CS433 Assignment 2 Report

Ketan Chaturvedi (190428)
Tushar Singla (190918)

Part - 1: Locks

	1	2	4	8	16
OpenMP critical section	0.113850	1.158749	3.720373	11.081583	26.351934
POSIX mutex lock	0.156068	1.059807	2.173702	4.497067	12.565017
Lamport Bakery Lock	0.333491	4.208357	10.519756	28.873267	70.871623
Spin Lock	0.067867	1.419165	6.243244	13.922404	49.456356
Test & test & set lock (TTS)	0.067845	0.663809	2.673079	8.499138	15.499678
Ticket Lock	0.071490	1.852813	5.322573	15.970555	31.699822
Array Lock	0.112240	3.768978	7.579644	15.213697	25.769559
POSIX binary Semaphore	0.455650	1.163232	9.914395	23.717439	38.830129
Best	Spin lock/TTS perform slightly better	TTS Lock	POSIX mutex lock	POSIX mutex lock	POSIX mutex lock

* All the readings taken on 8-core Intel(R) Xeon(R) W-2145 CPU @ 3.70GHz
* All readings are in seconds

Results clearly proves that Lamport Bakery algorithm incurs lot of overhead because of its heavy software design. Its latency diverges quickly as threads are increased. Spinlock also shows similar divergence. In spinlock, certains threads might get thrashed or context-switched out and this might be the reason for its bad performance. TTS has lower latency as expected because of its efficient design. Ticket and array lock also does not diverge rapidly. POSIX mutex lock performs better than other designs in almost all the cases.

Part - 2: Barriers

Number of cores vs time in seconds

	1	2	4	8	16
POSIX barrier interface	0.364451	2.625084	5.175693	24.339093	46.635094
#pragma omp barrier	0.354920	0.419898	0.432157	0.860190	0.917570
Centralised sense-reversing barrier	0.034195	0.238667	1.216186	3.770793	9.252100
Tree barrier using busy-wait on flags	0.007879	0.273189	0.453021	0.643045	0.760385
Centralised barrier using POSIX condition variable	0.035193	3.196563	7.164096	39.487354	81.208000
Tree barrier using POSIX condition variable	0.009407	6.376943	26.364304	44.250863	77.322661
Best	Tree barrier using busy-wait on flags	Centralised sense-reversing barrier	#pragma omp barrier	Tree barrier using busy-wait on flags	Tree barrier using busy-wait on flags

* All the readings taken on 8-core Intel(R) Xeon(R) W-2145 CPU @ 3.70GHz

We observe that omp barriers and tree barriers using busy-wait on flags perform quite well even for 16 threads. Time taken by them is almost the same and they scale well. It seems that omp barrier is using a version tree barrier. We also see that the POSIX barrier interface and Centralised barrier using POSIX condition variables have similar run time. Centralised sense reversing barrier performs quite well initially but fails to scale. One thing we noticed was that barriers using POSIX condition variables took a very long time even for a small number of threads.