STREAM COMPUTING

Performance Engineers

Can your algorithms be made faster?

We make scientific code that performs

StreamComputing helps you port algorithms to GPUs to allow 2 to 10 times speed-up.

Which algorithms map best to GPUs and other vector-processors? What kind of algorithms are faster when using GPUs and other accelerators?

Explore this document and learn more.

StreamComputing is a well-known company in performance engineering (making software faster) and OpenCL (run software on GPUs). We provide **trainings** and **consultancy**.

Which algorithms can be used?

Each part has a description of the “computational dwarf”, examples of application areas and some words from the OpenCL perspective. **Notice that this text is a starting point, lots of things can still change.** You will notice a bit of an overlap, as some algorithms have aspects of two or more. This implies that some problems can have more than one solution.

Dense Linear Algebra

The classic vector and matrix operations, traditionally divided into Level 1 (vector/vector), Level 2 (matrix/vector), and Level 3 (matrix/matrix) operations. Applications are various.

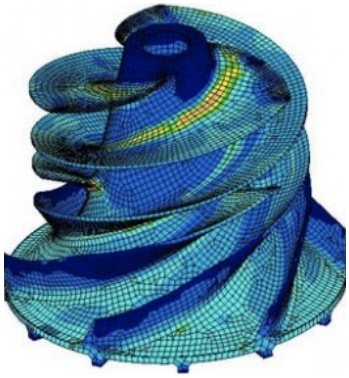
Applications:

- Linear algebra: LAPACK, ATLAS.
- Clustering algorithms / Data-mining: StreamCluster, K-means

Normally implemented by loops, and in many cases easy to parallelise in OpenCL.

Sparse Linear Algebra

Multiplication involving matrices composed primarily of zeroes. Computations can be done much more efficiently by moving the non-zero elements are around the diagonal of the matrix.



Applications:

- Finite element analysis
- Partial differential equations

Using OpenCL, there are two methods. Solve the problem with a series of operations, which results in a large overhead. The second method is using a series of successive approximations, minimising the error-function.

Spectral Methods

Solving certain differential equations, often involving the use of the Fast Fourier Transform. Spectral methods can be used to solve ordinary differential equations (ODEs), partial differential equations (PDEs) and eigenvalue problems involving differential equations.

Applications:

- Fluid Dynamics
- Quantum Mechanics
- Weather Prediction.

N-Body Methods

An N-body simulation is a one in which a dynamical system of particles, usually under the influence of physical forces, such as gravity. Computations can be done both ways (A influences B, and the same B influences A) and the state of the whole system is updated after each round. The basic algorithm is O(N^2). Optimisations for larger systems are possible by neighbour-administration and leaving far-away particles out of the computation – run-time approach-selection is desirable.

Applications:

- Astronomy: cosmology (e.g. formation of galaxies)
- Computational Chemistry: Molecular Dynamics (e.g. protein folding), Molecular modelling
- Physics: Fluid Dynamics, Plasma Physics

OpenCL-implementations can do tens of rounds per second with millions of particles, outperforming scalar implementations with ease.

Structured Grids

In a structured or regular grid all the elements have the same dimensions. Think squares and blocks.

Computations that depend on neighbours in an irregular grid.

Applications:

- Image Processing: Gaussian image blurring
- Physics Simulations: transient thermal differential equation solver
- Finite Element Method

In OpenCL the grids (of working-groups) are regular too, so mapping is quite easy. The problem to solve is how to do the communication between the between neighbours.

Unstructured Grids

All grids that are not regular grids. Different elements have a different number of neighbours.

This group has a lot of overlap with backtracking.

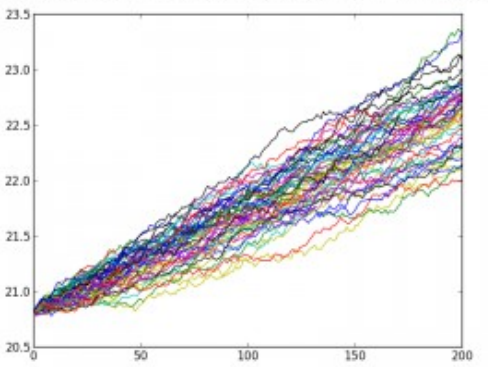
Applications:

- Computational fluid dynamics
- Belief propagation

The difficulty comes with the mapping of the irregular grid on the hardware.

Map-Reduce & Monte Carlo

Each process runs independently from the other, so nearly no communication is required between processes. In case of huge data-sets and compute-intensive algorithms GPUs can be used in combination with Big Data solutions like Hadoop.



Applications:

- Monte-Carlo: computation of pi, portfolio analysis, collision simulation, Sequence Alignment
- Distributed Searching

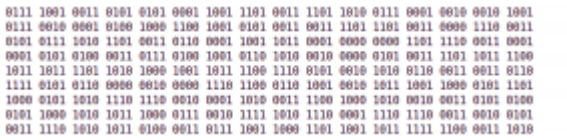
As the communication between the nodes are minimal, this is one of the fastest methods using GPUs.

Combinational Logic

These algorithms generally involve performing simple operations on very large amounts of data, which exploit bit-level operations.

Applications:

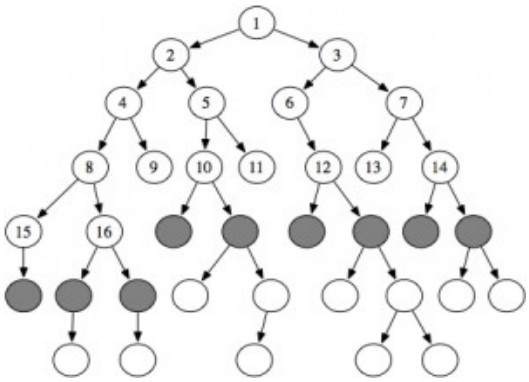
- Computing checksums, CRCs
- Encryption & Decryption
- Hashing
- Hamming weight



Not all hardware is fit for these types of operations. Device-selection is critical.

Graph Traversal

Graph traversal is the problem of visiting all the nodes in a graph in a particular manner, updating and/or checking their values along the way. Tree traversal is a special case of graph traversal. Has indirect lookups and little computation.



Applications:

- Search: depth-first search, breadth-first search, finding all nodes within one connected component
- Sorting: Quick-sort
- Serialisation/Deserialisation
- Maze generation
- Collision detection

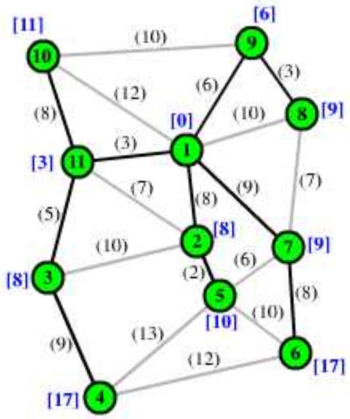
Dynamic Programming

This is an algorithmic technique that computes solutions by solving simpler overlapping subproblems. Many dynamic programming problems operate by filling in a grid that represents the solution space, where one location in the grid holds the final answer.

Applications:

- Graph problems: Floyd's AllPairs, shortest path, Bellman-Ford algorithm
- Sequence alignment: Needleman-Wunsch, Smith-Waterman

As "dynamic" applies, better performance is reached when tuned during run-time.



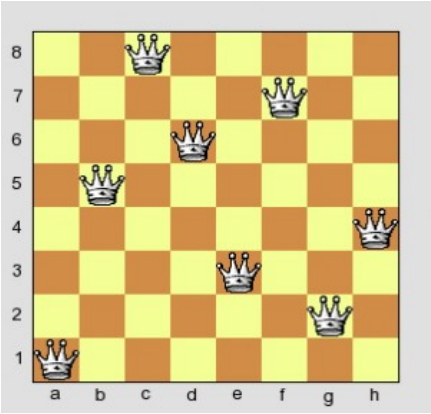
Backtracking

Building up all possible solutions and eliminating invalid solutions, most times in one step, as there is no overview of all possible solutions at the beginning. It is effectively a depth-first search of a problem space and therefore a special case of dynamic programming, but with a strong accent on the step-wise creation of the solution-space. The generic solution for this group of algorithms is branch-and-bound (divide and conquer).

Applications:

- Puzzles: N-queens, crosswords, verbal arithmetic, Sudoku, Peg Solitaire.
- Travelling salesman
- Knapsack, subset sum, and partition problems
- Integer Linear Programming
- Boolean Satisfiability
- Combinatorial Optimisation

If the problem-space is expected to be equally divided, a limited number of starting positions is created to walk the problem space in parallel.



Probabilistic Graphical Models

A graph that combines uncertainty (probabilities) and logical structure (independence constraints) to compactly represent complex, real-world phenomena.

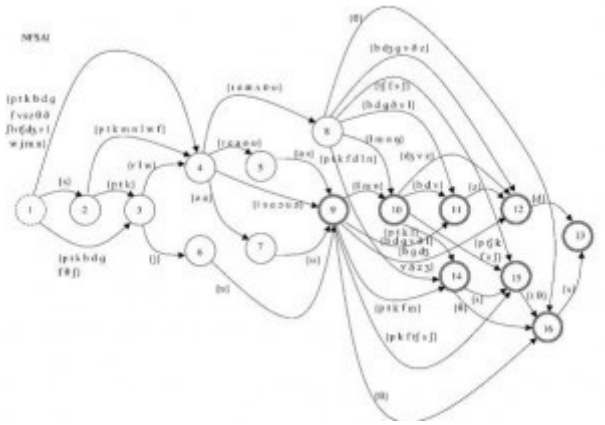
Applications:

- Bayesian networks: belief networks, probabilistic networks, causal network, knowledge maps
- Hidden Markov models
- Neural networks

As more processes need to update the same node (a typical case for atomics), a lot of time is consumed here.

Finite State Machines

A mathematical model of computation used to design both computer programs and sequential logic circuits. It is conceived as an abstract machine that can be in one of a finite number of states.



Applications:

- Video Decoding, Parsing, Compression
- Data Mining
- Find Repeating Patterns

What can this do for you?

These are only some examples in which GPUs can make a huge difference. Speed-ups of 2 to 10 times seem to be the norm, but combined with other software performance engineering techniques, 100 to 1000 times are possible.

StreamComputing has the expertise and know-how to help you get these speed-ups in scientific code.

Goodbye hours! Hello seconds!

If you want to request a demo to be tailored for you, call us to plan a meeting at your best convenience.

STREAM COMPUTING
Performance Engineers

Your contact:	V.G.Hindriksen
Phone :	+31 854865760
Cell :	+31 6 45400456
E-mail :	vincent@streamcomputing.eu
Newsletter :	http://bit.ly/OpenCLnewsletter
Twitter :	http://twitter.com/StreamComputing
Blog :	http://www.streamcomputing.eu/blog