Embedding Parallel Computation in a Stochastic Mesh Network: A Morphogenetic Approach

Max Orhai
Portland State University

Follow this and additional works at: https://pdxscholar.library.pdx.edu/anthos

Part of the Computer Sciences Commons

Let us know how access to this document benefits you.

Recommended Citation
https://doi.org/10.15760/anthos.2011.22

This Article is brought to you for free and open access. It has been accepted for inclusion in Anthós by an authorized administrator of PDXScholar. For more information, please contact pdxscholar@pdx.edu.
Embedding parallel computation in a stochastic mesh network: a morphogenetic approach

Each node has a spatial neighborhood which determines which other nodes it may communicate with. The size of these neighborhoods determines the amount of connectivity in the network. Nodes are tiny computers all running the same simple program, with just enough memory to hold two numbers.

The sequence of images below shows the algorithm in action. As the integers from 1 to 100 in random order are injected into the system at an arbitrarily chosen node, a dynamic linear linkage active data structure is grown which sorts the numbers in parallel as a length-conserving and deadlock-avoiding path is found. To avoid this problem, the network size should be larger than the input data, with sufficient connectivity.

What can go wrong?
In too-sparse or locally over-crowded networks it is possible for the linkage to grow into a cul-de-sac. Then the buffers of the nodes in the linkage fill up with partially-sorted numbers, which may later be released with the addition of new nodes to the network.

Potential future work:
What other kinds of programs can be realized as parallel dataflow graphs in physically realistic space?

How can these structures be made to grow, adapt, and heal themselves if disrupted?

How might the system be affected by the unforeseen constraints of an implementation technology?

The chart above shows the performance of the parallel insertion sort. In a sequential computer, this algorithm takes time proportional to the square of the input size. By distributing the work spatially, we are able to complete the task in linear time!

Time steps to sort a maximally disordered sequence

References and related work:
