Jordan Verkh-Haskell
David Chiu
Operating Systems
4/30/24

An Exploration of Belady's Anomaly

Belady's Anomaly, first discovered by Laszlo Belady, is an anomaly in machines using a paging system which occurs when an increase in memory leads to a decrease in performance. This is contrary to expectations, as an increase in memory should usually lead to an increase in performance.

To understand the anomaly, it is first important to understand the structures that allow it in the first place. Paging is a memory management scheme used in conjunction with virtual memory to make it possible to run a process in very little physical memory. Pages in virtual memory are matched with frames of the same size in physical memory, and a process is allocated a certain number of frames to use. As pages are needed, they are slotted into available frames. When there are no more available frames and a new page is requested, the page has to be fetched from memory, which is called a page fault. There are three types of page faults, compulsory, capacity, and conflict. Compulsory and capacity misses are unavoidable, but conflict misses can be avoided. Compulsory misses are a product of all frames being empty initially, so every call to a new page will cause a page fault. However, once all frames are full, a page must also be removed to bring a new page in. That is where page replacement schemes come in, they determine which page to remove when a new page is requested. Replacing a page takes time, and reduces CPU utilization, so it is important to minimize page faults as much as possible. Page replacement schemes can be split into two main categories, stack-based and non stack-based. Belady's Anomaly is a product of non stack-based replacement schemes (Geeks 2023), mainly the First In First Out (FIFO) scheme, although other schemes that function similarly to FIFO can suffer from Belady's Anomaly as well.

Stack replacement schemes monitor references to pages and keep an updated list of which page or pages are optimal to evict next. The methods through which they do this varies, but the principle remains the same. These schemes result in a page stack such that, for a given list of n pages at the end of the stack for an arbitrary number of frames, the list of n+1 pages should not change between the different numbers of frames. To put it a different way, at any given point in time the pages in memory with n frames available will be a subset of the pages in memory with n+1 frames available. This makes the page being replaced independent of the number of frames available, with replacement simply happening more often with less frames.

This is not the case for schemes that do not follow stack-replacement, as the page that gets evicted is dependent on its placement in the frames, which causes discrepancies between the pages in memory at different numbers of frames, such that there may be a page in memory with n frames that is not present in memory with n+1 frames. This can lead to a pattern of page faults occurring more often with a larger number of frames, as the page references can line up in a way such that a page is consistently requested that is present in n frames, but not in n+1 frames. To illustrate the anomaly, take a given sequence of page requests, and 2 machines, one of which has 3 frames, and the other which has 4. Belady's 1969 paper on the anomaly provides a usable sequence for demonstration, [1,2,3,4,1,2,5,1,2,3,4,5] (Belady 1). This sequence can be illustrated as follows for the Least Recently Used (LRU) algorithm.

1	2	3	4	1	2	5	1	2	3	4	5
🟥 1			🟥 4			🟥 5			🟥 3		
	🟥 2			🟥 1			🟩 1			🟥 4	
		🟥 3			🟥 2			🟩 2			🟥 5
1	12	123	234	134	124	125	125	125	123	234	345

As illustrated, the LRU algorithm with 3 frames available has 10 page faults and 2 page hits. When the number of pages is increased to 4, the number of page faults decreases.

1	2	3	4	1	2	5	1	2	3	4	5
🟥 1				🟩 1			🟩 1				🟥 5
	🟥 2				🟩 2			🟩 2			
		🟥 3				🟥 5				🟥 4	
			🟥 4						🟥 3		
1	12	123	1234	1234	1234	1245	1245	1245	1235	1234	2345

The number of page faults decreases from 10 to 8, and the number of hits increases from 2 to 4. Importantly, the locations of the page hits remain the same when the number of frames increases, and the new hits are simply added. This can be understood by the bottom row in the table, which shows the number of frames in memory for every reference. The pages in memory with 3 frames are always a subset of the pages in memory with 4 frames, which means there will never be a page fault with 4 frames in a location where there wouldn't be with 3 frames. This prevents Belady's Anomaly, as it means there cannot possibly be more page faults with n+1 frames as there are with n frames. However, this is not the case for a scheme such as FIFO.

1	2	3	4	1	2	5	1	2	3	4	5
🟥 1			🟥 4			🟥 5					🟩 5
	🟥 2			🟥 1			🟩 1		🟥 3		
		🟥 3			🟥 2			🟩 2		🟥 4	
1	12	123	234	134	124	125	125	125	235	345	345

FIFO suffers from 9 page faults with 3 frames, so in this case it actually performs better than LRU, however the problem appears when FIFO is run with 4 frames.

1	2	3	4	1	2	5	1	2	3	4	5
🟥 1				🟩 1		🟥 5				🟥 4	
	🟥 2				🟩 2		🟥 1				🟥 5
		🟥 3						🟥 2			
			🟥 4						🟥 3		
1	12	123	1234	1234	1234	2345	1345	1245	1234	1234	2345

As shown here, with this reference pattern, FIFO suffers from an additional page fault when the number of frames is increased, bringing it from 9 page faults with 3 frames to 10 page faults with 4 frames. The cause of this is based on the first reference of 5, where page 1 is evicted right before it is referenced again. This leads to a cycle that causes every page to be evicted immediately before being referenced again, causing a continuous chain of page faults. This occurs because after the first reference of 5, the pages in memory for 3 frames are [1,2,5], whereas for 4 frames they are [2,3,4,5]. This discrepancy illustrates that n+1 pages are inconsistent with n+1 frames, leading to Belady’s Anomaly. Even though this discrepancy is fixed 2 references later, where 3 frames contain [1,2,5] and 4 frames contain [1,2,4,5], it is too late and FIFO has entered its worst case cycle. The solution to this problem is illustrated very well in LRU, where a page reference should refresh the priority of that page, but in FIFO, page priority is not updated when a page is re-referenced, leading to a recurring cycle of page faults independent of page hits. This can be clearly illustrated by removing the page hits from the table.

1	2	3	4	1	2	5	3	4
1			4			5		
	2			1			3	
		3			2			4
1	12	123	234	134	124	125	235	345

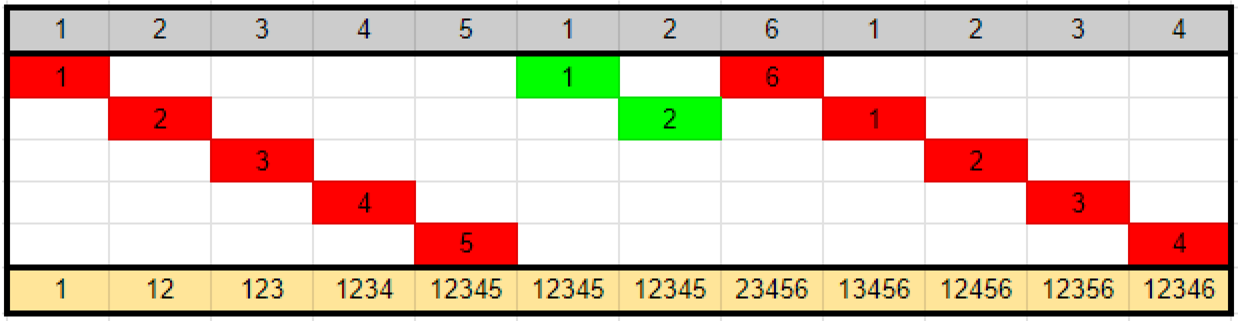
1	2	3	4	5	1	2	3	4	5
1				5				4	
	2				1				5
		3				2			
			4				3		
1	12	123	1234	2345	1345	1245	1234	1234	2345

When the page hits are removed, the page fault pattern of FIFO is illustrated as being cyclic, independent of any page hits. Therefore, the number of frames available has a direct impact on the number of page faults, because placement of a page in a given frame is not independent of page faults. With this observation, it becomes the case that a greater number of frames will actually lead to an equal or greater number of page faults.

Take the reference pattern [1,2,3,4,5,1,2,6,1,2,3,4]. This pattern is nearly identical, but the initial cycle has an initial 5 at the end, the highest reference is 6 instead of 5, and the cycle at the end is cut short 1 to maintain the total of 12 references. When tested with 3, 4, and 5 frames, the number of page faults is 10 between all tests.

1	2	3	4	5	1	2	6	1	2	3	4
1			4			2			2		4
	2			5			6				
		3			1			1		3	
1	12	123	234	345	145	125	126	126	126	236	346

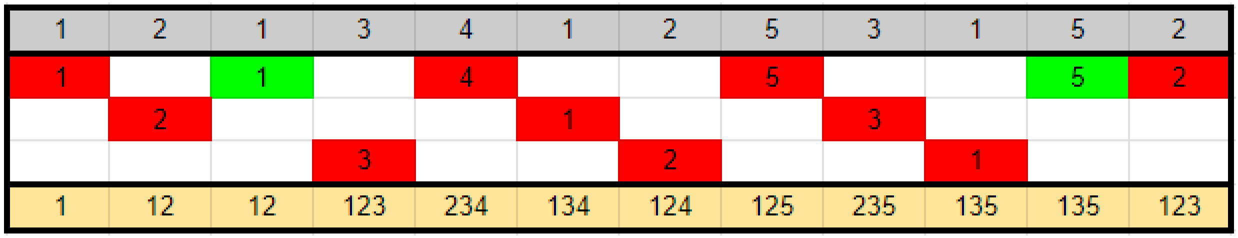
1	2	3	4	5	1	2	6	1	2	3	4
1				5							
	2				1			1		3	
		3				2			2		4
			4				6				
1	12	123	1234	2345	1345	1245	1256	1256	1256	2356	2456



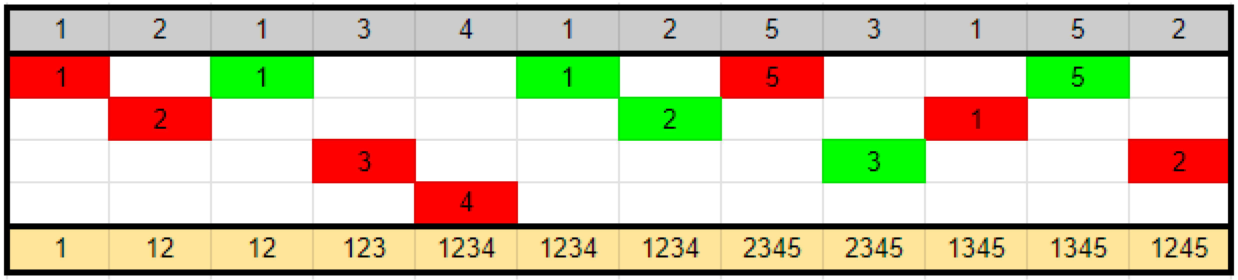
1	2	3	4	5	1	2	6	1	2	3	4
1					1		6				
	2					2		1			
		3							2		
			4							3	
				5							4
1	12	123	1234	12345	12345	12345	23456	13456	12456	12356	12346

While this does not demonstrate Belady's anomaly directly, as the number of page faults does not change, it does make a good demonstration of what patterns lead to Belady's anomaly. Loops are the main culprit for Belady's anomaly, with FIFO suffering the most in cases where the number of pages is only slightly larger than the number of frames. This leads to the pattern in which a page is referenced immediately after being evicted, which can be seen demonstrated in all of these examples. Notably, while all 3 examples have the same number of page faults, both 3 and 4 frames have page hits after the 6 compulsory misses, page faults due to a page never having been loaded into memory, whereas the 5 frame example has no page hits following the compulsory misses. It is also worth noting that the lack of Belady's anomaly here may be explained by the cluster rule for reference string production in Belady's original paper, where a requirement of the rule is that s<l<a<2s, where s is the smaller number of frames, l is the larger number of frames, and a is the number of unique pages (Belady 3). Given this, Belady's anomaly would not occur between 3 and 4 frames, as a is not less than 2s. While Belady's anomaly does not occur between 4 and 5 frames either, the page hits still change locations. This meets the requirement for Belady's anomaly to occur, although this particular string does not lead to that end result.

It can also be demonstrated that Belady's anomaly does not occur in cases where loops are minimal or not present at all, such as the pattern [1,2,1,3,4,1,2,5,3,1,5,2]



1	2	1	3	4	1	2	5	3	1	5	2
1		1		4			5			5	2
	2				1			3			
			3			2			1		
1	12	12	123	234	134	124	125	235	135	135	123



1	2	1	3	4	1	2	5	3	1	5	2
1		1			1		5			5	
	2					2			1		
			3					3			2
				4							
1	12	12	123	1234	1234	1234	2345	2345	1345	1345	1245

As this example shows, going from 3 frames to 4 frames causes a drop from 10 page faults to only 7. In addition, despite failing to meet stack consistency on references 8 and 12, there is no loss, because there is a page fault in the 3 frame example in those places as well. This leads to the conclusion that loops tend towards Belady's anomaly, as a scheme without reference refresh priority will inevitably lead towards evicting more common pages. It is worth noting that in the case of a complete cycle in references, with a purely looping pattern, such as [1,2,3,4,1,2,3,4,1,2,3,4], all replacement schemes would suffer maximum or near maximum page fault rate. However, in that case, the fault does not lie with the page replacement scheme, and instead calls for more efficient code or more frames to compensate.

One final thing that is interesting to note is the relationship between the ratio of pages to frames and the causation of the anomaly. A 2009 paper by McMaster, Sambasivam, and Anderson ran tests on both FIFO and random replacement with a wide variety of parameters to test the regularity and scale of Belady's anomaly. In their paper, they found that in FIFO, Belady's anomaly occurs most frequently when the number of frames is 75% the number of pages (McMaster et al. 8). This matches with the outcome that Belady's outcome consistently occurs in a 4 frame system with a reference string containing 5 unique pages, as 80% is the closest frame value to the 75% that Belady's anomaly most commonly occurs at. This leads to a final conclusion that Belady's anomaly can occur when the number of frames is between 50% and 100% of the number of unique pages, with the most common occurrence in the middle, at 75%.

In conclusion, Belady's anomaly is an anomaly in paging systems using FIFO or similar systems that causes an increase in available frames to reduce performance. This occurs as a result of the pages being evicted in a sequence that leads to a smaller number of frames containing the requested page where a larger number of frames may not. This outcome occurs most commonly when the number of frames is about 75% of the number of unique pages. While Belady's anomaly is uncommon, the requirements for it to occur are easy to fulfill, so it is worth being concerned about. Belady's anomaly is most easily circumvented by using a stack-based replacement scheme, as those cannot suffer from the anomaly.

Bibliography

L. A. Belady, R. A. Nelson, and G. S. Shedler. An anomaly in spacetime characteristics of certain programs running in a paging machine. *Communications of the ACM*, 12(6):349–353, June 1969.
https://dl.acm.org/doi/pdf/10.1145/363011.363155

aagarwal2, aishwary. "Belady's Anomaly in Page Replacement Algorithms." *GeeksforGeeks*, 2 May 2023, www.geeksforgeeks.org/beladys-anomaly-in-page-replacement-algorithms/.

McMaster, Kirby, et al. "How Anomalous Is Belady's Anomaly? ." *Issues in Informing Science and Information Technology*, vol. 6, 2009, pp. 825–836, https://doi.org/10.28945/1101.