Harnessing the Driving Force of Dependencies

Eva Burrows

Bitonic Sort DDA

The data dependency of the bitonic sort is a combination of several Butterfly DDAs of increasing height.

Bitonic sort for 32 elements:

```
\[ \text{Bitonic Sort DDA Code} \]
```

What is a DDA?

Data Dependency Algebras (DDAs):

Data dependencies are inherent in the notion of algorithm. DDAs turn dependencies into explicit, programable entities. Hence DDAs are more fundamental than general dependency analysis of, e.g., loops in an imperative program. They abstract how parts of a computation depend on data supplied by other parts at a fine-grained level.

Code Example to the Right:

DDA concept (API) definition in Magnolia, a syntax interface with axioms describing its properties.

Hardware Space-Time DDAs:

A hardware architecture's space-time communication layout can be defined as a special DDA. It has projections into space and time, where space comprises the hardware's static spatial connectivity pattern.

Mapping Computations to Hardware:

Finding an embedding of the computation's DDA into the space-time DDA of the hardware.

Flexible Spatial Placements of Computations

DDA Projections:

Define placements of computations on processors in a network or on a multi-core, or on an FPGA. DDAs can also serve as a graphical aid when mapping computations to highly parallel chips.

Example: Butterfly DDA

Appears in many divide & conquer algorithms, FFT, etc. The figures below show various placement patterns for a 32 node butterfly in space-time, as controlled by the space and time projections. The figures are generated from the Butterfly DDA.

This work is carried out under the supervision of Prof. Magne Haveraaen at Bergen Language Design Laboratory, University of Bergen, Norway. The Magnolia compiler is implemented by Anya Helene Bagge. (BLDL -- http://bldl.ii.uib.no)