An overview of Parallel Programming Languages between 2000-2020

- By Aditya Singh Thakur, Vikhyat Dhamija, Pratik Mistry

Introduction:

- Parallel Programming Languages in last two decades (2000 - 2020)
- Features
- Evolution
- Future Trends

2000-2009:

ZPL (2004): *Z-level Programming Language*

- Uses array abstraction to implement a data parallel programming model for implicit parallelism
- After founded in 1993, latest version has Increased Portability (Windows, MacOS), Improved Performance and MPI-Specific

SequenceL (2007): *Originally invented in 1989*

- No explicit parallelism as compiler performs auto parallelism based on processing power available
- Advantages: Ease of programming, platform portability/optimization, code clarity and readability

Coarray Fortran i.e. CAF (2008): *Previously F-*

- Extension of Fortran created in 1990 that includes coarrays for parallel processing
- Program is interpreted as if it were replicated no. of times and executed asynchronously
- Support for latency hiding and avoidance
- Enhanced support for synchronization for fine-grain control over program execution

Chapel (2009):

- Supports data, task and nested parallelism with goal of increasing supercomputer productivity
- Based on Partitioned global address space (PGAS) parallel programming model

2009-2012:

OpenHMPP (2009):

- Based on set of compiler directives providing standard to handle hardware accelerators without the complexity associated with GPU programming
- Widely used by HPC actors in Oil & Gas,Energy,Manufacturing,Finance, Education & Research

ParaSail (2009): *Parallel Specification and Implementation Language*

- Supports both implicit and explicit parallelism where every expression is defined to have parallel evaluation semantics that compiler uses to make decisions on treating activities as parallel
- Uses a pointer-free programming model for objects to grow and shrink

Ateji PX (2010):

- Extension of Java to facilitate parallel computing on multi-core processors, GPU, Grid and Cloud
- Key offerings - data parallelism, task parallelism, recursive parallelism, speculative parallelism, the Actor model, data flow, stream computing

C++ AMP (2012): *Accelerated Massive Parallelism*

- Programs can compile and execute on data-parallel hardware like GPU's
- *restrict(amp)* feature can be applied on any function (including lambdas) to execute on C++ AMP
- Available as *<amp.h>* header file in the C++ concurrency namespace

2012-2014:

P-Amoeba (2012):

- Parallel implementation of AMOEBA in Java that utilize message passing library MPJ Express
- Derived from AMOEBA (distributed operating system) developed with goal to build a timesharing system in 1996
- Amoeba Interface Language used for creating stubs for accessing remote services

Cilk Plus (2013):

- Previous versions: Cilk (1994) and Cilk++ (2008)
- Key Features: Task parallelism (Fork–join model), Parallel loops using D&Q, Array notation (only in Cilk Plus) to apply operation on array parallely

Unified Parallel C (2013):

- Extension of C that uses a single program, multiple data (SPMD) model of computation
- Provides explicitly parallelism, shared address space, Synchronization primitives and a memory consistency model, Explicit communication primitives and Memory management primitives

Dryad (2014):

- Research project at Microsoft Research developed for data parallel applications and modelled as DAG's
- Parallelizes dataflow graph by distributing the computational vertices across execution engines

2014-2018:

Global Arrays (2014):

- It’s the library for parallel computing founded in 1994
- APIs for shared-memory programming on distributed-memory computers for multidimensional arrays

POSIX PThreads (2017):

- Developed in 1995 by IEEE for Unix with synchronization methods as Mutex (or Mutually exclusive locks), Read-write lock and Conditional variable (a test-and-set operation)
- Key Features: Very Lightweight and Efficient Communications/Data Exchange
- Latest version defines a standard OS interface and environment, command interpreter (shell), and common utility programs to support applications portability at the source code level

OpenACC (2018):

- Developed by Cray, CAPS, Nvidia and PGI for heterogeneous CPU/GPU Systems in 2012
- Capable of new controls over data movement, support for explicit function calls and separate compilation allowing the creation and reuse of libraries of accelerated code

2018-2019:

OpenMP (2018): *Open Multi-Processing*

- Developed in 2000, the API supports multi-platform shared-memory multiprocessing programming in C, C++, and Fortran
- Uses a portable, scalable model for a simple and flexible interface for developing parallel applications

X10 (2019):

- Designed specially for parallel computing using the partitioned global address space model in 2004
- Provides user-defined primitive struct types; globally distributed arrays, and structured and unstructured parallelism

Threading Building Blocks i.e. TBB (2019):

- C++ Library when used, a computation is broken down into tasks that can run in parallel
- Implements *work stealing* to balance a parallel workload across available processing cores in order to increase core utilization and scaling

2020 - Present:

Charm++ (2020):

- For enhancing programmer productivity (1980)
- High-level abstraction of parallel program and delivers good performance many hardware platforms
- Charm++ runtime system sends a message to the invoked object that reside on the local or remote processor in a parallel computation

RaftLib (2020):

- C++ library founded in 2014 that handles threading, memory allocation & placement and auto-parallelization of compute kernels
- Simple iostream-like operators can be used to assemble a massively parallel program

OpenCL (2020): *Found in 2009*

- Framework for program execution across heterogeneous platforms for task & data parallelism
- Latest version supports Shared virtual memory, Nested parallelism, Generic address space, Images, C11 , atomics, Pipes, Android installable client driver extension

ISPC (2020):

- C based compiler with extensions for single program, multiple data programming found in 2010
- Program appears serial but number of program instances execute in parallel on the hardware

2020 - Present:

Go (2020): *Founded in 2009*

- Built-in facilities and library support for writing concurrent programs using *goroutines*
- Apart from CPU parallelism, it also supports asynchronous programming for modern apps

Scala (2020): *Scalable + Language*

- Designed to grow with the demands of its users
- Built-in data-parallel programming support desired for Big data applications
- Also supports asynchronous programming, software transactional memory, and event streams, Java Concurrency API

CUDA (2020): *Compute Unified Device Architecture*

- Founded in 2007, its a software layer that gives direct access to the GPU's virtual instruction set and parallel computational elements
- Advantages: Scattered reads, unified virtual memory, shared and unified memory, faster downloads and readbacks to and from the GPU and full support for integer and bitwise operations

Future Trends:

- Increasing demands due to upward trends in AI and BigData
- Cheaper and faster processors and interconnections
- More freely available software tools
- Enhanced performance and speed-up every year
- Introduction of new languages due to Quantum Computing

References (1):

- https://www.tandfonline.com/doi/abs/10.1080/13658816.2011.645477?journalCode=tgis20
- https://en.wikipedia.org/wiki/Amoeba_(operating_system)
- https://computing.llnl.gov/tutorials/pthreads/#AppendixA
- https://jaxenter.com/ateji-px-an-extension-of-the-java-language-with-parallel-programming-primitives-101775.html
- https://www.slideshare.net/PatrickViry/ateji-px-manual
- https://en.wikipedia.org/wiki/ParaSail_(programming_language)
- https://en.wikipedia.org/wiki/ZPL_(programming_language)
- https://en.wikipedia.org/wiki/Charm%2B%2B
- https://en.wikipedia.org/wiki/Cilk#Cilk_Arts_and_Cilk++
- https://en.wikipedia.org/wiki/Coarray_Fortran
- https://en.wikipedia.org/wiki/Dryad_(programming)
- https://en.wikipedia.org/wiki/C%2B%2B_AMP
- https://en.wikipedia.org/wiki/OpenACC
- https://en.wikipedia.org/wiki/OpenCL
- https://en.wikipedia.org/wiki/OpenHMPP
- https://en.wikipedia.org/wiki/Global_Arrays
- https://en.wikipedia.org/wiki/CUDA
- https://en.wikipedia.org/wiki/Unified_Parallel_C

References (2):

- https://en.wikipedia.org/wiki/Threading_Building_Blocks
- https://www.ericsson.com/en/reports-and-papers/ericsson-technology-review/articles/the-future-of-cloud-computing
- https://readwrite.com/2019/01/29/computing-power-suppliers-are-shifting-gears-for-ai/
- https://en.wikipedia.org/wiki/X10_(programming_language)
- https://en.wikipedia.org/wiki/SequenceL#:~:text=SequenceL%20is%20a%20general%20purpose,and%20code%20clarity%20and%20readability.
- https://en.wikipedia.org/wiki/OpenMP
- https://en.wikipedia.org/wiki/X10_(programming_language)
- https://ispc.github.io/

Thank You