Yang Lowe and Benny Roover
11/30/18
Comp 40 HW 7
Optimization Lab Notes

Description of benchmarks:

- small - midmark.um
- medium - advent.umz with partial solution as input
- large - sandmark.umz

Benchmark	Time (s)	Instructions	Rel to start	Rel to prev	Improvements
small	3.53	30.21 x 10^9	1.000	1.000	No improvements (starting point)
medium	30.81	-	1.000	1.000	
large	88.18	-	1.000	1.000	
small	2.76	26.41 x 10^9	0.782	0.782	Compiled with optimization turned on and linked against -lcii-01
medium	23.85	-	0.774	0.774	
large	68.19	-	0.773	0.773	
small	2.63	25.68 x 10^9	0.745	0.953	Compiled with optimization turned on and linked against -lcii-02
medium	22.81	-	0.740	0.956	
large	65.45	-	0.742	0.960	
small	1.93	13.00 x 10^9	0.547	0.734	Removed bitpack module, and coded bitpack_getu and bitpack_newu locally in -um.c
medium	16.18	-	0.525	0.709	
large	47.20	-	0.535	0.721	
small	1.90	11.96 x 10^9	0.538	0.984	Localized all functions in um.c into one large main()
medium	15.67	-	0.509	0.968	
large	47.36	-	0.537	1.003	
small	1.73	11.66 x 10^9	0.490	0.911	Inlined all ALU, IO, and memory_interface functions into um.c. Got rid of registers module
medium	14.27	-	0.463	0.911	
large	43.00	-	0.488	0.908	

small	1.21	8.28 x 10^9	0.343	0.699	Removed UArray in memory.c and replaced with a standard C array
medium	10.42	-	0.338	0.730	
large	29.39	-	0.333	0.683	
small	0.45	4.16 x 10^9	0.127	0.372	Combined memory and um into one file and one function in um.c. We essentially inlined every memory function
medium	2.92	-	0.095	0.280	
large	10.88	-	0.123	0.370	
small	0.22	2.63 x 10^9	0.062	0.489	Replaced the Hanson sequence that stored segment pointers with a dynamic C array and an expand function
medium	1.67	-	0.054	0.572	
large	5.62	-	0.064	0.516	
small	0.17	2.18 x 10^9	0.048	0.227	Replaced the Hanson stack that stored available ids with a dynamic C array (utilized like a stack) and an expand function
medium	1.37	-	0.044	0.820	
large	4.23	-	0.048	0.753	