Alexander Varshavsky - av653
Idan Levi - il177

Results - ASST2

Notes: To run project please type:
> make all
> make MP (or) make MT

You can also change search value in the multitest.h file at:
#define SEARCH_VALUE <value>
You can change the number of runs right above main() at:
#define MAX_RUNS <value>

Results (time in milliseconds):

Array Size	Section Size	# Threads/Procs	Threads	Procs
500	250	2	0.03932	0.34718
1000	250	4	0.09642	0.52458
5000	250	20	0.45294	1.99772
15000	250	60	1.31362	5.74182
25000	250	100	2.20338	9.94816
25000	200	125	4.77318	23.09594
25000	100	250	8.10708	46.03852
25000	50	500	11.37458	74.30792

Conclusions:
a general trend of: time vs. size of list to search for Processes as well as time vs. size of list to search for threads
Conclusion: Generally threads were much faster than procs on any level and size of array or number of threads/procs;

- a tradeoff point for Processes vs threads
It seems that no matter how small the array is, the threads still performed better since overhead of creation of new threads are much smaller than creating new process.

- a tradeoff point for parallelism for Processes and threads
If the array is significantly large and processing in a single thread will be much longer than overhead needed to create new threads then the multiprocessing might be more efficient. For small array sizes, single thread processing is much more efficient.