ECE60827 HW Sim Assignment #2 Lab Report

Conor X Devlin, PUID: 0035599728

Question #1: "Flat Scheduler"

Looking at the gaussian and lud plots (Appendix A, Charts 1-3 and Appendix B, Charts 1-3) we find that in going from *baseline* to *8cluster* there are less but longer and more 'intense' lines (going off the color scale) indicating that *8cluster* is performing as expected in the sense that it is applying more CTAs to more SMs and thus they're darker as they've got more capacity to hold SMs. This is then further reflected in the *flat* and plot where the intensity is highest (black) but there's fewer parallel lines meaning that less SMs are being scheduled per sets of instructions as they're being utilized more efficiently than just *baseline* or *8cluster*. The *flat* scheduler features a similar chart, in the sense consecutive cores share similar levels of CTAs (hence the connected thick lines), to both *baseline* and *8cluster* save for the fact that its SMs end up being utilized more intensely (more CTAs per core) for both *Lud* and *Gaussian* and thus have fewer parallel lines overall.

In terms of IPC we find that *baseline* has the lowest average (across all benchmarks) IPC (1,387.9) compared to *8cluster* (2,492.8) and *flat* (2,363.9) (Appendix C, Chart 1 and Table 1). The introduction of the flat scheduler had a minor loss (-9.30%) in the IPC. Looking at average L2 cache misses (Appendix C, Chart 2 and Table 2) we find that *baseline* features averages misses of 1,065,116 whereas *flat* comes in the best at 1,060,697 and *baseline* performs the worst at 1,077,660. Lastly considering average L1 misses (Appendix C, Chart 3 and Table 3), we find that *baseline* performs the best at 2,360,188 and *8cluster* and *flat* having an increase of 1.43% (2,394,048) and 2.67% (2,423,141) respectively. Based on these statistics only, there is a large jump in performance (as measured by IPC) going from *baseline* to either *8cluster* or *flat* but then between them there exists only minor (as in <%1.0) differentials which suggest that my implementation could've been more effective (a more efficient flat scheduler that is) or that simply throwing cores at these specific benchmarks is the main factor in general improvements to a point as the IPC performance nearly doubled with <3.0% degradation in cache misses which is a largely positive improvement overall.

Question #2: "Greedy Scheduler"

Looking at the gaussian and lud plots (Appendix A, Charts 1, 2, 4 and Appendix B, Charts 1, 2, 4) we find that our *greedy* scheduler implementation still has lines but the individual SMs are spaced out and *not connected* (SM#0 then SM#2 then SM#4…) unlike *flat* which had all SMs directly connected in its scheduling (SM#0 then SM#1 then SM#2). Once can tell that my implementation of *greedy* still fills up certain SMs to max occupancy but doesn't seem to fill the immediate next SM rather going to the next SM in the following cluster this is why, for the *greedy* implementation all clusters have a few SMs scheduled in the first few instruction cycles. *Flat* on the hand, would schedule all cores in a SM cluster before moving on to the next immediate (hence the contiguous line). Observing both the *gaussian* and *lud* charts we've selected the major difference between my implementations of *flat* and *greedy* mostly come down to the overall occupancy of the clusters *greedy* and *flat* both (likely) schedule across the same amount of SMs but *greedy* appears to schedule across more clusters comparatively.

Next, looking at IPC again, we find that *baseline* has the lowest average (across all benchmarks) IPC (1,387.9) compared to *8cluster* (2,492.8) and *greedy* (2,417.0) which outperformed *flat* (2363.9) but still underperforms *8cluster* (Appendix C, Chart 1 and Table 1). The introduction of the *greedy* scheduler outperforms *flat* (3.83%) and *baseline* (74.15%) in the IPC but still underperforms *8cluster* (-5.46%). Looking at the average L2 cache misses (Appendix C, Chart 2 and Table 2) we find that *greedy* outperforms *baseline* (2.60%), *8cluster* (1.45%) and *flat* (1.04%) but only marginally. Finally considering average L1 misses (Appendix C, Chart 3 and Table 3), *greedy* slightly underperforms *baseline* (-0.60%) but then slightly outperforms *8cluster* (0.82%) and *flat* (2.01%). Overall, we find a analogous relationship between both *flat* and *greedy*, both outperforming *baseline* as measured by IPC and then either marginally under- or outperforming each other and *8cluster* with only minor percentage gains or losses between them. Once again this suggests that for all of the benchmarks, we're going to experience the largest shift in performance merely going from *baseline* to *8cluster* and that *flat* or *greedy* can provide similar possible improvements but they must be tweaked per benchmark to achieve this.

[My reference graph for b+tree is found in Appendix D]

Appendix A: SASS Gaussian Plots

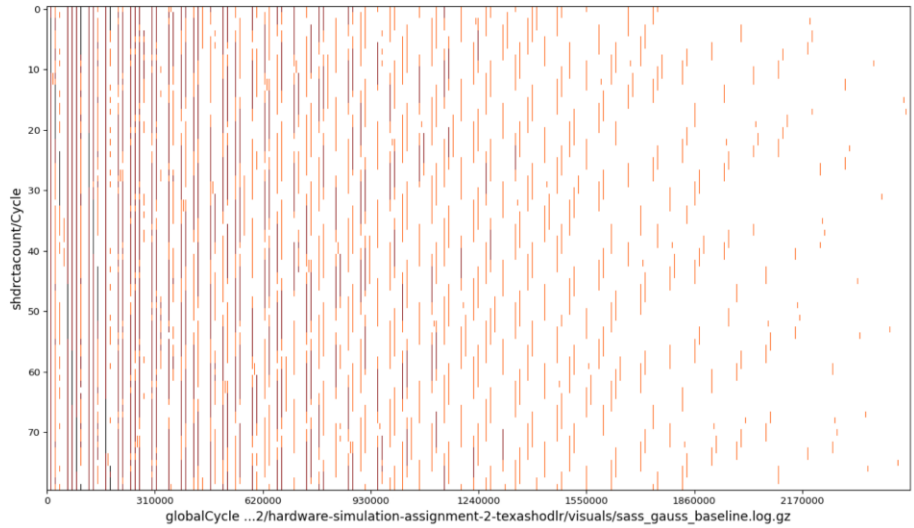

Chart 1: Baseline SASS Gaussian

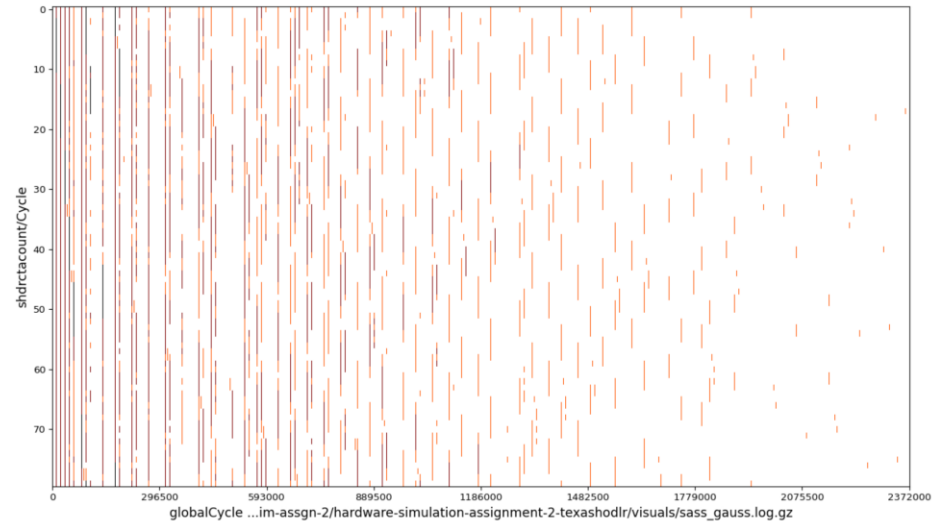

Chart 2: 8cluster SASS Gaussian

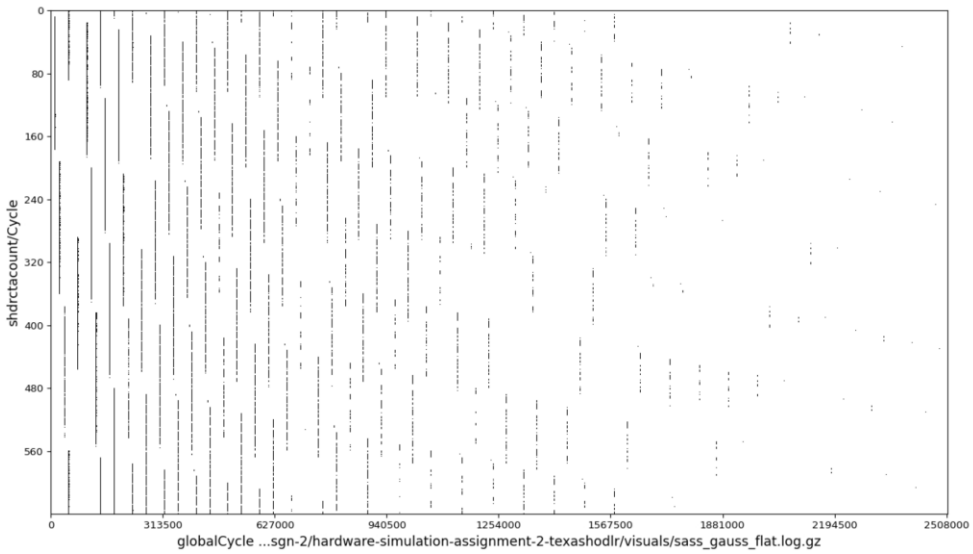

Chart 3: Flat Scheduler SASS Gaussian

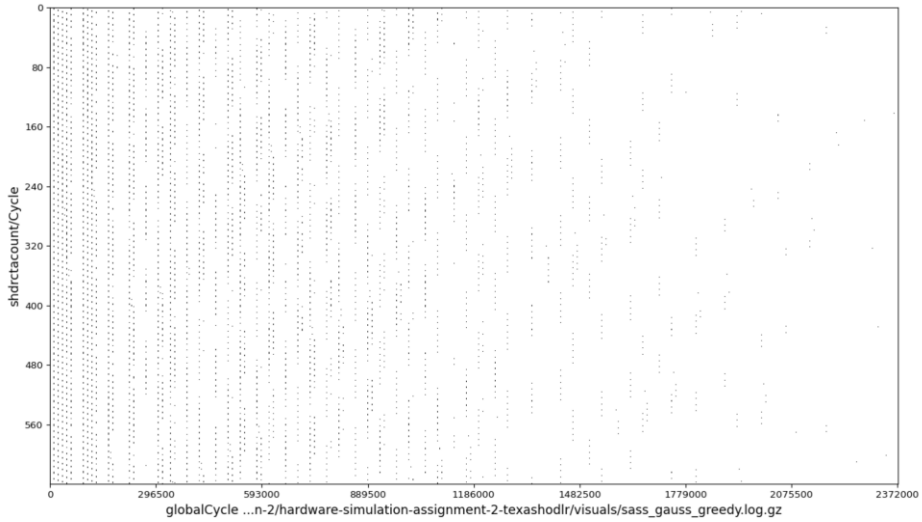

Chart 4: Greedy Scheduler SASS Gaussian

Appendix B: SASS Lud Plots

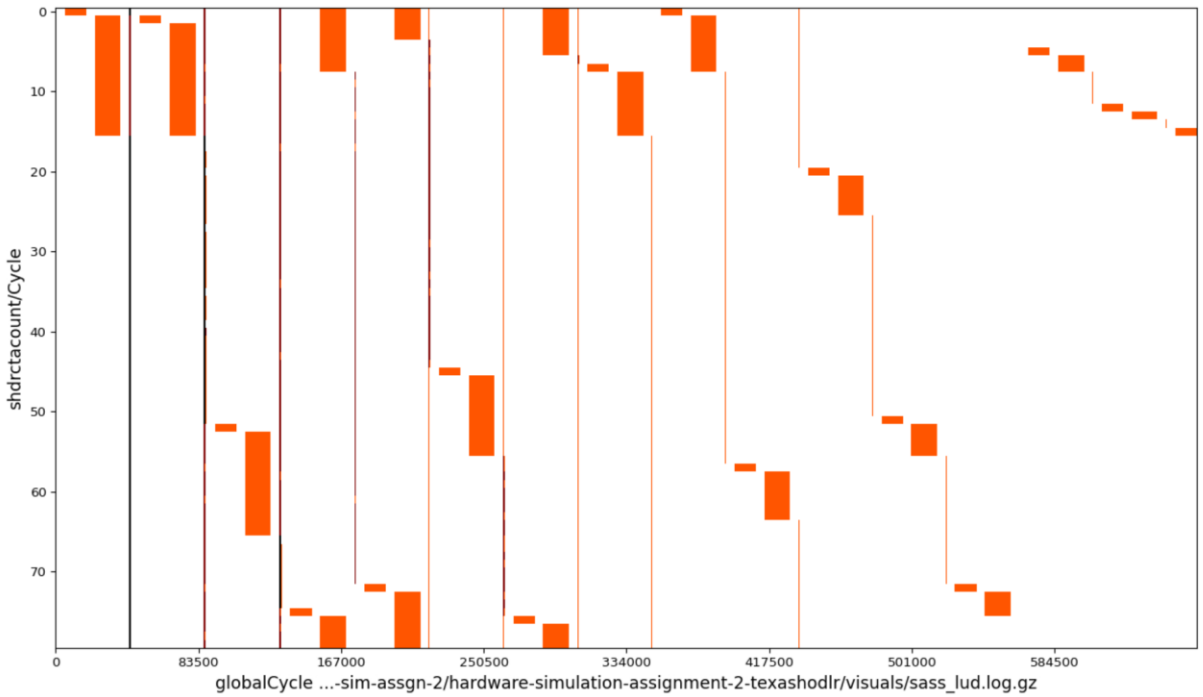

Chart 1: Baseline SASS Lud

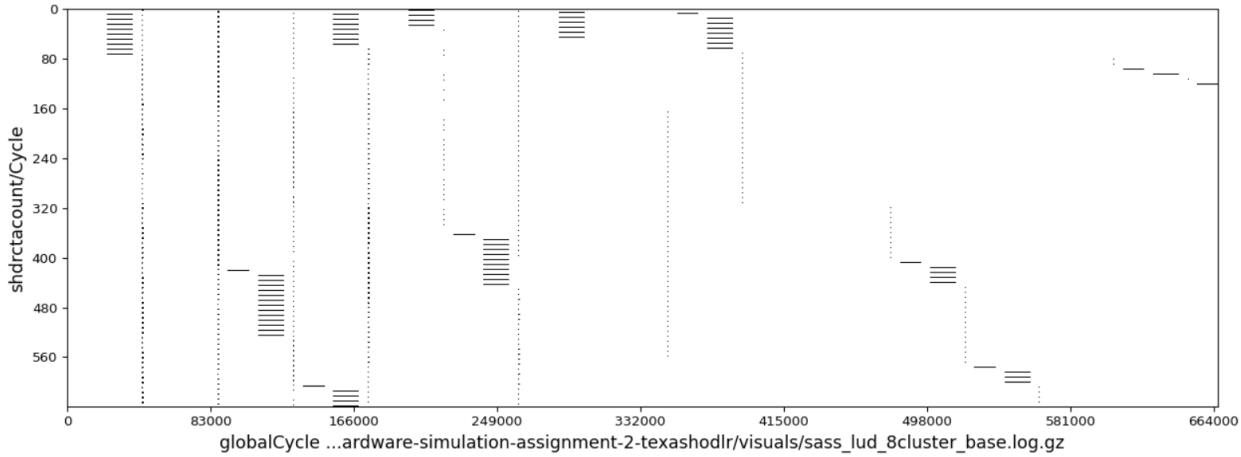

Chart 2: 8cluster SASS Lud

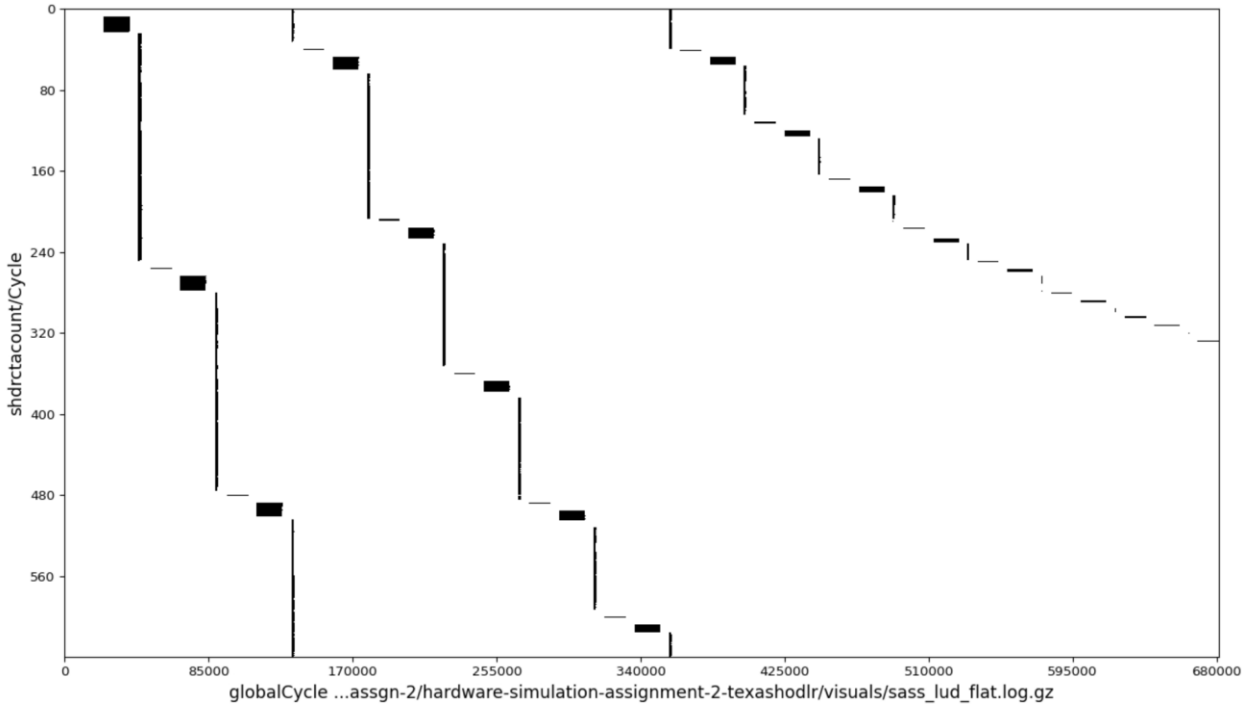

Chart 3: Flat Scheduler SASS Lud

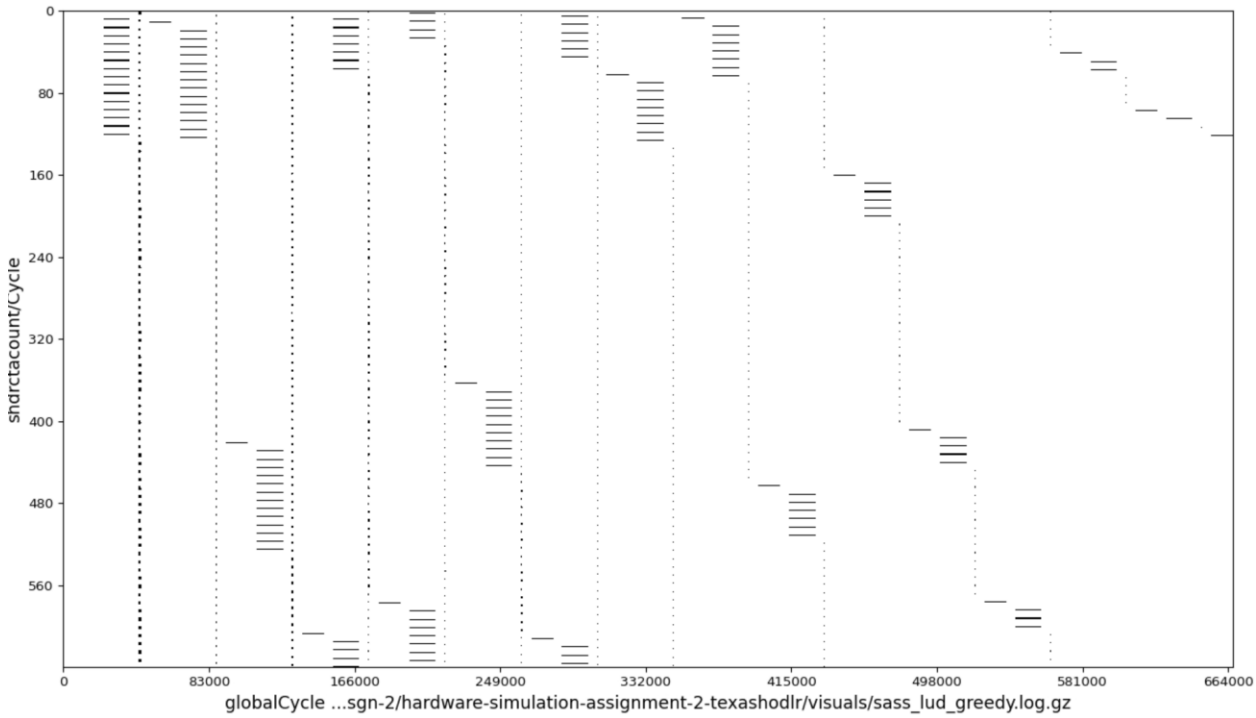

Chart 4: Greedy Scheduler SASS Lud

Appendix C: IPC, L1 and L2 Data and Plots

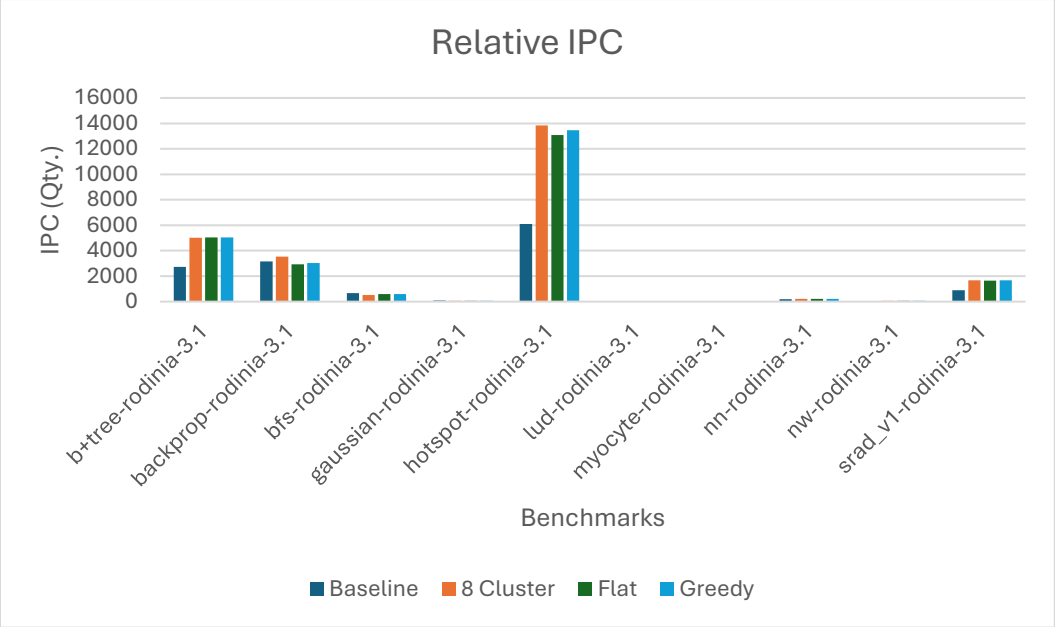

Chart 1: Relative IPCs

Relative IPC	Baseline	8 Cluster	Flat	Greedy	8 v. base	Flat v. base	Greedy v. base
b+tree-rodinia-3.1	2720.5005	5005.6147	5030.1743	5032.8486	183.9960956	184.898856	184.9971577
backprop-rodinia-3.1	3159.5266	3521.498	2923.8472	3032.1399	111.4565074	92.5406737	95.96817131
bfs-rodinia-3.1	658.5748	516.5286	595.1138	597.8841	78.43127311	90.3638888	90.78453958
gaussian-rodinia-3.1	88.0491	58.9381	55.6225	58.9577	66.93776541	63.1721392	66.96002571
hotspot-rodinia-3.1	6101.5273	13829.9707	13085.2529	13452.3252	226.6640797	214.458647	220.4747195
lud-rodinia-3.1	39.0404	38.3718	37.6555	38.3718	98.28741509	96.4526491	98.28741509
myocyte-rodinia-3.1	0.1368	0.2588	0.2587	0.2588	189.1812865	189.108187	189.1812865
nn-rodinia-3.1	198.2916	214.2039	204.0006	214.2039	108.024697	102.879093	108.024697
nw-rodinia-3.1	23.874	58.8908	57.5732	58.8908	246.6733685	241.154394	246.6733685
srad_v1-rodinia-3.1	889.2903	1684.0114	1649.1361	1684.0114	189.3657673	185.444067	189.3657673
Average	1387.88114	2492.82868	2363.86348	2416.98922			

Table 1: Relative IPCs

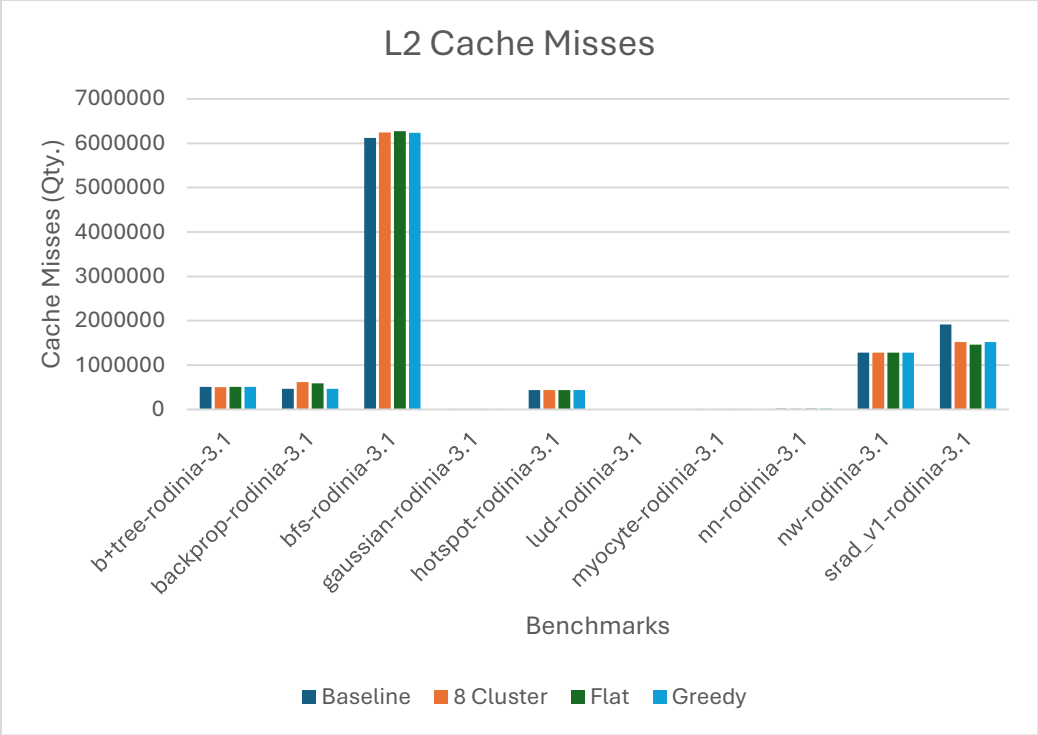

Chart 2: L2 Cache Misses

L2_total_cache_misses	Baseline	8 Cluster	Flat	Greedy	8 v. base	Flat v. base	Greedy v. base
b+tree-rodinia-3.1	507855	505194	509651	508959	-0.52396846	0.353644249	0.217384883
backprop-rodinia-3.1	466956	615150	591423	469932	31.7361807	26.65497392	0.637319148
bfs-rodinia-3.1	6121713	6246095	6273215	6233216	2.03181691	2.474830166	1.821434621
gaussian-rodinia-3.1	10998	10998	10998	10998	0	0	0
hotspot-rodinia-3.1	437248	437248	437375	437248	0	0.029045302	0
lud-rodinia-3.1	8192	8192	8192	8192	0	0	0
myocyte-rodinia-3.1	14679	14679	14679	14679	0	0	0
nn-rodinia-3.1	16037	16037	16037	16037	0	0	0
nw-rodinia-3.1	1280384	1280384	1280384	1280384	0	0	0
srad_v1-rodinia-3.1	1912540	1517191	1465019	1517191	-20.6714108	-23.3993015	-20.67141079
Average	1077660.2	1065116.8	1060697.3	1049683.6			

Table 2: L2 Cache Misses

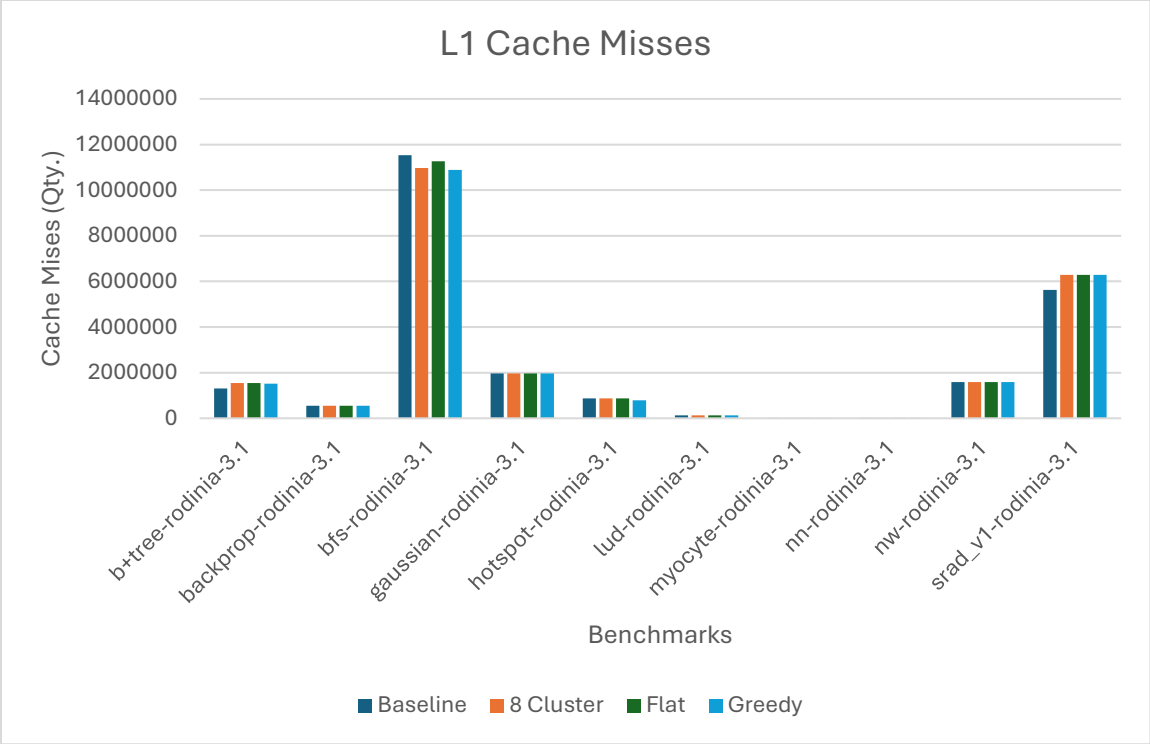

Chart 3: L1 Cache Misses

L1_total_cache_misses	Baseline	8 Cluster	Flat	Greedy	8 v. base	Flat v. base	Greedy v. base
b+tree-rodinia-3.1	1304537	1544037	1538835	1514572	18.359004	17.96024183	16.10034825
backprop-rodinia-3.1	549083	550790	550646	546949	0.31088196	0.284656418	-0.388647982
bfs-rodinia-3.1	11530619	10965778	11271354	10883515	-4.89861819	-2.2484916	-5.612049102
gaussian-rodinia-3.1	1967014	1967004	1967030	1967030	-0.00050838	0.000813416	0.000813416
hotspot-rodinia-3.1	866176	870616	868808	789940	0.5125979	0.303864342	-8.80144451
lud-rodinia-3.1	130432	131072	131072	131072	0.49067713	0.490677134	0.490677134
myocyte-rodinia-3.1	17316	17316	17316	17316	0	0	0
nn-rodinia-3.1	16037	16037	16037	16037	0	0	0
nw-rodinia-3.1	1589248	1589248	1589248	1589248	0	0	0
srad_v1-rodinia-3.1	5631421	6288591	6281072	6288591	11.6697011	11.53618243	11.66970113
Average	2360188.3	2394048.9	2423141.8	2374427			

Table 3: L1 Cache Misses

Appendix D: PTX B+Tree Validation

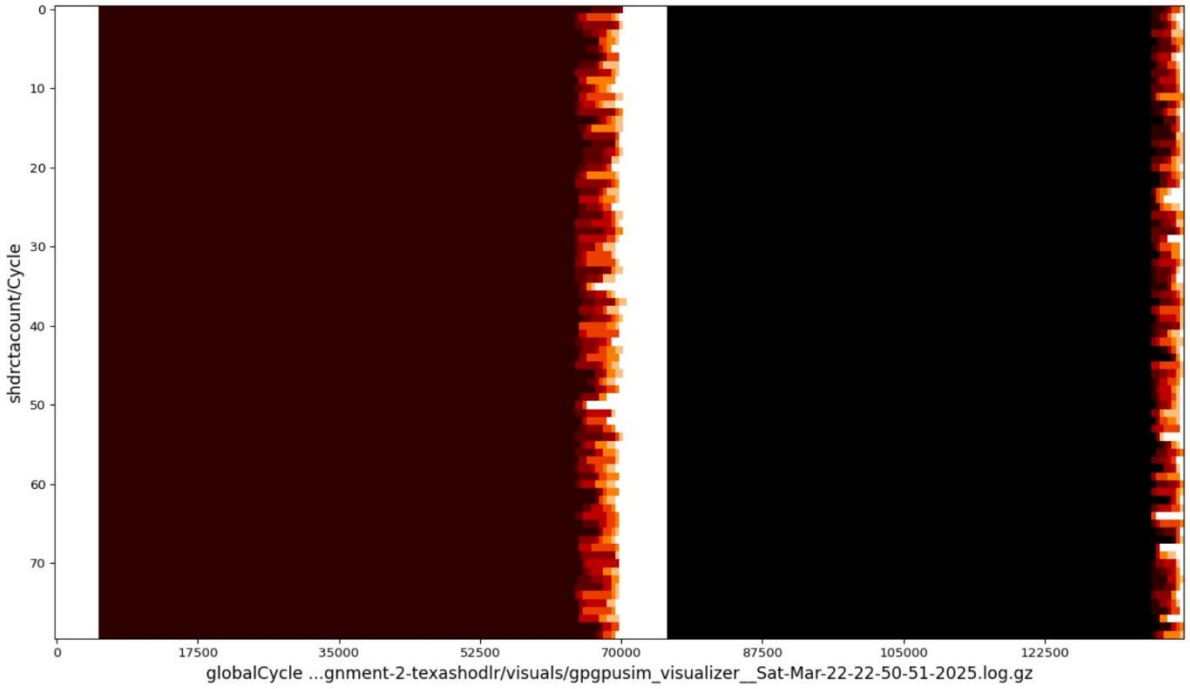

Chart 1: B+tree SASS validation