“Ball Computer” or How I Learned to Stop Worrying and Love Computing with 3D Wireless Grids

Amir Mansoor Kamali
Department of Computer Science, University of York, York, UK
amir@cs.york.ac.uk

Christopher Crispin-Bailey
Department of Computer Science, University of York, York, UK
chrisb@cs.york.ac.uk

Jim Austin
Department of Computer Science, University of York, York, UK
jim.austin@york.ac.uk

Extended Abstract

On October 2009 a project started in the Advanced Computer Architecture Group, Department of Computer Science, University of York to determine to what extent wireless connections can be used in massively parallel computers. This was mainly motivated by the increasing volume of research on small high-speed short range wireless devices, especially on-chip radio devices.

As the first step three categories of wireless technology were surveyed to spot best candidates for a wireless communication scheme for a parallel platform. A desired wireless device for this project is a technology capable of sending data wirelessly over a 1-2 cm link with data rates around tens of GB/s, consuming reasonably low power (ideally comparable with wire technologies) while occupying a fraction of square centimetre of chip area.

The strengths and weaknesses of available capacitive coupling, inductive coupling and on-chip radio devices were surveyed. The survey show that although coupling technologies (both capacitive and inductive) are very good in terms of energy consumption, data rate and occupied area but both suffer from the very short communication distances they support. On-chip radio devices support cm-range links, the state-of-the-art on-chip radios are capable of transferring tens of GB/s. The newly emerged Tera-Hertz radio modules has brought the hope that in near future much higher data rates can be delivered by these devices. The intrinsic weakness of radio devices is their high power consumption. The survey shows that radio devices are improving in terms of power consumption but still better technologies are needed to compete with wireline devices. The survey covers more than 140 research projects during the last decade and the results can be found in more details in the paper our group has presented in MoWNeT13 conference (1).

To evaluate the performance of a wireless interconnect network in a parallel platform a new concept machine is introduced which is named *Ball Computer*. The Ball Computer is a 3D multi-channel wireless hexagonal network in which each node has a processor and a number of wireless transceivers. The position of the nodes are fixed at the present plan. The whole circuitry of a node is sought to be small enough to be placed inside a plastic (or any other suitable substance) ball. Like all wireless modules the balls can be manufactured very cheaply especially when produced in large volumes.

Radio modules are tuned to different channels to avoid packet collision between simultaneous data transactions. The choice of channels is the subject of a two-step network partitioning algorithm designed and introduced particularly for this platform.

The first step of the network partitioning algorithm is called *zoning*. In the zoning sub-algorithm the whole network is partitioned into overlapping zones to make sure that no hidden node can exist in any zone. This way all zone members can sense all data transactions to and from any other zone members. The zones are sought to have overlaps to support nodes with redundant paths to increase the robustness of the network.

In the second step (*channel assignment* sub-algorithm) different frequency channels are assigned to zones in order to avoid interference between transactions in two or more zones. While zoning guarantees no packet collision inside a zone, channel assignment does not let different zones operate on the same channel and therefore risk a packet collision.

There are a number of challenges that are left for next stages of the project. Among them are power delivery and heat dissipation. This does not mean that they are trivial matters; in contrast, the authors of this manuscript appreciate the importance of these issues. They are important challenges on their rights. But regarding the period of time available, the group decided to focus on a number of issues and left the others for the next stage. It should be noticed that the details of the proposed network are prone to change when tackling the power delivery or heat dissipation challenges.

At this stage of project the idea of ball computer is implemented in a simulation environment. For simplicity of analysis the network is simulated in an ideal world in which many real-world sources of noise are not simulated. They are just modelled by a degree of randomness in packet transmission times. They will be analysed more accurately on next stages. A set of simulation and data visualisation tools is designed and implemented which can be used for next stages of this project as well as other researchers interested in the same fields. The reason why one of the existing network simulators was not used was that the group was not sure if any of those simulators operate on the exact level of abstraction needed in this project.

The high level of abstraction of the simulator dictates changes in the tasks used for evaluating the network. Real-world tasks have too much trivial task-related details that have no use in our level of analysis. Tasks should be simplified and made ready to be used in the simulator. Through this conversion process something is made that is called a *task-model*. A task-model is nothing more than an artificial traffic generator which mimics the I/O behaviour of one of a group of tasks in real world but at the same time has a certain degree of stochastic nature. The size and the generation pattern of packets are inherited from the real task by the task-model. But the contents of the packet and the type of operation applied to it are not known in a task-model.

Two task-models were designed and tested in this project. They differ from each other in the sense that they have different levels of task dependency.

The first task-model is called simple parallel task-model in which a node sends a number of packets to some of its neighbours. The neighbours which are chosen randomly, process the packet in a time proportionate to the packet size and then sends it back to the original node. The task finishes when all nodes receive all the results from their neighbours. The operation of the nodes in this task-model depend only on the operation of their direct neighbours. In other words, there is just a very geometrically local dependency between nodes. Weather forecasting is an example of such a task.

The other task-model which is inspired by lots of divide-and-conquer mathematical transformations is called FFT task-model. Inspired by(but not restricted to) real-world FFT, the FFT task-model recursively splits its workload into pieces, finds idle neighbours and sends the data to them for parallel execution of the task. This repeats until the data size is small enough for a local execution of FFT task which has a logarithmic execution time. It is just to pay tribute to the original FFT but our analysis shows that the order of execution time does not hugely affect the network performance. The completion of the original FFT task depends on many nodes possibly far beyond its immediate neighbours. In fact a large tree of dependencies is created every time this task-model is executed.

A revised version of the FFT task-model is also implemented to accommodate a load balancing mechanism. This is due to the observation made during the experiments. The original FFT task-model divides the workload into equal pieces and assumes that the receivers of data have almost equal chances to find other nodes to share the task with them. But this does not always happen in real world. The topology and the geometry of the network dramatically changes the chances of the receivers to find new candidates. To compensate for this imbalance the task-model first scans the network to get an overall idea about the dependency tree (which is imbalanced in almost all cases) and then works out the share of each neighbour based on the size of the sub-tree it represents.

The performance of the network is measured in form of speed-up and utility factor as expressed by Equation 1 and Equation 2:

Equation 1 $$\boldsymbol{Speed-up} = \frac{\boldsymbol{total\ compute\ time}}{\boldsymbol{execution\ time}}$$

Equation 2 $$\boldsymbol{Utility}\ (\boldsymbol{N}) = \frac{\boldsymbol{Speed-up}}{\boldsymbol{N}}$$

All times are calculated in terms of simulation iterations. In other words, the quantum of simulated time in this project is one iteration of the simulator.

The results show that the partitioning algorithm has successfully eradicated the packet collision problem. Not even a single instance of packet collision is detected during thousands of experiments made in this project. It should be reminded that the simulator is an ideal network in which some sources of noises in the real world is yet to be simulated.

The results show that all task-models have better performances when they deal with larger data sizes. Also it shows that the performance of FFT task-models improves by using multi-tasking processors. Multi-tasking nodes can effectively recruit the nodes that are waiting for results from other nodes.

The overall behaviour of the network is studied by changing the data rate, data size, network size and the computational ability of nodes. The results show that the performance of the FFT task-model depends on the network size while that of Simple Parallel task-model does not change. The performance of both task-models improve when the link data rates increase and also when the computational capacity decreases. Another exciting result is that the ratio of data rate divided by computational ability is the most important factor in determining the performance of the network with both task-models. Another important result is that the performance of the FFT task-model does not always increase with an increase to the network size. It means that there is an optimum network size beyond which the performance of the FFT task-model stops and then decreases. For this reason an FFT task-model should put a cap on the number of nodes it recruits for its execution to achieve its peak performance.

As a conclusion, this project showed that there are opportunities in replacing wires with wireless connections. However, wireless devices still suffer from their high power consumption despite their recent improvements.

This project shows that an energy-efficient wireless parallel computer cannot be made right now as there are still serious challenges to tackle on this field. Instead, this project suggests that it is the right time to start thinking about such a platform because radio devices are rapidly improving in all aspects. The first wireless parallel computer is not far from us. We tried to work out how it may look like and how good it may work but its real features and capabilities are yet to be determined. We keep working on this interesting topic.

Bibliography

1. *On Advantages and Limitations of 3D wireless Grids as Parallel Platforms.* **Kamali, Amir Mansoor, Crispin-Bailey, Christopher and Austin, Jim.** Montreal : IEEE, 2013. International Conference on Selected Topics in Mobile and Wireless Networking, MoWNeT'2013. pp. 48-55.