CSCC69-A2 ExecutionReport

2020-03-02

Tables

matmul

algo	memsize	hit count	miss count	clean evictions	dirty evictions	total reference	hit rate	miss rate
rand	50	2,053,642	995,166	974,797	20,319	3,048,808	67.3588	32.6412
	100	2,724,449	324,359	320,021	4,238	3,048,808	89.3611	10.6389
	150	2,952,651	96,157	94,361	1,646	3,048,808	96.8461	3.1539
	200	2,992,492	56,316	54,823	1,293	3,048,808	98.1529	1.8471
clock	50	2,007,618	1,041,190	1,040,163	977	3,048,808	65.8493	34.1507
	100	2,007,781	1,041,027	1,039,964	963	3,048,808	65.8546	34.1454
	150	3,015,630	33,178	32,066	962	3,048,808	98.9118	1.0882
	200	3,015,942	32,866	31,705	961	3,048,808	98.9220	1.0780
fifo	50	1,921,581	1,127,227	1,104,750	22,427	3,048,808	63.0273	36.9727
	100	1,965,266	1,083,542	1,071,884	11,558	3,048,808	64.4601	35.5399
	150	3,014,416	34,392	33,211	1,031	3,048,808	98.8720	1.1280
	200	3,014,948	33,860	32,640	1,020	3,048,808	98.8894	1.1106
lru	50	2,007,632	1,041,176	1,040,155	971	3,048,808	65.8497	34.1503
	100	2,042,381	1,006,427	1,005,364	963	3,048,808	66.9895	33.0105
	150	3,015,956	32,852	31,741	961	3,048,808	98.9225	1.0775
	200	3,015,971	32,837	31,676	961	3,048,808	98.9230	1.0770
opt	50	2,461,370	587,438	586,423	965	3,048,808	80.7322	19.2678
	100	2,956,038	92,770	91,707	963	3,048,808	96.9572	3.0428
	150	3,022,221	26,587	25,475	962	3,048,808	99.1280	0.8720
	200	3,029,571	19,237	18,075	962	3,048,808	99.3690	0.6310

blocked

algo	memsize	hit count	miss count	clean evictions	dirty evictions	total reference	hit rate	miss rate
rand	50	3,513,426	8,470	6,129	2,291	3,521,896	99.7595	0.2405
	100	3,516,644	5,252	3,577	1,575	3,521,896	99.8509	0.1491
	150	3,517,600	4,296	2,852	1,294	3,521,896	99.8780	0.1220
	200	3,518,083	3,813	2,424	1,189	3,521,896	99.8917	0.1083
clock	50	3,516,707	5,189	2,929	2,210	3,521,896	99.8527	0.1473
	100	3,517,899	3,997	2,710	1,187	3,521,896	99.8865	0.1135
	150	3,517,986	3,910	2,675	1,085	3,521,896	99.8890	0.1110
	200	3,518,761	3,135	1,994	941	3,521,896	99.9110	0.0890
fifo	50	3,515,398	6,498	4,428	2,020	3,521,896	99.8155	0.1845
	100	3,517,561	4,335	2,904	1,331	3,521,896	99.8769	0.1231
	150	3,517,673	4,223	2,791	1,282	3,521,896	99.8801	0.1199
	200	3,518,741	3,155	1,988	967	3,521,896	99.9104	0.0896
lru	50	3,516,782	5,114	2,852	2,212	3,528,196	99.8548	0.1452
	100	3,518,128	3,768	2,710	958	3,521,896	99.8930	0.1070
	150	3,518,139	3,757	2,666	941	3,521,896	99.8933	0.1067
	200	3,518,237	3,659	2,519	940	3,521,896	99.8961	0.1039
opt	50	3,518,186	3,710	2,702	958	3,521,896	99.8947	0.1053
	100	3,518,898	2,998	1,954	944	3,521,896	99.9149	0.0851
	150	3,519,389	2,507	1,410	947	3,521,896	99.9288	0.0712
	200	3,519,639	2,257	1,118	939	3,521,896	99.9359	0.0641

simpleloop

algo	memsize	hit count	miss count	clean evictions	dirty evictions	total reference	hit rate	miss rate
rand	50	7,925	2,955	330	2,575	10,880	72.8401	27.1559
	100	8,111	2,769	187	2,482	10,880	74.5496	25.4504
	150	8,173	2,707	141	2,416	10,880	75.1195	24.8805
	200	8,178	2,702	142	2,360	10,880	75.1654	24.8346
clock	50	8,094	2,786	220	2,516	10,880	74.3934	25.6066
	100	8,188	2,692	136	2,456	10,880	75.2574	24.7426
	150	8,199	2,681	125	2,406	10,880	75.3585	24.6415
	200	8,200	2,680	125	2,355	10,880	75.3676	24.6324
fifo	50	7,945	2,935	305	2,580	10,880	73.0239	26.9761
	100	8,121	2,759	174	2,485	10,880	74.6415	25.3585
	150	8,167	2,713	142	2,421	10,880	75.0643	24.9357
	200	8,175	2,705	138	2,367	10,880	75.1379	24.8621
lru	50	8,109	2,771	210	2,511	10,880	74.5312	25.4688
	100	8,196	2,684	129	2,455	10,880	75.3309	24.6691
	150	8,201	2,679	125	2,404	10,880	75.3768	24.6232
	200	8,201	2,679	125	2,354	10,880	75.3768	24.6232
opt	50	8,195	2,685	125	2,510	10,880	75.3217	24.6783
	100	8,229	2,651	50	2,501	10,880	75.6342	24.3658
	150	8,229	2,651	2	2,499	10,880	75.6342	24.3658
	200	8,229	2,651	2	2,499	10,880	75.6342	24.3658

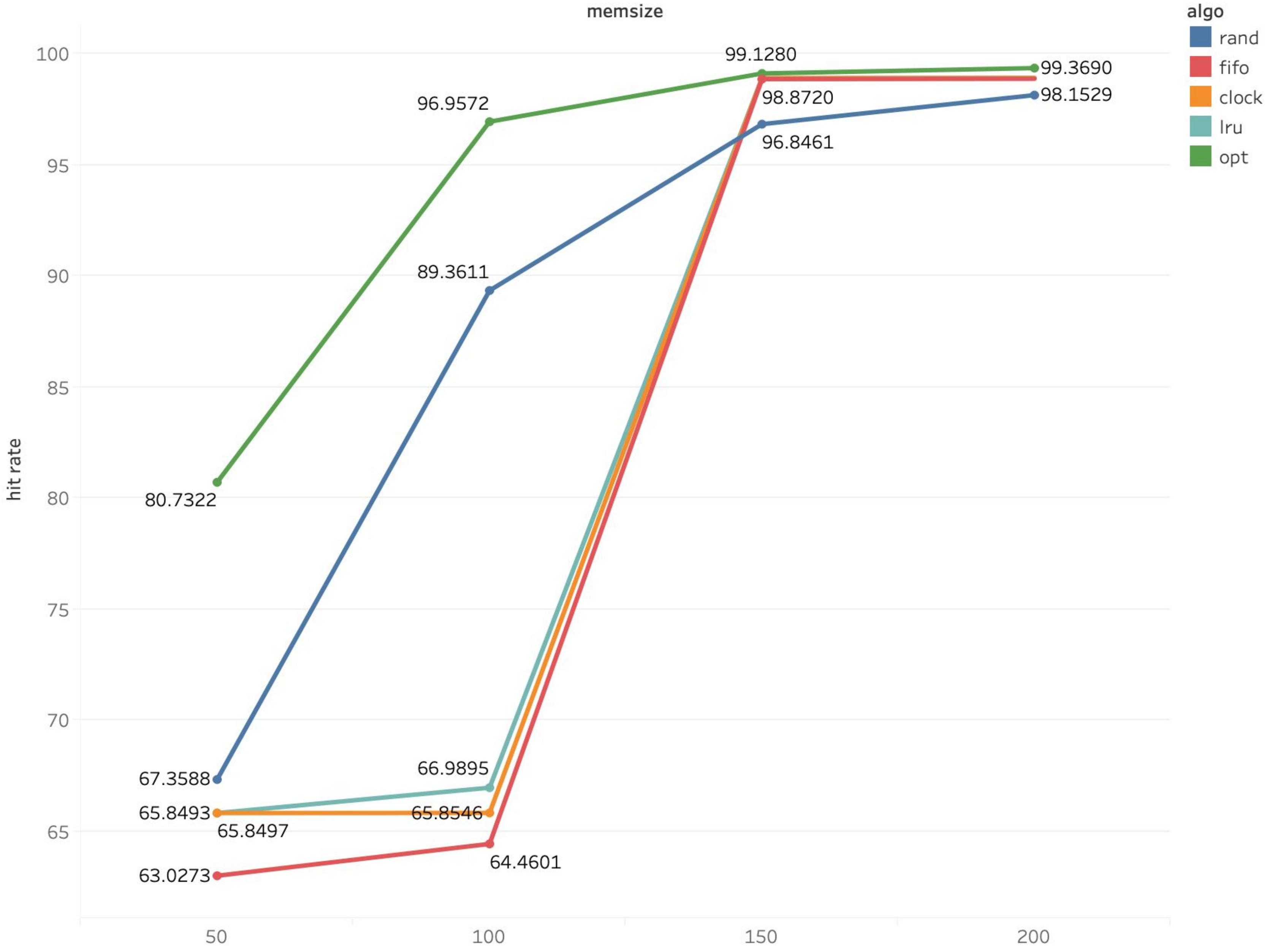
randomSum

algo	memsize	hit count	miss count	clean evictions	dirty evictions	total reference	hit rate	miss rate
rand	50	58,554	33,614	32,253	1,311	92,168	63.5296	36.4704
	100	65,973	26,195	25,507	588	92,168	71.5791	28.4209
	150	72,982	19,186	18,679	357	92,168	79.1837	20.8163
	200	80,643	11,525	11,141	184	92,168	87.4957	12.5043
clock	50	58,836	33,332	32,585	697	92,168	63.8356	36.1644
	100	65,922	26,246	25,877	269	92,168	71.5237	28.4763
	150	71,844	20,324	20,039	135	92,168	77.9490	22.0510
	200	75,021	17,147	16,834	113	92,168	81.3959	18.6041
fifo	50	56,821	35,347	33,932	1,365	92,168	61.6494	38.3506
	100	64,978	27,190	26,507	583	92,168	70.4995	29.5005
	150	72,144	20,024	19,550	324	92,168	78.2745	21.7255
	200	76,208	15,960	15,532	228	92,168	82.6838	17.3162
lru	50	58,821	33,347	32,606	691	92,168	63.8193	36.1807
	100	66,520	25,648	25,443	105	92,168	72.1726	27.8274
	150	71,762	20,406	20,154	102	92,168	77.8600	22.1400
	200	73,721	18,447	18,145	102	92,168	79.9855	20.0145
opt	50	68,812	23,356	23,287	19	92,168	74.6593	25.3407
	100	76,687	15,481	15,365	16	92,168	83.2035	16.7965
	150	82,490	9,678	9,511	17	92,168	89.4996	10.5004
	200	86,929	5,239	5,022	17	92,168	94.3158	5.6842

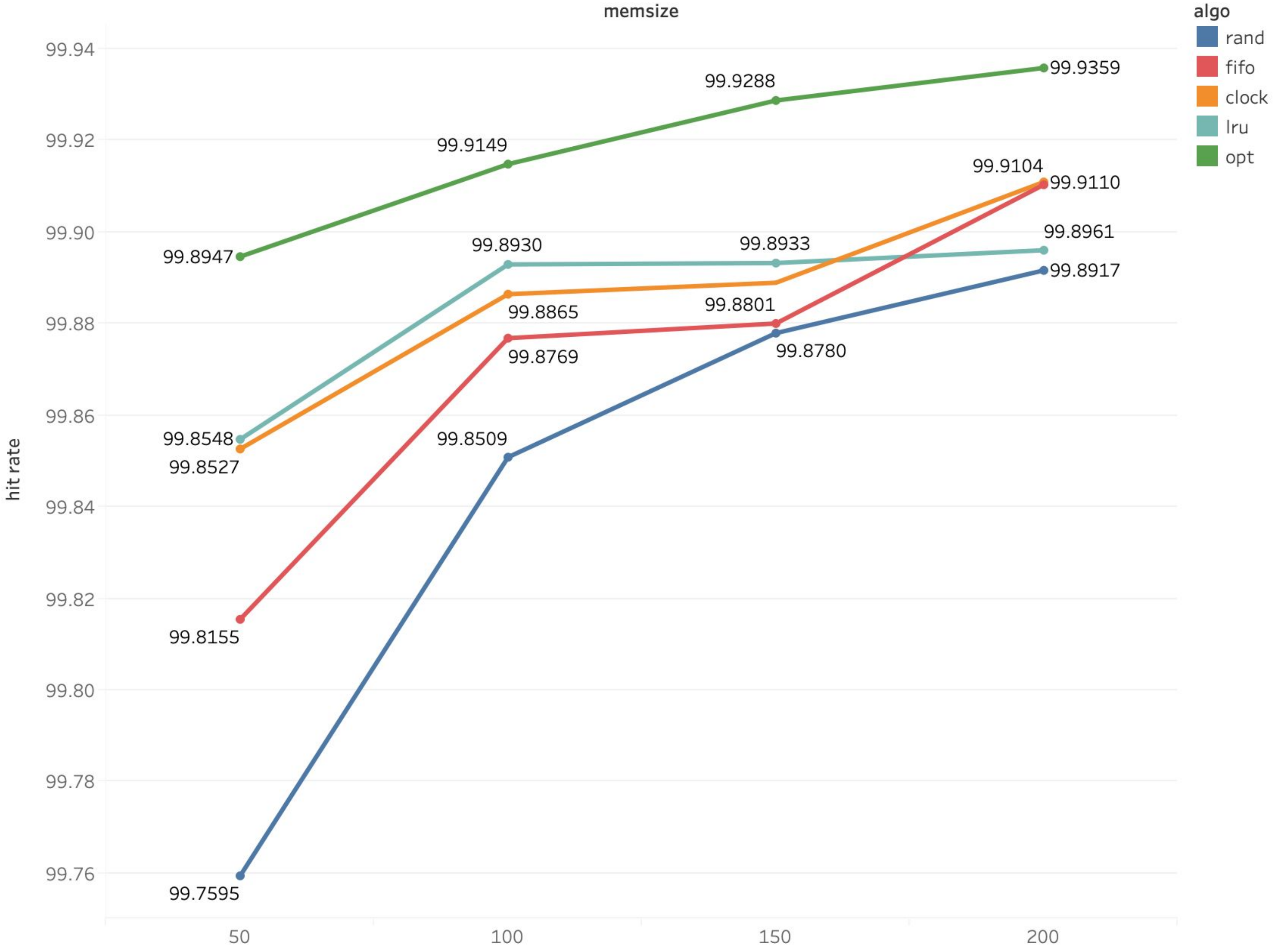
Data Visualizations

(ps: for reference)

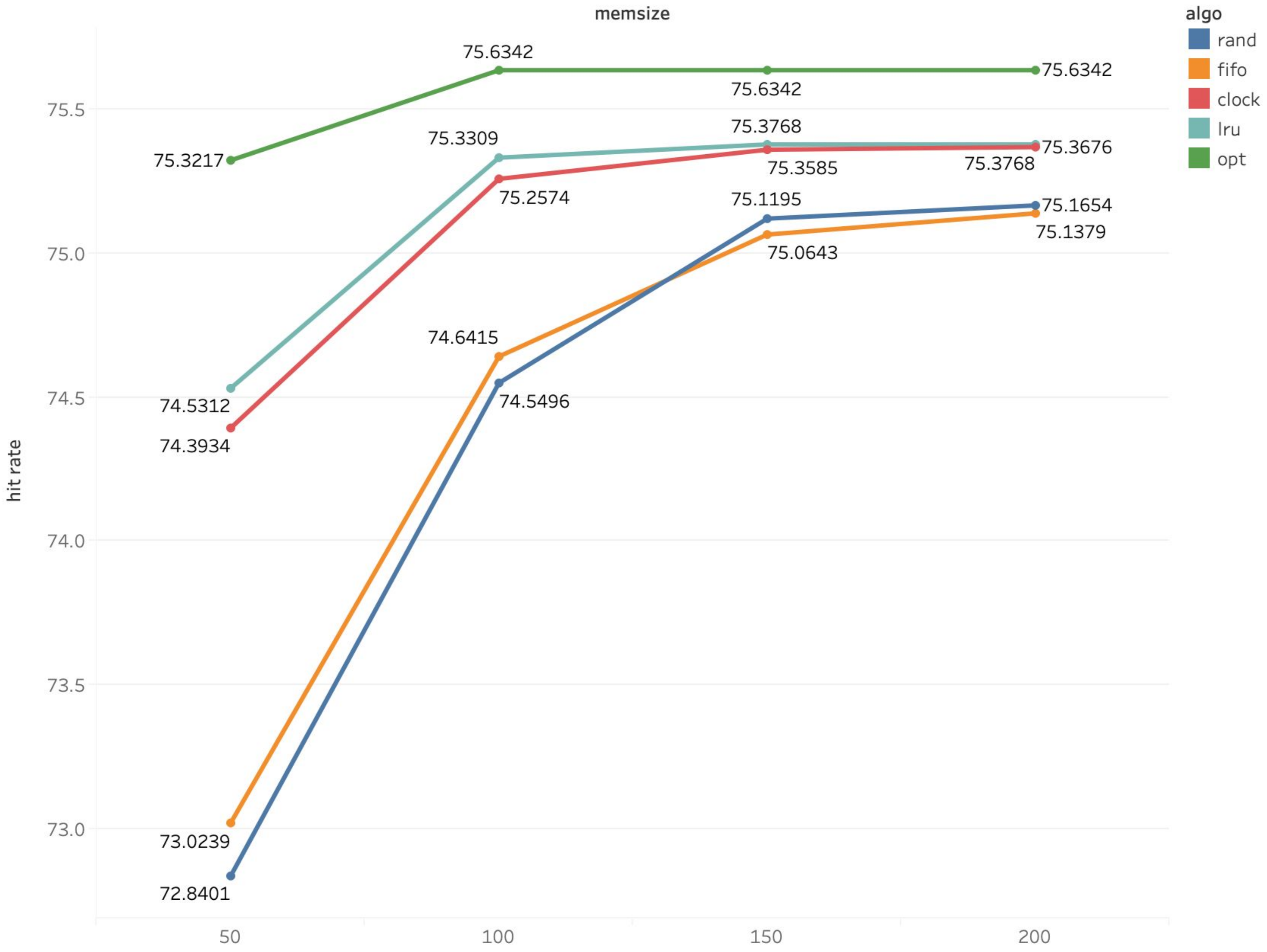
matmul


memsize
algo
rand
fifo
clock
lru
opt
hit rate
100
95
90
85
80
75
70
65
50
100
150
200
99.1280
99.3690
96.9572
98.8720
98.1529
96.8461
89.3611
80.7322
67.3588
66.9895
65.8493
65.8546
65.8497
63.0273
64.4601

blocked

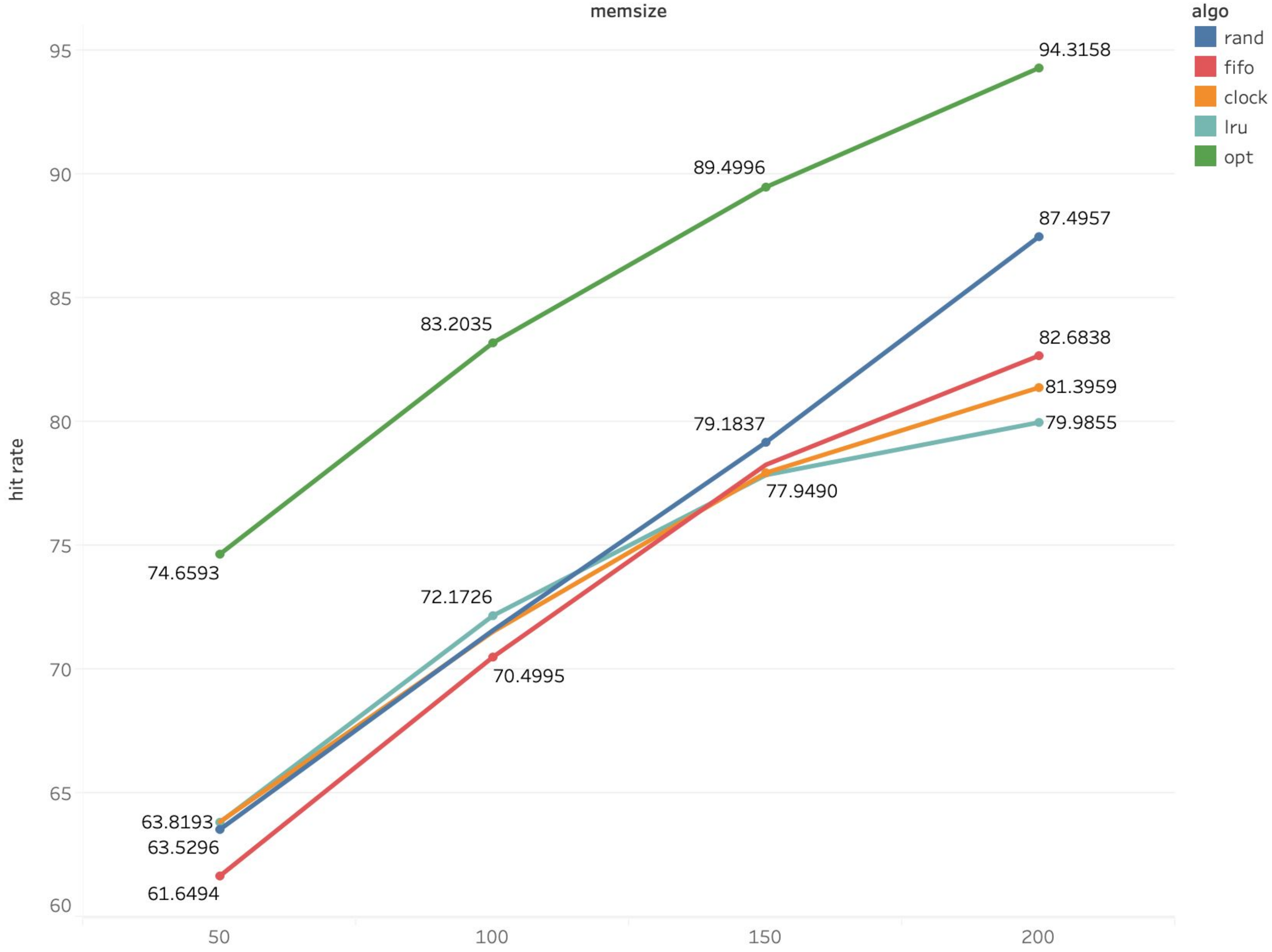

memsize
hit rate
99.94
99.92
99.90
99.88
99.86
99.84
99.82
99.80
99.78
99.76
50
100
150
200
algo
rand
fifo
clock
lru
opt
99.9359
99.9288
99.9149
99.9104
99.9110
99.8961
99.8947
99.8930
99.8933
99.8917
99.8865
99.8801
99.8769
99.8780
99.8548
99.8527
99.8509
99.8155
99.7595

simpleloop



memsize
algo
🟦 rand
🟧 fifo
🟥 clock
🟦 lru
🟩 opt
hit rate
75.5
75.0
74.5
74.0
73.5
73.0
50
100
150
200
75.6342
75.6342
75.6342
75.3217
75.3309
75.3768
75.3676
75.3585
75.3768
75.2574
75.1195
75.1654
75.1379
75.0643
74.6415
74.5496
74.5312
74.3934
73.0239
72.8401

randomSum


memsize
algo
rand
fifo
clock
lru
opt
hit rate
95
90
85
80
75
70
65
60
50
100
150
200
94.3158
89.4996
87.4957
83.2035
82.6838
81.3959
79.1837
79.9855
77.9490
74.6593
72.1726
70.4995
63.8193
63.5296
61.6494

Comparison Paragraph

For all programs, OPT has the best performance (hit rate) among all memory management algorithms in all different memory sizes.

In most cases, their performances will increase along with the increasing memsize, although they many have different increasing rate.

When memsize is large enough(can contain most vaddrs), algorithms may only have little effects on the overall performance, since there are few replacements will happen (eg: blocked).

FIFO, CLOCK, and LRU all tend to keep recently used pages when replacements happen. Thus, some time they will have similar performance (eg: matmul). But they are different in their ways of tracking the recently used pages. So the performance could be affected by their deviations (eg: simpleloop).

RAND is not always a worse algorithm. By adjusting memory referencing behavior and the size of referenced memory, we exposed the disadvantages of FIFO, CLOCK, and LRU. Hence, RAND becomes a better choice (eg: randomSum).

LRU Description

LRU: **latest recently used** algorithm. LRU will make decision based on the latest referenced time of each page in physical memory and evict the page with earliest used time when doing replacement. Here, we used a counter to track the referenced times of pages and keep updating them. The idea of LRU is trying to predict futures based on past experiences.

LRU is doing well when recently referenced pages will be re-referenced shortly. As in simpleloop program, some pages are saved in physical memory after the first iteration, and those pages that were just saved keep re-referenced in later iterations. From simpleloop graph, we can see the hit rate of LRU is closest to OPT.

LRU is doing poorly when memsize is relatively small and recently referenced pages will not be re-referenced soon. For example, a page address list {0,1,2,0,1,2} and the memsize is 2. Running LRU with this list will lead to 0 hit rate. Another example is the matmul program with memsize 50/100. We can see LRU performs even more poorly than RAND does.