Jeff Gensler
jgensl2@uic.edu
jgensler

361 HW3 Report

It is quite easy to see that any program that behaves like infinite memory is desirable. What does this really mean? Any program with the fewest number of page misses and highest number of page hits is desirable. Based on the sample input files given, it is clear that photographs with little to no variation in color value would perform the best. But having small variation in color value is not enough to succeed. The timing of the variation is also important. If there was just enough variation in color value to exceed the the memory size AND this variation was spread out in very quick succession, it is obvious that the cache would perform poorly. Take the two pictures supplied that performed poorly, light_drops.raw and GrandeJatte.raw. If we look at the pictures themselves, we can see there is a lot of **unsustained** variation. “Reading” from left to right, we are constantly seeing new colors (colors unlike the previous 256 seen) In the case of light_drops it is probably the same few colors per line. In GrandeJatte we are seeing all sorts of colors. In either case, there is a poor locality of reference as well as poor sustained referencing. Contrasting this, if we look at VRUPL_logo.raw versus LUG_newbiews.raw, we can see the same trends but this time with high locality of reference. The locality of reference is evident by the low variation of colors. Again, we see that variation in color is not enough. Here, we can see that VRUPL’s diagonal content aids in seeing the same color for shorts bursts of time, which is performant for our page table.

What the data says:

As we analyze based on the size of the frame table, I will start with a small frame size and increase until the pictures perform near unlimited. From the start, we can see that VRUPL is far more performant that than the other three. LUG performs at 50% of its max, light_drops at 18%, and GrandeJatte at 11%. If we take a look at average numner of time swapped, we see something quite interesting. We see that LUG has a larger number of times swapped yet more hit percentage than light_drops. This is most likely from the fact that LUG has very few **types** of colors but very frequent **changes**. light_drops has many types (shades) of colors, so the average is actually very low (each very specific color may only be needed a few times).

Let us look at the difference between a frame table size of 1536 and 3072.
1536: light_drops percent of max: 58% LUG: 72%
3072: light_drops: 81% LUG: 78%
Why would the overall performance of light_drops somehow radically surpass that of LUG?
I would attribute this to LUG’s **change** in average times switched. Basically, how helpful was the increase in the cache in terms of times of replacement. For light_drops, the average is halved from 1536 to 3072! Yet for LUG, we are left with 80% of the previous run’s average number of times swapped. It must be the case that despite LUG’s simplicity, the cache is still not enough to capture small variations in color from letter to letter. What does this mean in terms of frame

tables? Gradual changes to required page tables are preferred to sharp drastic ones. This seems fairly obvious but the data brings us to this conclusion logically.