Universität Heidelberg   Prof. Dr. Robert Strzodka
Institut für Technische Informatik   Steffen Lammel

Advanced Parallel Computing
Term 2015 (Summer)

Exercise 5

- **Return electronically (MOODLE) until Tuesday, 26.05.2015 14:00**
- **Include name on the top sheet. Stitch several sheets together.**
- **A maximum of two students is allowed to work jointly on the exercises.**

5.1 Reading

Read the following paper and provide a review as explained in the first lecture (see slides):
- Shucai Xiao (Virginia Tech, US); Wu-chun Feng (Virginia Tech, US), Inter-Block GPU Communication via Fast Barrier Synchronization, IPDPS2010
- Torsten Hoefler, Torsten Mehlan, Frank Mietke and Wolfgang Rehm, Fast Barrier Synchronization for InfiniBand, CAC 2006, co-located with IPDPS 2006.

(25 points)

5.2 Barrier implementation and performance analysis

In this exercise, the goal is to develop a scalable barrier with minimal overhead. Refer to these references in order to implement a barrier which you consider to be optimal:

- John M. Mellor-Crummey, Michael L. Scott. Algorithms for Scalable Synchronization on Shared-Memory Multiprocessors. ACM Transactions on Computer Systems, 1991.
- Holger Fröning et al, Highly Scalable Barriers for Future High-Performance Computing Clusters HiPC 2011.

First, implement the barrier using a central counter that is incremented using atomic operations. Besides the PTHREAD barrier, this serves as a reference.

Then, implement a barrier that is optimized according to the two references above, in particular using tournament, dissemination and tree structures. After each optimization step measure the barrier performance. Thus, you will obtain results for three additional barrier implementations:
1. Tournament
2. Tournament and dissemination
3. Tournament, dissemination and tree structure

Write a test program in which each thread performs a certain number of barrier operations. Ensure that your barriers are working correctly, for instance by arbitrarily slowing down a single thread. Each thread should successfully pass the same number of barriers.

Conduct a performance assessment by comparing average barrier latency against the PTHREAD barrier implementation. Increase the number of threads participating in the barrier operation according to the following table. Report average barrier latency in the table and in a graphical representation. Perform a sufficient large number of barrier operations in order to obtain stable

results. Derive average barrier latency by dividing the overall execution time by the number of barrier operations performed.

Thread count	1. PTHREAD BARRIER Average barrier latency	2. COUNTER BARRIER Average barrier latency	3. OPTIMIZED BARRIER 1 Average barrier latency	4. OPTIMIZED BARRIER 2 Average barrier latency	5. OPTIMIZED BARRIER 3 Average barrier latency
1					
2					
4					
8					
12					
16					
24					
32					
40					
48					

Graphical representation:

(60 points)

Total: 85 points