From C++11 Attributes to Parallel FastFlow Code

Automatic generation of efficient parallel code by using REPARA attributes and the FastFlow run-time

University of Pisa and Torino, Italy - REPARA FP7 EU-STREP project

What

The whole computer hardware industry embraced parallel algorithms, such as multicore, GPGPUs, and cloud. For these platforms, the extreme optimisation of sequential algorithms is no longer enough to squeeze the real machine power. Parallel Patterns provide a mechanism to express parallelism at a high level of abstraction and to make easier the transformation of existing legacy applications for parallel environments.

In the long term writing parallel programs ought to be as efficient, portable, and correct as it has been to write existing legacy applications for parallel environments.

How

FastFlow is a C++ parallel programming framework advocating high-level, pattern-based parallel programming. It greatly supports streaming and data parallelism, targeting heterogeneous platforms composed of clusters of shared-memory platforms, possibly equipped with computing accelerators. The FastFlow run-time efficiently supports fine grain parallelism via non-blocking multi-threading with lock-less synchronisations zero-copy messaging and asynchronous accelerator offloading.

Almost any x86-64, Arm, PPC system with a C++11 compiler is supported, e.g. Linux, MacOS, Windows. GPGPUs are also supported via either OpenCL or CUDA.

Open source software under LGPLv3

http://mc-fastflow.sourceforge.net/

Signal filtering

A high-performance filter for real-time video restoration through variational methods.

- Two-stage pipelined filtering: detection and correction
- Correction phase using variational method
- Successfully restoration of up to extreme levels of impulsive and gaussian noises
- Seamless portability to x86/64 CUDA Cxx GPGPUs

DNA alignment (Bowtie2 & BWA)

The Bowtie2 and BWA, which are among the fastest and most used alignment tools for genome analysis.

Bowtie2, BWA and several other parallel alignment tools exhibit the same concurrency exploitation pattern (pool of threads working on independent tasks).

- They can be all parallel to FastFlow by exploiting the task-farm pattern
- Only few lines of code changed
- Higher performance with decreased development effort

and much more ...

- High-performance deep packet inspection
- High-frequency streaming applications
- High-performance Haplotype assembly
- Parallel C4.5 classification
- Parallel InBzip
- Parallel Smith-Waterman
- Large graph building and analysis
- Video stereo matching
- Parallel Gillespie simulation
- Parallel Numerical Algorithm (Cholesky, LU)
- Parallel C++ memory allocator

Credits: M. Dunato, M. Toscani, M. Aldinucci
Univ. of Pisa and Torino, Italy