Studying the Effects of Naive Routing on Packet Latency in SpiNNaker's Heterogeneous Interconnection Network

Jonathan Heathcote

Abstract

The SpiNNaker project aims to produce a massively parallel computer containing one million low-power processor cores for running neural simulations in real-time. The system consists of up to 1,200 circuit boards each hosting 48 18-core SpiNNaker chips. The boards are connected via high-speed serial links which multiplex multiple chip-to-chip connections across board boundaries. A model of SpiNNaker was built to asses the effects of these high-speed links on data transmission latency. Our results show that an 80% overhead is introduced by the links caused in part by the naive routing scheme for which some improvements are suggested.

1 Introduction

The SpiNNaker system is a machine designed for emulating the behaviour of the brain using a novel, massively-parallel computer architecture. The system will ultimately be made up of 1,036,800 low-power ARM processors much like those found in mobile phones, running a real-time one billion neuron brain model.

The system is made up of 18-core SpiNNaker chips connected together via a 16-wire interconnect known as '2-of-7' links. 1,200 circuit boards containing 48 SpiNNaker chips each are combined to form the complete system. Because the links require so many wires it is impractical to use them to connect boards together. Instead, the links are multiplexed onto a small number of high-speed, 4-wire links.

Until recently, prototype systems consisted of only a single board (using only '2-of-7' links). New, larger systems using the multiplexed links between boards continue to use naive 'dimension-order routing' to route packets of data between chips which falsely assumes all links are equal.

Due to the real-time nature of brain simulation, the latency of packets sent between chips must be kept low. A software simulation of SpiNNaker and the two types of interconnect was built in order to study the effects of the naive routing scheme on latency. This paper describes the design of the simulator and describes our findings.

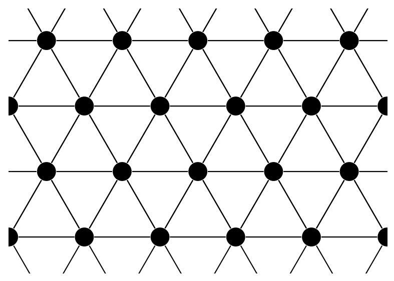


Figure 1: SpiNNaker chips (dots) and their links (lines).

2 SpiNNaker Topology

Before describing the simulation it is helpful to understand the topology and interconnection of a SpiNNaker system in greater detail.

The smallest unit in the system is the SpiNNaker chip consisting of 18 low-power ARM processors, a router and some memory [4]. Chips are arranged in a mesh with each chip connected to six of its neighbours to the as shown in figure 1 using 16-wire '2-of-7' connections.

The top and bottom edges of the mesh are joined together forming a cylinder whose ends are then joined together to form a torus (doughnut) known as a toroidal mesh.

Packets are routed from chip to chip using dimension order routing. The mesh can be considered a three-dimensional[1] space as shown in

[1]The dimensions are non-orthogonal which is the cause of some of the slightly unintuitive properties of dimension order routing in SpiNNaker.

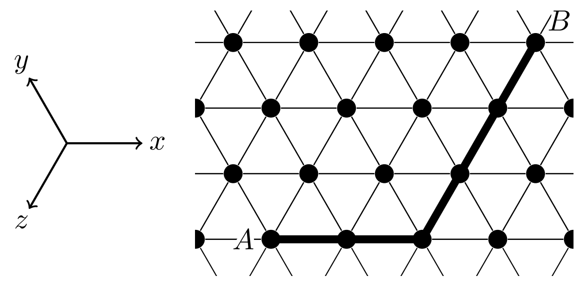


Figure 2: Dimension order routing's path between A and B on a hexagonal mesh.

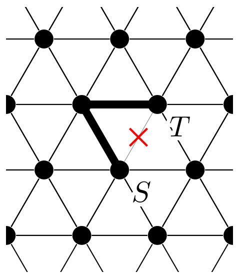


Figure 3: Emergency route from S to T when the normal route is unavailable.

figure 2. Packets travel along each dimension in turn until they can get no closer at which point they move onto the next dimension. Such paths can be trivially computed as shown in [2].

If a packet's path is blocked for a certain amount of time, for example due to a bad link, 'emergency routing' via an adjacent chip is attempted as shown in figure 3. This novel approach allows the system to automatically avoid link failures and congestion.

Boards containing 48 SpiNNaker chips are logically arranged into a hexagon as shown in figure 4. The small hexagons represent a single SpiNNaker chip and touching hexagons are connected via '2-of-7' links. The exposed connections on the six sides of the large hexagon are multiplexed onto six high-speed serial links to be connected to other boards using only 24 wires rather than the 768 wires required for all 48 '2-of-7' links. This allows an arbitrarily large, continuous mesh to be constructed using multiple boards with manageable wiring requirements.

Each high-speed serial connection is able to transfer data at up to 3.2 Gb/s [5]. This is greater than the eight '2-of-7' links which have a total maximum of 2.0 Gb/s, as a result, there

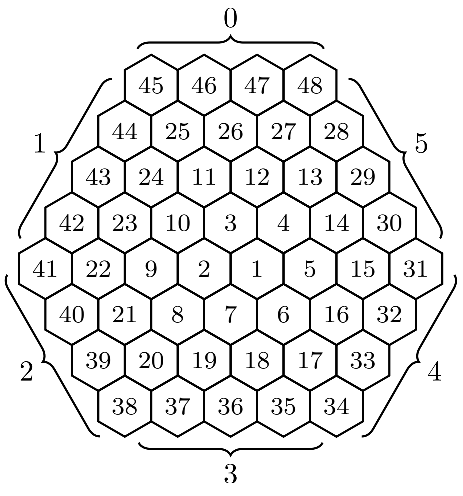


Figure 4: The logical arrangement of SpiNNaker chips on a board.

is no bottleneck in bandwidth. Unfortunately there is a latency cost when the links are multiplexed, transmitted and demultiplexed.

3 Simulator

A simplified software model of the SpiNNaker interconnection system was simulated to measure the effects of the serial links. The model aims to extend the model used by Navaridas et al. [3] to test the suitability of the toroidal mesh topology for neural simulation. As then, a relatively crude model is used to keep the simulation run-time manageable.

3.1 Modelling Simplifications

The model reduces a SpiNNaker chip (and its software) to a simple traffic generator and router. These simplified chips are then collected into boards and the boards into complete systems.

3.1.1 Traffic Generation

Packets are emitted by the traffic generators at random intervals destined for a single random chip in the system. This simplification, also made by [3], ignores the 'long' packet format (which is routed identically) and multicast routing capabilities provided by real SpiNNaker systems.

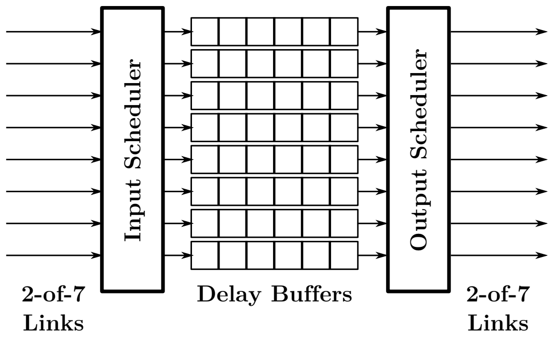


Figure 5: High-speed serial link model.

The traffic generator notably does not use a realistic traffic profile for a neural simulation but instead opts for a uniform random traffic profile, the worst case for a mesh network [1].

3.1.2 Links

The '2-of-7' links are modelled as a simple request/acknowledge interface. Requests (containing some data) are sent and, after a realistic delay, an acknowledge is returned allowing the next request to be sent.

The high-speed serial links are modelled by the system depicted in figure 5. The input scheduler moves one incoming packet into a delay buffer at a regular interval. The output scheduler takes one packet emerging from a delay buffer and transmits it on an outgoing '2-of-7' link. This is done at a regular interval in a similar fashion to the input scheduler.

4 Experimental Results

Experiments were initially conducted to validate the model's basic behaviour. These were followed by observations of the latency of packets in the system and emergency routing behaviour. All simulations ran for the equivalent of 20,000 simulated CPU cycles before being terminated.

4.1 Confirmation of Topological Properties

Table 1 shows the path lengths of packets in the system compared to their theoretically predicted values.

Sys. Size	Mean		Maximum	
(Chips)	Comp.	Meas.	Comp.	Meas.
12×12	5.653	5.646	9	9
24×24	10.326	10.290	17	17
48×48	19.663	19.119	33	33

Table 1: Path lengths in the simulated network: computed vs. measured.

The mean measured distances match those expected to 3 s.f. with the exception of the 48×48 system. This discrepancy may be due to the fact that packets taking the longest routes may not have arrived before the simulation was terminated. As a result, the mean distance in the system is likely to be skewed in favour of packets taking less time to arrive.

The maximum distances observed matched the expected values for every network tested.

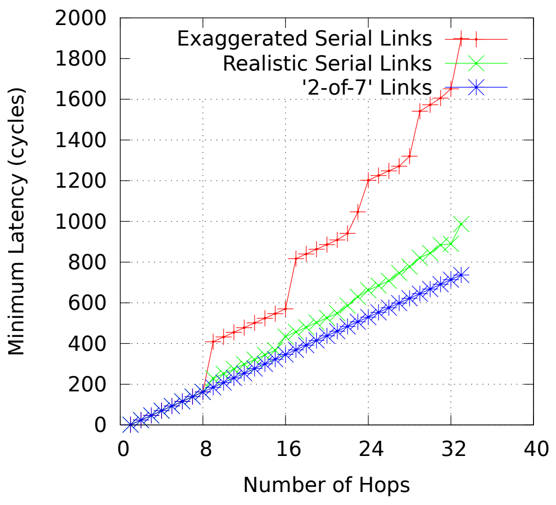
4.2 Communication Latency

The latency added by the serial links can be seen in figure 6a where the minimum packet latency is shown against the number of links (hops) used by the packet in a 48×48 chip system. The system is shown using only '2-of-7' links, with realistic serial links and with serial links with exaggerated latencies as board-to-board links.

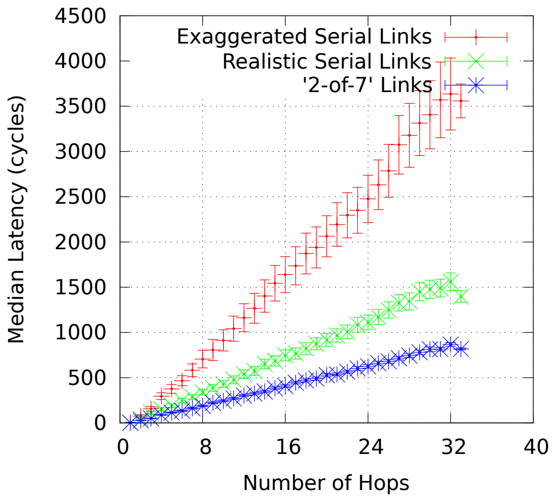
Steps can be seen every time the number of hops passes a multiple of 8, the number of consecutive chips in any single dimension on a board, after which a serial link is crossed causing increased latency. This step change can be clearly seen in the exaggerated links but is also visible in realistic links.

An extra step appears at 28 hops, the cause of which is not currently known by the author.

The median latency of an n-hop path increases smoothly as shown in figure 6b as the probability of crossing a boundary increases with the number of hops carried out. From the gradient of these lines it can be seen that the realistic serial links result in an 80.4% latency overhead.



(a) Minimum latencies



(b) Median latencies, error bars = 0.1 × Inter-Quartile Range

Figure 6: Packet latencies using various types of board-to-board links.

4.3 Dimension Order Routing Effects

Dimension order routing relies on all 'hops' taking the same amount of time to provide routes with minimal-latency (rather than just minimum hop count). This assumption breaks down in heterogeneous systems.

Figure 7 shows how the latencies vary across an idle system from a single point. It can be seen that where board boundaries (shown in white) are crossed there is a general increase in latency. Because of this, the contours are visibly distorted from the expected hexagonal shape. This is particularly visible at the bottom left and top right of the figure.

A step in latency is also visible within individual boards as can be seen in figure 8. Here there are clear edges where the dimension order routing traverses the dimensions in an order which incurs an extra board crossing. This effect is visible as a step-change in latency along the diagonals of the upper-right boards.

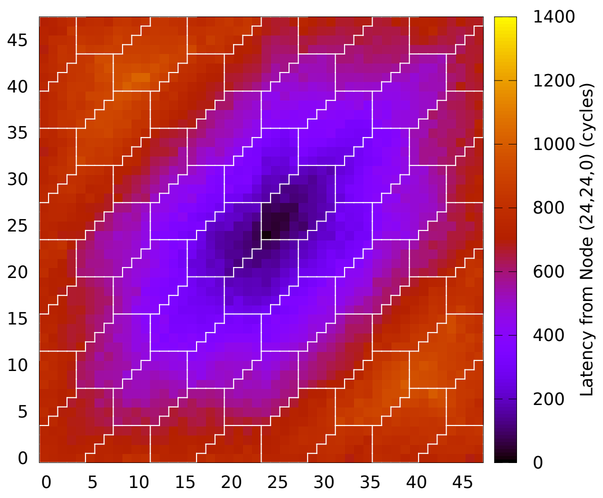


Figure 7: Heat-map of average packet latency to each chip from the central node. Note: a skewed perspective is used.

4.4 Emergency Routing

The 'emergency routing' feature of the SpiNNaker system was enabled and the frequency of use is shown in figure 9. There are clearly visible hot-spots along horizontal edges of many of the boards.

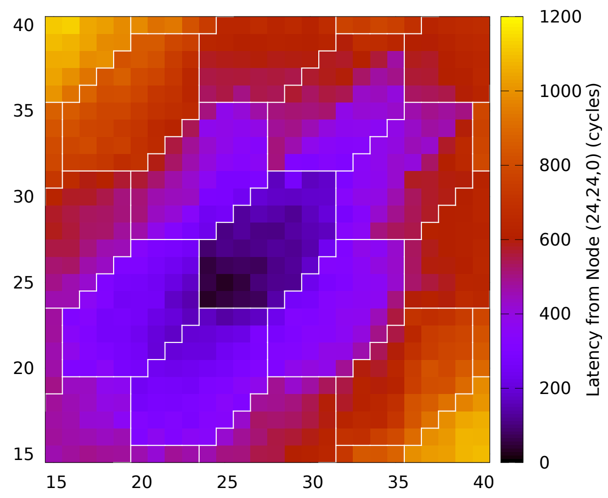


Figure 8: Latency to each chip from the central node (24,24) to a section of a 48 × 48 system. The serial latency in this system has been exaggerated to aid visibility but the effect is still present with realistic latencies.

If emergency routing is used while trying to cross north over a serial link the packet will be diverted to the west node. Once at the west node, the router tries to send the packet north-east over the same busy serial link and transmission is again delayed. This also increases congestion at that node which further increases the likelihood emergency routing will be used. This pattern is a likely cause of the hotspots displayed in the figure.

5 Conclusions & Further Work

In this paper we have described the SpiNNaker architecture and its heterogeneous interconnection. A simulator was developed for a simplified model of SpiNNaker which includes a model of the multiplexed high-speed serial inter-board links.

Experimental results show an 80% latency overhead to the median packet latency. Some of this latency can be attributed to the non-optimal path choices made by naive dimension order routing. Future work could investigate the use of alternative routes to reduce the number of borders crossed. This small modification to the routing algorithm could potentially reduce the median latency.

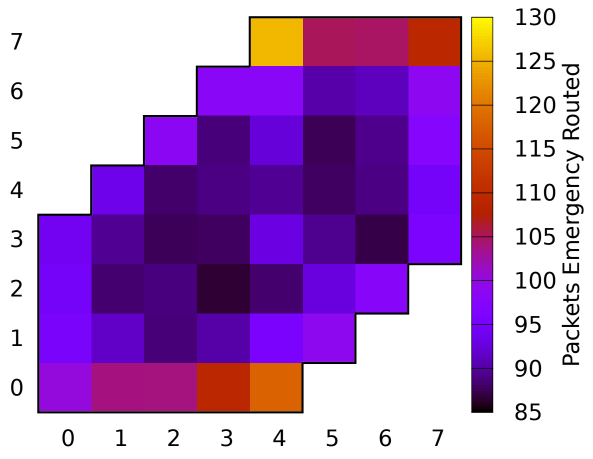


Figure 9: Heat-map showing the emergency-routing usage averaged across all boards in a normally loaded system after 20,000 cycles. Hotter areas are visible along horizontal board edges.

Finally, the 'emergency routing' facility provided by SpiNNaker unfortunately makes the assumption that all links are independent. This assumption does not apply to the serial links where multiple links share a single multiplexed path. As a result, emergency routing simply extends the path length a packet takes without improving performance. Future work may examine the effects of selectively disabling emergency routing for boundary links to avoid this effect.

References

[1] William James Dally and Brian Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers, 2004.

[2] F. Garcia Nocetti, I. Stojmenovic, and J. Zhang. Addressing and routing in hexagonal networks with applications for tracking mobile users and connection rerouting in cellular networks. *Parallel and Distributed Systems, IEEE Transactions on*, 13(9):963–971, 2002.

[3] J. Navaridas, M. Luján, J. Miguel-Alonso, L.A. Plana, and S. Furber. Understanding the interconnection network of spinnaker. In *Proceedings of the 23rd international conference on Supercomputing*, pages 286–295. ACM, 2009.

[4] L.A Plana, S.B. Furber, S. Temple, M. Khan, Y. Shi, J. Wu, and S. Yang. A gals infrastructure for a massively parallel multiprocessor. *IEEE Design*

& Test of Computers, 24(5):454–463, September–October 2007.

[5] Xilinx Inc. *Spartan-6 FPGA Family Product Brief.*