Towards Exascale Heterogeneous Computing

Challenges on Programming Models and Languages for Post-Petascale Computing FP3C

Project Overview

This task is a part of JST-ANR "Framework and Programming for Post Petascale Computing (FP3C)" project which is a collaboration between France and Japan. This project aims to contribute to establish software technologies, languages and programming models to explore extreme performance computing beyond petascale computing, on the road to exascale computing.

XcalableMP (XMP) http://www.xcalablemp.org/

- Directive-based language extension for scalable and performance-aware parallel programming
- It will provide a base parallel programming model and a compiler infrastructure to extend the base languages by directives.

Programming model and Language basic design

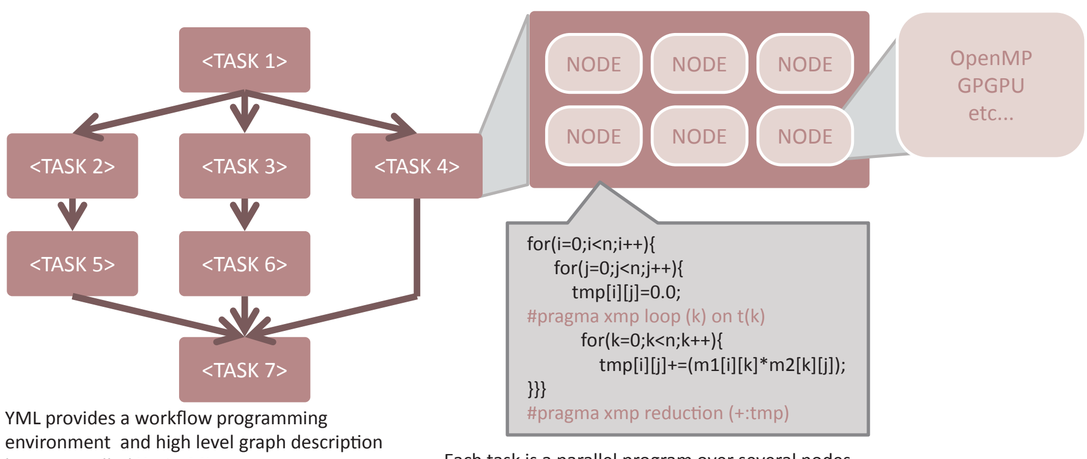
Two parallel programming models – XMP and YML – developed in Japan and France are combined to realize scalable parallel programming for post-petascale machines.

YML http://yml.prism.uvsq.fr/

- A workflow programming environment
- High level graph description language called YvetteML
- It allow programmers to use existing components in work-flow description.

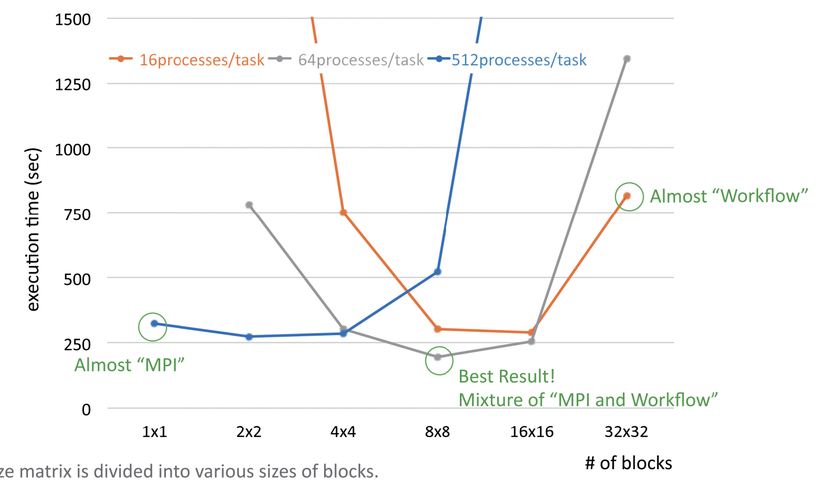
Multi-level hierarchical programming with XMP and YML

Incorporating parallel programs generated by XMP into tasks of YML workflow, a new multi-level hierarchical programming model will be developed.



YML provides a workflow programming environment and high level graph description language called YvetteML

Each task is a parallel program over several nodes.
XMP language can be used to descript parallel program easily!



A fixed size matrix is divided into various sizes of blocks.
Each block is calculated as a task.
While 512 processes are used in total, 16, 64 and 512 processes are used in each task.

Welcome Back, Systolic Arrays! -A Comparison of FPGAs, GPU, and CPUs -

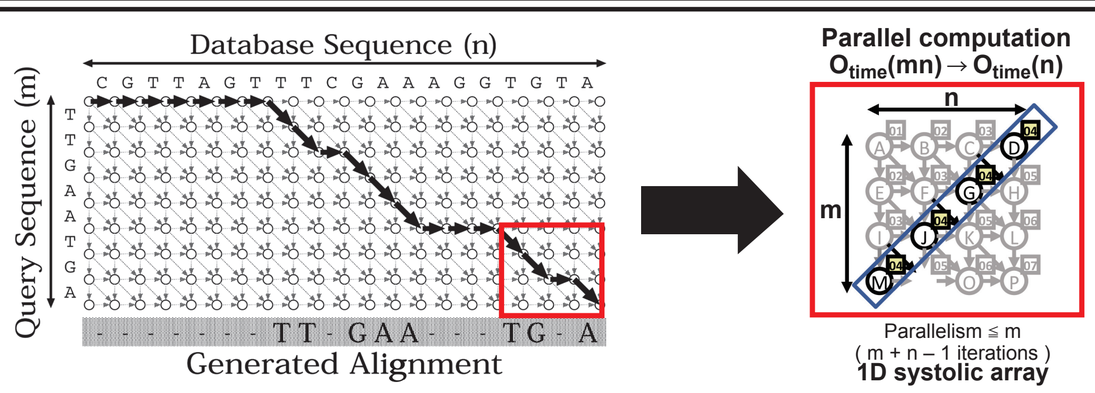
Project Overview

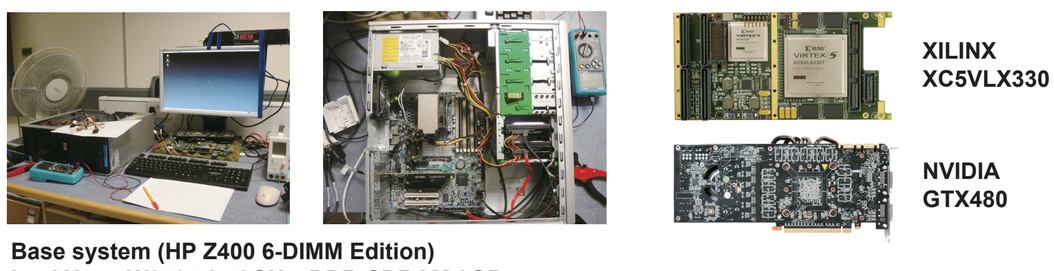
We have not found out any omnipotent architectures for general-purpose computing yet. To achieve highly efficient power performance, it is important to match the characteristics of application and device architectures. This project aims to discover the tradeoff point of power-performance among FPGAs, GPUs, and CPUs. This joint research has been continued by University of Tsukuba and Imperial College London.

Systolic Array

The structure of systolic arrays is complicated and it should be considered both in hardware and software. FPGAs have the advantage from this standpoint that they are flexible hardware but also software. The investigation is important at the first step toachieve exascale heterogeneous computing and will find out the tradeoff point toward the improvement of power performance.

Sample application (Smith-Waterman algorithm)





XILINX
XC5VLX330

NVIDIA
GTX480

Base system (HP Z400 6-DIMM Edition)
Intel Xeon W3505 2.53GHz, DDR-SDRAM 3GB
(*) Cooling fans and some components were disconnected for power measurement.

Performance Comparison in CPU, GPU, and FPGA

[measurement condition]
Smith-Waterman algorithm, Blosum50, Opening gap 12, Continuous gap 2,Query length: 8000, Database : 8000 x 1000Base system was used for all measurement condition in the following table.

		CPU	GPU	FPGA
Measurement Environment	Core Device	Intel Xeon W3505@2.53GHz		
	Acceleration Device	None	NVIDIA GTX480	XILINX XC5VLX330
	Program	SWPS3[1]	CUDASW[2]	Proposed
	Compiler	Intel ICC v11.1	NVCC 3.2-0.2.1221	XILINX ISE12.3
Measured Performance	Speed (GCUPs)	10	8	129
	Peak Power (W)	105	354	109
	Energy amount (J)	653	2,768	52
	Power-performance	4.2	1	53.2
Cf.) Previous Researches	Speed (GCUPs)	34.4[3]	16.1[4]	-
	Energy amount(J)	195(*A)	1,407 (*A)	-

[1] SWPS3 Project site: http://www.inf.ethz.ch/personal/sadam/swps3/
[2] CUDASW++ http://www.nvidia.com/object/swplusplus_on_tesla.html
[3] M. Aldinucci, et. al.: Efficient streaming applications on multi-core with FastFlow: the biosequence alignment test-bed, Proc. of Int'l Conf. on Parallel Comp., 273-280, 2009.
[4] Y. Liu, et.al.: CUDASW++: optimizing Smith-Waterman sequence database searches for CUDA-enabled graphics processing units, BMC Res., 2(1):73-82, 2009.
(*A) The references have no information about their power consumption.
Each value is estimated by using our experimental results.

CUPS: Cell Updates Per Second

$$CUPS = \frac{\#\_of\_Cells}{Total\_Time} = \#\_of\_Cells \times\_Freq.$$

Interim Result and Future Works

Although the computation is complex, its fine-grained nature shows promise for an efficient FPGA implementation. We will continue our quest to achieve further speedup and energy efficiency for designs targeting various applications based on systolic arrays, and to provide effective tools for the productive automation of such designs.