Max value. Given a request sequence I, we let B_i denote the ith k-block for FIFO. Assume that there are b complete k-blocks. FIFO faults k times per complete k-block and up to $k-1$ times for the possible final k-block. Thus, $\text{FIFO}(I) \leq bk + (k-1)$. Assume that LRU faults α_i times in B_i. With b complete k-blocks, which are at least as long as k-phases, LRU faults at least $b + k - 1$ times. Thus, $\Sigma_{i=1}^{b} \alpha_i \geq b + k - 1$.

We now compute a lower bound on the length of the request sequence I based on the number of complete k-blocks in it and the algorithms' behavior on it.

As a first step, with every request on which FIFO faults and LRU has a hit, we associate a distinct request where FIFO has a hit. Let r be such a request to a page p in B_i. Since it is a hit for LRU, p must have been requested in the maximal subsequence of requests I' consisting of k distinct pages and ending just before r. Consider the first such request, r', in I'. If it were a fault for FIFO, FIFO could not have faulted again on r. Thus, r' was a hit for FIFO and we associate r' with r.

To establish that the association is distinct, assume that r' also gets associated with a request r''. Without loss of generality, assume that r'' is later than r. For FIFO to fault on both r and r'', there must be at least k distinct pages different from p in between r and r''. However, since we are assuming that LRU has a hit on r'', by the property of LRU, the page requested by r'' must have been requested during the same k distinct pages. Thus, by the construction above, the page that gets associated with r'' (and r) will be later than r, which is a contradiction.

Thus, if LRU faults α_i times in B_i, by the procedure above, we identify at least $k - \alpha_i$ distinct requests. In total, there are at least $\Sigma_{i=1}^{b}(k - \alpha_i) = bk - \Sigma_{i=1}^{b}\alpha_i$ distinct hits for FIFO in I and, since there are b complete k-blocks,

at least bk faults. Thus, the length of I is at least $2bk - \Sigma_{i=1}^{b}\alpha_i$, and

$$\frac{\mathrm{FIFO}(I) - \mathrm{LRU}(I)}{|I|} \leq \frac{bk + k - 1 - \Sigma_{i=1}^{b}\alpha_i}{2bk - \Sigma_{i=1}^{b}\alpha_i}.$$

By the lower bound on $\Sigma_{i=1}^{b}\alpha_i$ above, and the arithmetic observation that $\frac{u-y}{v-y} < \frac{u-x}{v-x}$, if $u < v$ and $x < y < v$, we have that

$$\frac{bk + k - 1 - \Sigma_{i=1}^{b}\alpha_i}{2bk - \Sigma_{i=1}^{b}\alpha_i} \leq \frac{bk + k - 1 - (b + k - 1)}{2bk - (b + k - 1)} = \frac{b(k-1)}{b(2k-1) - k + 1}.$$

Clearly, $\max_{|I|=n, I \in \mathcal{L}(G)}\{\mathrm{FIFO}(I) - \mathrm{LRU}(I))\}$ is unbounded as a function of n. Since $\lim_{b\to\infty} \frac{b(k-1)}{b(2k-1)-k+1} = \frac{k-1}{2k-1}$, we have $\mathrm{Max}^G(\mathrm{FIFO}, \mathrm{LRU}) \leq \frac{k-1}{2k-1} = \frac{1}{2} - \frac{1}{4k-2}$. □

From [14] and Lemma 1, we have the following:

Theorem 1. $\mathcal{I}(\mathrm{FIFO}, \mathrm{LRU}) = [-1 + \frac{1}{k}, \frac{1}{2} - \frac{1}{4k-2}]$.

3.1 FWF

FWF performs very badly compared to the other algorithms considered here, LRU, FAR, and FIFO. The following is folklore:

Lemma 2. *For any sequence I and any conservative or marking algorithm $\mathcal{A}$, we have $\mathcal{A}(I) \leq \mathrm{FWF}(I)$.*

Thus, for any graph G, $\mathrm{Min}^G[\mathrm{FWF}, \mathcal{A}] = 0$, where $\mathcal{A}$ is either FAR, LRU, or FIFO. Hence, LRU, FIFO, and FAR all dominate FWF.

Theorem 2. *For the path access graph P_N, where $N \geq k+1$ (for* LRU*, for any graph containing P_{k+1}), $\mathcal{I}^{P_N}[\mathrm{FWF}, \mathcal{A}] = \left[0, 1 - \frac{1}{k}\right]$, where $\mathcal{A} \in \{\mathrm{LRU}, \mathrm{FAR}\}$.*

For FWF versus FIFO, a result almost as tight holds:

Theorem 3. *For any graph G containing a path with $k+1$ vertices, if k is odd, then $\mathcal{I}^G[\mathrm{FWF}, \mathrm{FIFO}] = \left[0, 1 - \frac{1}{k}\right]$, and if k is even, then $[0, \frac{k^2-k-1}{k^2}] \subseteq \mathcal{I}^G[\mathrm{FWF}, \mathrm{FIFO}] \subseteq \left[0, \frac{k-1}{k}\right]$.*

4 Path Graphs

Lemma 3. *For the path access graph P_N, we have* $\mathrm{Max}^{P_N}(\mathrm{FIFO}, \mathrm{LRU}) \leq \frac{1}{2} - \frac{1}{2k}$.

Proof. Consider any request sequence I. We divide the sequence up into phases as described now (these are *not* k-phases). Initially, define a direction by where LRU makes its kth fault compared with its cache content. Without loss of generality, we assume this happens going to the right on the path.

We start the first phase with the first request and later explain how subsequent phases are started. In all the phases, we start to the left (relatively). In all phases, except the first, LRU has the first $k-1$ distinct pages that will be requested during that phase in cache. In all phases, the first fault by LRU in the phase, after having processed the first $k-1$ distinct pages, is to the right. We maintain this as an invariant that holds at the start of any phase, though the direction can change, as we will get back to at the end of the proof. The exception in the first phase, adding an extra $k-1$ faults to the cost of LRU as compared with the analysis below, will not influence the result in the the limit for the length of the request sequence going towards infinity.

We want to analyze a phase where LRU faults to the right before it faults to the left again. These faults to the right may not appear consecutively. There may be some faults in a row, but then there may be hits and then faults again, etc. Thus, assume that there are m maximal subsequences of requests to the right where LRU faults—all of this before LRU faults going to the left again. Assume further that these maximal subsequences of requests give rise to $s_1, s_2, \ldots, s_m$ faults, respectively, where, by definition, $m \geq 1$, and let $s = \Sigma_{i=1}^{m} s_i$.

For now, we assume that for all i, $s_i < k$. Thus, LRU moves left and right at least m times; maybe more times where it does not give rise to faults. Since it does not fault going to the left during these turns, the faults are to pages further and further to the right. Let E_{right} denote the extreme rightmost position it reaches during these faults to the right.

When LRU faults again to the left after having processed E_{right}, we consider the leftmost node E_{left}, where LRU faults after the s faults described above, but before it faults to the right again. We end the phase with the first request to E_{left} after the s faults. We define subsequent phases inductively in the same way, starting with the first request not included in the previous phase, possibly leaving an incomplete phase at the end.

We now consider the costs of the algorithms and the length of the sequence per phase. LRU faults s times going to the right during the m turns in the phase. Additionally, LRU must fault at least t times going from E_{right} to E_{left}, where t is defined by there being $k+t$ nodes between E_{left} and E_{right}, including both endpoints. This sums up to $s+t$ faults.

For FIFO, we postpone the discussion of the first s_1 distinct pages seen in a phase. Just to avoid any confusion, note that these pages are immediately to the right of E_{left} (the endpoint of the previous phase) and thus not the pages that LRU faults on. After that, consider the maximal subsequence of at most k distinct pages. This subsequence starts with the (s_1+1)st distinct request (the last request to it before the s_2 faults) and continues up to, but not including the first request that LRU has one of its s_2 faults on. We know that there are at

most k pages there, because LRU only faults s_1 times there. Assume that FIFO faults f_1 times on this subsequence. Since FIFO is conservative, $f_1 \leq k$.

We define more such subsequences repeatedly, the $(m-1)$st of these ending just before LRU's first fault of the s_m faults, and the mth including the s_m faults and k of the $k+t$ nodes before we reach E_{left}. Finally, we return to the question of the first s_1 distinct pages seen in the phase. These overlap with the "t pages" from the previous phase; otherwise we would not have started the phase where we did. If FIFO faults on one of these pages when going through the t pages in the previous phase, it will not fault on them again in this phase. Thus, we only have to count them in one phase, and choose to do this in the previous phase. In total, FIFO faults at most $(\Sigma_{i=1}^{m} f_i) + t$ times, and for all i, $f_i \leq k$.

The difference between the cost of FIFO and LRU is then at most $(\Sigma_{i=1}^{m} f_i) + t - (s+t) = (\Sigma_{i=1}^{m} f_i) - s = (\Sigma_{i=1}^{m}(f_i - 1)) - (s - m)$.

From the analysis of FIFO above, knowing that on a subsequence of length at most k, FIFO can fault at most once on any given page, if it faults f_i times, the subsequence has at least f_i distinct pages. Given that the subsequence starts at the left end of the "s_i pages" and ends at the right end of the "s_i pages", all pages that FIFO faults on, except possibly the leftmost, must be requested at least twice, giving at least $2f_i - 1$ requests. So, the length of the sequence is at least $(\Sigma_{i=1}^{m}(2f_i - 1)) + t$. We now sum up over all phases, equipping each variable with a superscript denoting the phase number.

First, the total length, L, is at least

$$L \geq \Sigma_j(\Sigma_{i=1}^{m^j}(2f_i^j - 1)) + t^j = \Sigma_j(\Sigma_{i=1}^{m^j} 2f_i^j) - m^j + t^j.$$

Since s expresses how far we move to the right and t how far we move to the left, and the whole path has a bounded number of nodes N, we have that $\Sigma_j t^j \geq \Sigma_j s^j - N$. Thus, $L \geq (\Sigma_j(\Sigma_{i=1}^{m^j} 2f_i^j) - m^j + s^j) - N$.

I has a number of complete phases and then some extra requests in addition to that. There must exist a fixed constant c independent of I such that the cost of FIFO on the extra part of any sequence is bounded by c. This follows since there is a limit of N on how far requests can move to the right. So if requests never again come so far to the left that LRU faults, all requests thereafter are to only k pages. This added constant can also take care of the initial extra cost of $k-1$. Since we are just using a lower bound on the sequence length, we can ignore the length of a possibly incomplete phase at the end. Thus,

$$\begin{aligned}
\frac{\text{FIFO}(I) - \text{LRU}(I)}{|I|} &\leq \frac{c + \Sigma_j \Sigma_{i=1}^{m^j}(f_i^j - 1) - (s^j - m^j)}{-N + \Sigma_j(\Sigma_{i=1}^{m^j} 2f_i^j) - m^j + s^j} \leq \frac{c + \Sigma_j \Sigma_{i=1}^{m^j}(f_i^j - 1)}{-N + \Sigma_j \Sigma_{i=1}^{m^j} 2f_i^j} \\
&\leq \frac{c + \Sigma_j m^j(k-1)}{-N + \Sigma_j m^j 2k} = \frac{c + (k-1)\Sigma_j m^j}{-N + 2k\Sigma_j m^j}
\end{aligned}$$

The second inequality follows since $s^j \geq m^j$, and the third inequality follows because $\frac{f_i^j - 1}{2f_i^j} \leq \frac{1}{2}$ and $k \geq f_i$ implies that $\frac{f_i^j - 1}{2f_i^j} \leq \frac{k-1}{2k}$.

For sequences where the number of phases does not approach infinity, as argued above, FIFO's cost will be bounded. For the number of phases approaching infinity, $\lim_{j\to\infty} \frac{c+(k-1)\Sigma_j m^j}{-N+2k\Sigma_j m^j} = \frac{k-1}{2k} = \frac{1}{2} - \frac{1}{2k}$, which implies the result.

Now, for this proof, we assumed that $s_i < k$. If $s_i \geq k$, we simply terminate the phase after the processing of the s_i requests that LRU faults on, and continue to define phases inductively from there. All the bounds from above hold with $t = 0$ and the observation that FIFO will not fault on the first s_1 requests in the next phase. The direction of the construction is now reversed. In this process, whenever we reverse the direction as above, we also rename the variable s to t and t to s, such that s continues to keep track of movement to the right and t of movement to the left, and the inequality $\Sigma_j t^j \geq \Sigma_j s^j - N$ still holds. □

Theorem 4. $\mathcal{I}^{P_N}[\mathrm{FIFO}, \mathrm{LRU}] = \left[0, \frac{1}{2} - \frac{1}{2k}\right]$, *and* LRU *dominates* FIFO *on paths.*

Note that FAR and LRU perform identically on paths, so FAR also dominates FIFO with the same interval.

5 Star Graphs

We let S_N denote a star graph with N vertices. A star graph has a central vertex, s, which is directly connected to $N-1$ other vertices, none of which are directly connected. Thus, we could also see a star graph as a tree with root s and $N-1$ leaves, all located at a distance one from the root. The algorithms FAR and LRU behave identically on star graphs. Neither of them ever evicts the central vertex.

Theorem 5. *For* $\mathcal{A} \in \{\mathrm{LRU}, \mathrm{FAR}\}$*, we have*

$$\begin{aligned}\left[-\tfrac{1}{2} + \tfrac{1}{2(k-1)}, \tfrac{1}{4} + \tfrac{1}{8k-12}\right] &\subseteq \mathcal{I}^{S_N}[\mathrm{FIFO}, \mathcal{A}] \\ &\subseteq \left[-\tfrac{1}{2} + \tfrac{1}{2(k-1)} + \tfrac{1}{2k(k-1)}, \tfrac{1}{4} + \tfrac{1}{8k-12}\right]\end{aligned}$$

In [14], it was shown that $\mathrm{Max}^G(\mathrm{FIFO}, \mathrm{LRU}) \geq \frac{k-1}{2k-1} = \frac{1}{2} - \frac{1}{4k-2}$. The above result shows that for star access graphs, that bound can be decreased by a factor of approximately two.

Since LRU and FAR perform identically on stars, $\mathrm{Min}^{S_N}(\mathrm{FAR}, \mathrm{LRU}) = \mathrm{Max}^{S_N}(\mathrm{FAR}, \mathrm{LRU}) = 0$.

Theorem 6. *For* $\mathcal{A} \in \{\mathrm{LRU}, \mathrm{FAR}, \mathrm{FIFO}\}$*, we have* $\mathcal{I}^{S_N}[\mathrm{FWF}, \mathcal{A}] = \left[0, \frac{1}{2}\right]$.

6 Cycle Graphs

We consider graphs consisting of exactly one cycle, containing $N \geq k+1$ vertices, and define $r = N - k$. We will concentrate on the case where $r < k$, since otherwise the cycle is so large that for the algorithms considered here, it works as if it were an infinite path. Thus, for example, there are sequences where FIFO

performs worse than LRU, but on worst case sequences, simply going around the cycle, the algorithms perform identically.

In the following statements of theorems, we use $N = k+r$ with $1 \leq r \leq k-1$, and $X_r = r(x-1) + \left\lceil \frac{N}{2^x} \right\rceil$, where $x = \left\lfloor \log \frac{N}{r} \right\rfloor$, and $\hat{N}$ to denote N if N is even and $N-1$ otherwise.

Theorem 7. *For the cycle access graph C_N,*

$$\left[-1 + \frac{r}{k}, \frac{1}{2} - \frac{1}{4k-2}\right] \subseteq \mathcal{I}^{C_N}[\text{FIFO}, \text{LRU}] \subseteq \left[-1 + \frac{1}{k}, \frac{1}{2} - \frac{1}{4k-2}\right]$$

Theorem 8. *For the cycle access graph C_N,*

$$\mathcal{I}^{C_N}[\text{FWF}, \text{LRU}] = \left[0, 1 - \frac{1}{k}\right]$$

Theorem 9. *For the cycle access graph C_N,*

$$\left[-\frac{X_r - r}{k}, 1 - \frac{X_r}{k}\right] \subseteq \mathcal{I}^{C_N}[\text{FIFO}, \text{FAR}] \subseteq \left[-\frac{X_r - 1}{k}, 1 - \frac{1}{k}\right],$$

$$\left[-\frac{r\left(\left\lfloor \log \frac{\hat{N}}{r} \right\rfloor - 1\right)}{N-1}, 1 - \frac{X_r}{k}\right] \subseteq \mathcal{I}^{C_N}[\text{LRU}, \text{FAR}] \subseteq \left[-\frac{X_r - 1}{k}, 1 - \frac{1}{k}\right], \text{ and}$$

$$\left[0, 1 - \frac{X_r}{k}\right] \subseteq \mathcal{I}^{C_N}[\text{FWF}, \text{FAR}] \subseteq \left[0, 1 - \frac{1}{k}\right].$$

7 Concluding Remarks

Relative interval analysis has the advantage that it can separate algorithms properly when one algorithm is at least as good as another on every sequence and is better on some. This was reflected in the results concerning FWF which is dominated by the other algorithms considered for all access graphs. It was also reflected by the result showing that LRU and FAR have better performance than FIFO on paths. The analysis also found the expected result that FAR, which is designed to perform well on access graphs, performs better than both LRU and FIFO on cycles.

However, it is disappointing that the relative interval analysis of LRU and FIFO on stars and cycles found that FIFO had the better performance, confirming the original results by [14] on complete graphs. Clearly, access graphs cannot automatically be used with arbitrary quality measures for online algorithms to show that LRU is better than FIFO. To try to understand other quality measures for online algorithms better, it would be interesting to determine on which such measures access graphs are useful for separating LRU and FIFO, and on which they are not.

References

1. Albers, S., Favrholdt, L.M., Giel, O.: On paging with locality of reference. Journal of Computer and System Sciences 70(2), 145–175 (2005)
2. Angelopoulos, S., Dorrigiv, R., López-Ortiz, A.: On the separation and equivalence of paging strategies. In: SODA '07. pp. 229–237 (2007)
3. Borodin, A., El-Yaniv, R.: Online Computation and Competitive Analysis. Cambridge University Press (1998)
4. Borodin, A., Irani, S., Raghavan, P., Schieber, B.: Competitive paging with locality of reference. Journal of Computer and System Sciences 50(2), 244–258 (1995)
5. Boyar, J., Favrholdt, L.M.: The relative worst order ratio for on-line algorithms. ACM Transactions on Algorithms 3(2) (2007), article No. 22
6. Boyar, J., Favrholdt, L.M., Larsen, K.S.: The relative worst order ratio applied to paging. Journal of Computer and System Sciences 73(5), 818–843 (2007)
7. Boyar, J., Gupta, S., Larsen, K.S.: Access graphs results for lru versus fifo under relative worst order analysis. In: Fomin, F.V., Kaski, P. (eds.) SWAT '12. LNCS, vol. 7357, pp. 328–339. Springer, Heidelberg (2012)
8. Boyar, J., Gupta, S., Larsen, K.S.: Relative interval analysis of paging algorithms on access graphs (2013), arXiv:1305.0669 [cs.DS]
9. Boyar, J., Irani, S., Larsen, K.S.: A comparison of performance measures for online algorithms. In: Dehne, F.K.H.A., Gavrilova, M.L., Sack, J.R., Tóth, C.D. (eds.) WADS '09. LNCS, vol. 5664, pp. 119–130. Springer, Heidelberg (2009)
10. Chrobak, M., Noga, J.: LRU is better than FIFO. Algorithmica 23(2), 180–185 (1999)
11. Denning, P.J.: The working set model for program behaviour. Communications of the ACM 11(5), 323–333 (1968)
12. Denning, P.J.: Working sets past and present. IEEE Transactions on Software Engineering 6(1), 64–84 (1980)
13. Dorrigiv, R., López-Ortiz, A.: A survey of performance measures for on-line algorithms. SIGACT News 36(3), 67–81 (2005)
14. Dorrigiv, R., López-Ortiz, A., Munro, J.I.: On the relative dominance of paging algorithms. Theoretical Computer Science 410, 3694–3701 (2009)
15. Karlin, A.R., Manasse, M.S., Rudolph, L., Sleator, D.D.: Competitive snoopy caching. Algorithmica 3, 79–119 (1988)
16. Karlin, A.R., Phillips, S.J., Raghavan, P.: Markov paging. SIAM Journal on Computing 30(3), 906–922 (2000)
17. Koutsoupias, E., Papadimitriou, C.H.: Beyond competitive analysis. SIAM Journal on Computing 30(1), 300–317 (2000)
18. Sleator, D.D., Tarjan, R.E.: Amortized efficiency of list update and paging rules. Communications of the ACM 28(2), 202–208 (1985)
19. Young, N.: The k-server dual and loose competitiveness for paging. Algorithmica 11, 525–541 (1994)