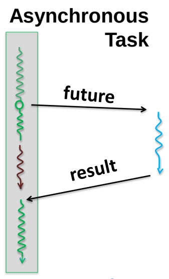
Performance Characterization of HPX - A Task-based Runtime System on the Xeon Phi™ Knights Landing

Patricia Grubel[1,2], Bryce Lelbach[2,3], Hartmut Kaiser[2,4]

[1]NMSU, Electrical and Computer Engineering, [2]STE||AR Group, [3]Berkeley Lab, [4]LSU Center for Computation and Technology

HPX – Task- based C++ Runtime



HPX is a general purpose C++ task-based runtime system that oversubscribes millions of asynchronous tasks onto a constrained number of physical threads or cores on multi-core, many-core, and heterogeneous systems. The goal of asynchronous task-based runtime systems is to ensure all processors are kept busy doing useful work. This is accomplished using futures where a value(s) required by one task, not immediately, can generate another to compute the required value(s) and can be scheduled to run on another core or hardware thread.

Intel® Xeon™ Phi Knights Landing (KNL)

- 64 “Silvermont” cores @1.4 GHz on single socket
- 4 hardware threads per core (1 per core for this study)
- 2 512 bit processing units
- 32KB L1 Cache Instruction & data each
- 1MB L2 Cache
- Memory -16 GB MCDRAM configured as L3 Cache,
  - 96 GB DDR4

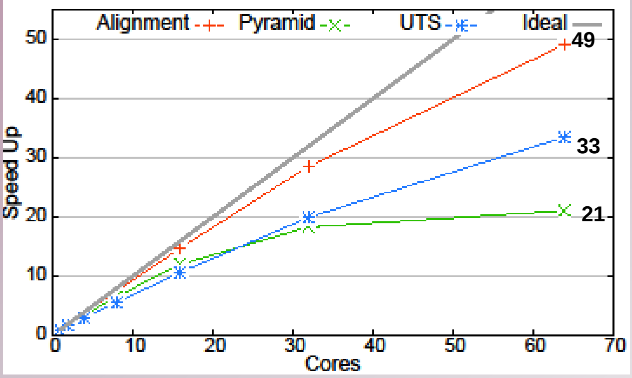
Note: This preliminary study uses 1 thread per core.

Overhead Categories

- Task Management Overheads
- Work Time Inflation

Task Management Overheads (TMO) include creation, deletion, and scheduling of tasks. On KNL these costs are on the order of microseconds per task and can be a significant percentage of execution time for fine grained tasks. UTS has the largest TMO with fine-grained tasks averaging 7.6 µsec. Alignment with coarse-grained tasks averaging 7.5 msec has the smallest TMO.

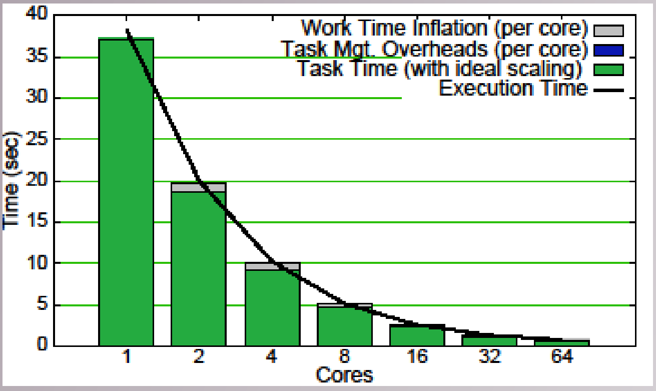
Work Time Inflation is the increase of the duration of the task caused by parallelization on the underlying hardware, and can be caused by cache misses, non-uniform memory and memory interconnect latencies, cache coherency, false sharing, and or memory bandwidth saturation. Pyramids and UTS have larger inflation of tasks due to data dependencies. Task management overheads and work time inflation are presented per core for comparison to parallel execution time.



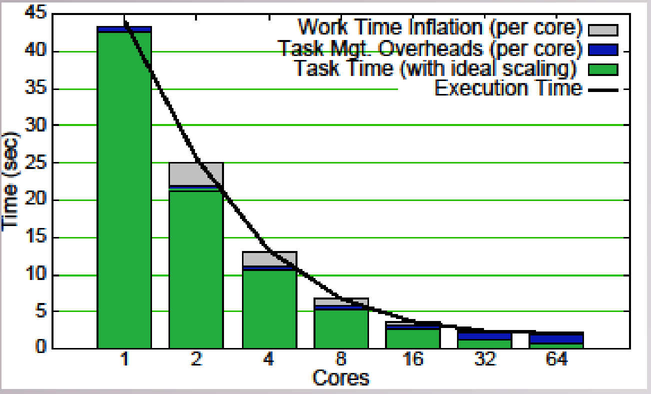
Speedup $= T_1 \div T_n$

Inncabs Benchmark Suite

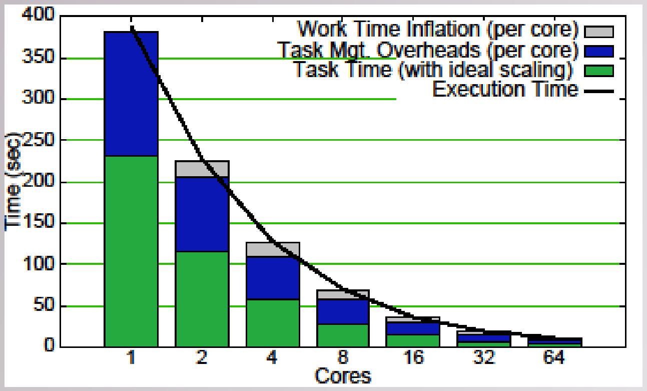
- C++ easily ported std::async → hpx::async
- Variety of Structures - Loop, Recursive or Mutex
- Variety of Task Granularity (Task Duration)



Alignment – aligns protein sequences, loop structure, coarse grain, average task 7.5 ms. Speedup 49 on 64 cores



Pyramids – 2D stencil solver, recursive balanced structure, median grain, average task 380 µs, overheads medium, Larger WTI due to data dependencies. Speedup is 21.



UTS – Unbalanced Tree Search, recursive unbalanced, fine grain, average task 7.6 µs, overheads large, speedup 33.

National Energy Research Scientific Computing Center

We appreciate the support from NERSC for use of the KNL early access white box nodes for these preliminary results. During this study NERSC was implementing Phase II of Cori, the newest supercomputer. Phase II brings on board 9304 KNL nodes with 68 cores adding 29.7 PFLOPS peak theoretical performance.

http://stellar-group.org