

What makes for worst-case performance?

Ivan Minevskiy
Department of Compute Science
University of British Columbia
201-2366, Main Mall, Vancouver, BC, V6T 1Z4
1-604-585-9617
ivan@cs.ubc.ca

1. Overview of Area

1.1 Average Case Performance

In asynchronous systems computational performance is determined by the speed of handshakes between its components. Very often it is intuitively assumed that the performance of a self-timed system can be measured by the average-case performance of its components. In “How to Achieve Worst-Case Performance”, Mark Greenstreet and Brian d’Alwis showed that such intuition is often wrong [3]. They looked at regular graphs of processors with Bernoulli processing times. They gave qualitative arguments that other networks should show similar behaviours, but did not quantify these claims. There have been suggestions that their results only apply to regular graphs. The goal of this project is to investigate the performance for various network topologies, degrees of connectivity, and processing time distributions to look for quantitative rules-of-thumb for the actual performance.

1.2 Percolation

Percolation theory deals with fluid flow (or any other similar process) in random media. In mathematics, percolation theory describes the behaviour of connected clusters in a random graph. Assume we have some porous material and we pour some liquid on top. Will the liquid be able to make its way from hole to hole and reach the bottom? We model the physical question mathematically as a three dimensional network of $n \times n \times n$ vertices. The edges between each two neighbors may be open (a hole) with probability p , or closed with probability $1 - p$, and we assume they are independent. For a given p , we want to find the probability that an open path exists from the top to the bottom. Such path is also called a “spanning cluster” or an “open cluster” [4].

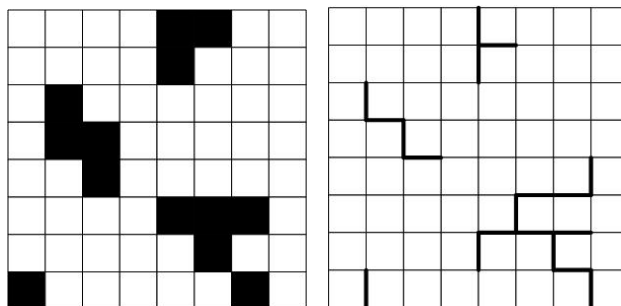


Figure 1: Sample site and bond percolation models

Usually we are interested in the behaviour for large n .

If the medium is a set of regular lattice points, then there are two types of percolation. A site percolation considers the lattice vertices as the entities to which probability is applied (Figure 1). A bond percolation applies probability to the edges (Figure 2).

In the case of infinite networks we want to know whether an infinite spanning cluster exists. In this case we may use Kolmogorov's zero-one law which states that a certain type of event, called a “tail event”, will either certainly happen or certainly not happen; that is, the probability of such an event occurring is zero or one. Hence, for any given p , the probability that an infinite cluster exists is either zero or one. Since this probability is increasing, there must be a critical p (denoted by p_c and often called “percolation threshold”) below which the probability is always 0 and above which the probability is always 1. There is a phase transition at p_c (phase transitions are what happens to water at a critical temperatures of 0 and 100 degrees Celsius). The behaviour of the model at and near this critical point displays the phenomena of scale invariance and renormalization. These central concepts from the modern theory of condensed matter physics and quantum field theory are usually difficult to grasp mathematically. I will

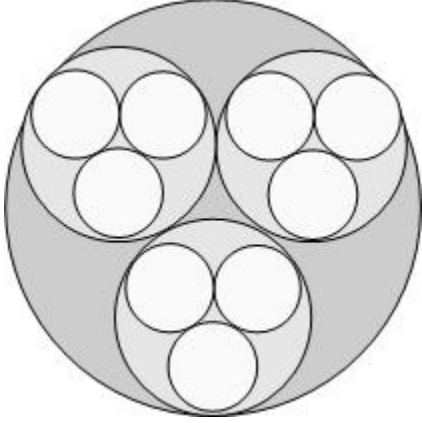


Figure 2: All sites except for the atomic ones are composed of three smaller sites

come back to them later.

In some cases p_c may be calculated explicitly. For example, for a two-dimensional square lattice, $p_c = \frac{1}{2}$ (proved by Harry Kesten in 1980 [7]).

We now wish to analyse the structure of percolating clusters more carefully. Pick a very large lattice with a spanning cluster. A percolating cluster is not very smooth. It is filled with holes and if we look at any of them we will see that it is full of holes as well. In fact, if we look at any region of lattice we will see a pattern that looks very similar to what we were looking at before. This phenomenon is called “scale invariance” and it occurs at phase transitions. Scale invariance means that the thing we are looking at looks the same at all magnifications. Thus, a percolation cluster is a self-similar fractal.

For the history of percolation theory, Grimmett [4] provides a short survey. For the connection between percolation networks, task graphs, and self-timed circuits, Greenstreet [3] provides the fundamental idea.

1.3 Renormalization

The renormalization group got its name from its early applications in quantum field theory.

Here is an example from [2]. Consider a infinite two-dimensional lattice with all of its sites separated into groups of 3 (Figure 2). A site can be coloured either black or white. It is black with probability $p_0 = p_c$ and white with probability $1 - p_0$. Denote the probability that a group of three atomic sites is coloured black

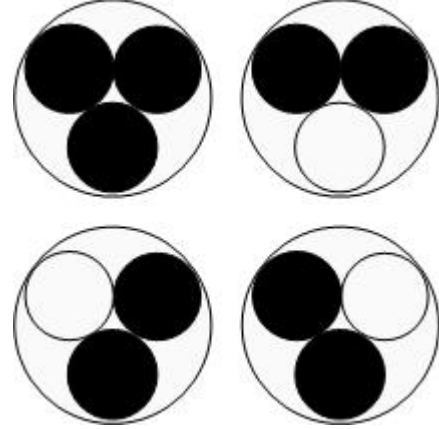


Figure 3: Four different ways to colour three sites so that their parent site is black

with probability p_1 . Non-atomic sites are black if two or more of its component circles are black. Figure 3 shows 4 different ways to colour 3 sites so that their parent site is black.

Hence, p_1 is the sum of probabilities of these 4 colourings. The first colouring has probability p_0^3 . The other three have probabilities $p_0^2(1 - p_0)$. Therefore, $p_1 = p_0^3 + p_0^2(1 - p_0)$.

Generalizing to sites at level n , $p_n = R(p) = p^3 + p^2(1 - p)$, where p is a probability of a black site at level $n - 1$. The function $R(p)$ is called the “renormalization group” (RG) transformation. The above $R(p)$ has three fixed points where $R(p) = p$. They are 0, 1, and p_c . When $p_0 < p_c$ the probability of an infinite black cluster is 0. When $p_0 > p_c$ its probability is 1. Thus, p_c is unstable and is called “critical fixed point”, while 0 and 1 are stable attractors. The macro-behaviour of the system is always determined by its fixed points of the transformation. For more in-depth introduction into renormalization, Baez [1] provides a great one.

Here is a short review on renormalization results relevant to this project (based on [10]). In late 70s Reynolds et al. successfully applied cell-to-site renormalization transformations [12]. They divided $N \times N$ lattice into many $n \times n$ cells. Then they renormalized each cell into a site using percolation threshold as a fixed point, $R(p_c) = p_c$. In early 80s Newman et al. claimed in their theorem that the number of infinite clusters coexisting is zero, one, or infinite [8]. Thus, the percolation phase transition at

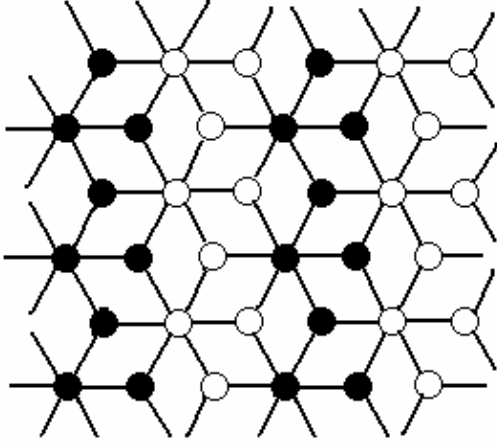


Figure 4: A 6-star network topology

the critical probability p_c is continuous and should be modeled similarly to thermal critical phenomena. In mid-90s Aizenman showed that, even on infinite square lattices, there is a low probability of two or more spanning clusters at p_c , but there is only one infinite cluster above p_c . It was also shown that in general $R(p_c) \neq p_c$. Previous estimates were not affected by this discovery, but it is considered unsafe to use cell-to-site transformation any more. Although, according to Hu et al. [6], a cell-to-cell transformation from $n \times n$ to $n/2 \times n/2$ is still valid, provided n is large enough.

2. Monte-Carlo Simulation

It was investigated how the degree of connectivity of a network topology affects its critical probability of a slow operation. A Matlab script for generating random networks was implemented. It starts with N rings of N processors each. Then new random edges are added which connect processors in neighbouring columns. Thus, all generated networks are 2-stage (i.e. all processors can be split into two groups such that there are no internal dependencies in a group). At most $2/3 \cdot N^3$ new edges are added, which is two thirds of possible ones. New edges are added in approximately 100 steps, $\left\lceil \frac{2}{300} N^3 \right\rceil$ edges per step, and are spread uniformly.

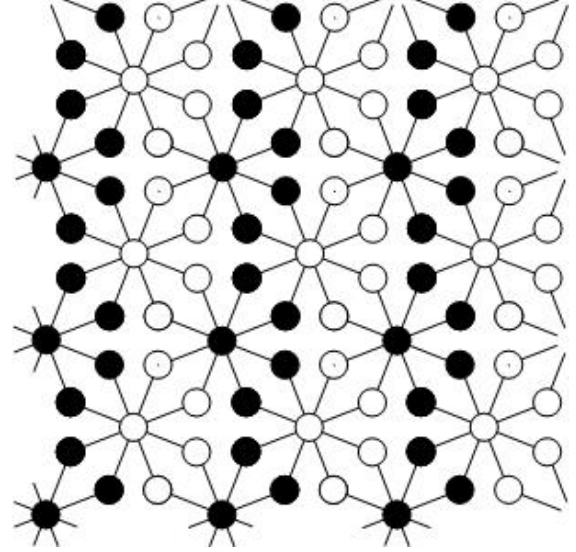


Figure 5: An 8-star network topology

Two connected stars networks with 6 and 8 rays per star were also implemented. Figures 4 and 5 show their layouts. These networks have approximately $N \times N$ processors and can be simulated as 2-stage networks.

Random topologies and the 6-star network were simulated with $N=30$ for 10000 generations. The 8-star network was simulated with $N=32$ for 10000 generations. Bernoulli time distribution was the only one used. Figure 6 summarizes the obtained results.

Table 1 shows that the obtained critical probabilities agree with the previously known results. The observed inconsistencies might be due to relatively small number of processors and generations in simulations or due to not absolutely uniform distribution of added edges.

As we can see from the chart, the star topologies do not fit into the general trend. A 6-star network has 4 connections per processor on average, but behaves as a network with 3 edges per node. Similarly, the performance of an 8-star, which has 5 edges average, is close to that of a network with mean 4.1 connections per processor. Table 2 summarizes results for star topologies.

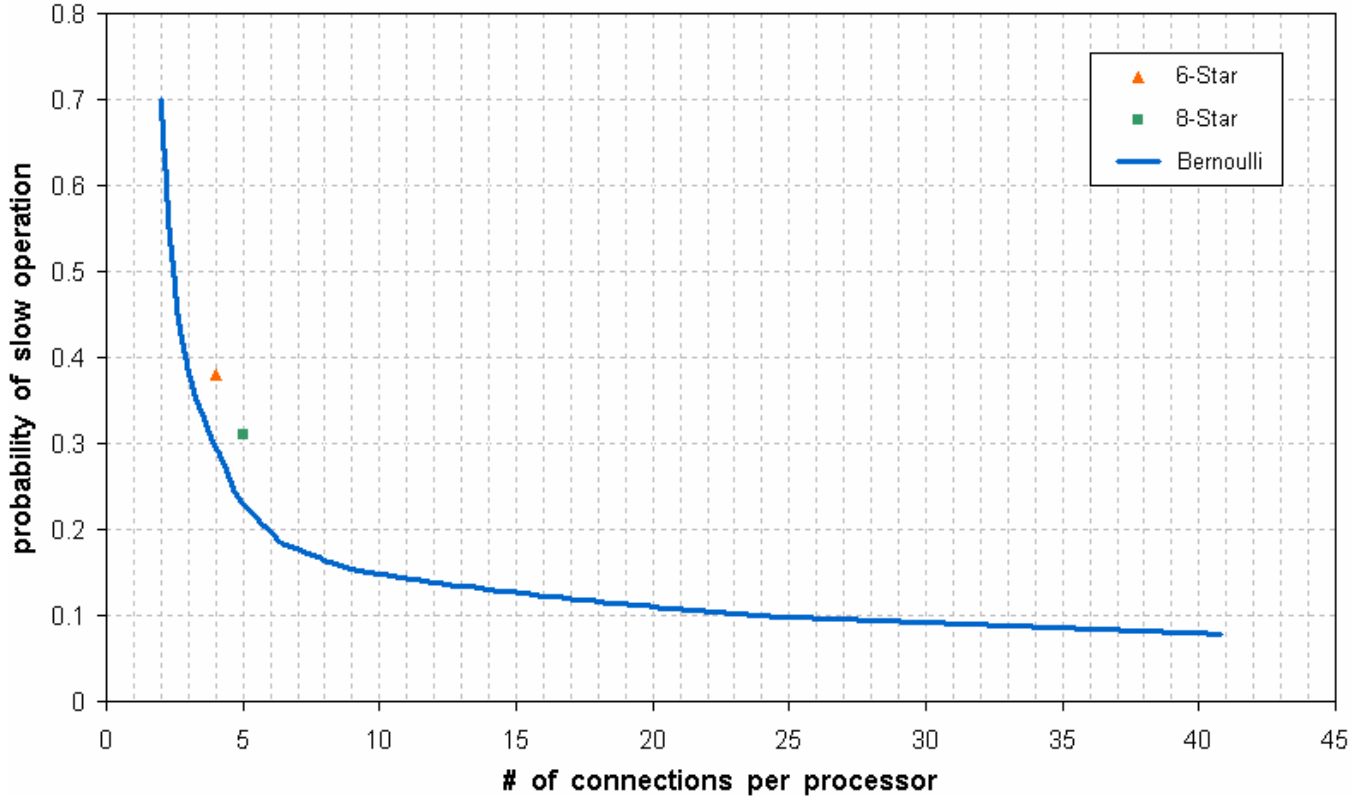


Figure 6: Monte-Carlo simulation results

| Average # of connections per processor | Equivalent topology | Obtained result | Known result | Reference |
|--|---------------------|-----------------|--------------|------------|
| 2 | ring | 0.7 | 0.72 | [5] |
| 3 | 3-connected mesh | 0.38 | 0.4 | Homework 2 |
| 4 | 4-connected mesh | 0.295 | 0.335 | [5] |

Table 1: Comparison of obtained results with previously known ones

| Topology | # of connections per processor | | | Obtained result | Average # of connections in a random topology with equivalent performance |
|----------|--------------------------------|-----|---------|-----------------|---|
| | Max | Min | Average | | |
| 6-star | 6 | 3 | 4 | 0.38 | 3 |
| 8-star | 8 | 2 | 5 | 0.31 | 4.1 |

Table 2: Obtained results for star topologies

3. Proposed Research

The work outlined above did not make any use of the scale invariance and renormalization phenomena. I propose to investigate how cell-to-cell transformations can be effectively applied to the above Monte-Carlo simulations. This would decrease required CPU time and would allow running simulations with bigger lattices and more generations.

The performed simulations took quite long because Matlab has a large overhead for operations on sparse matrices. Thus, I propose to implement simulation scripts in C instead.

It was observed that the performance of the star topologies do not fit into the random topology performance trend. Probably, the critical threshold depends on the minimal or maximal number of connections per processor, rather than on their average. I propose to investigate this fact.

The edges in the simulated random topologies were uniformly distributed. Thus, all processors had almost equal number of connections. I propose to investigate how performance is affected if connections are added non-uniformly. This might help to understand behaviour of star topologies and find a more general trend.

The simulated networks do not have bottlenecks. In the general case of an infinite lattice which was investigated, there are infinitely many paths between sites. I propose to simulate topologies with explicit bottleneck processors. For example, tree topologies connected at their roots.

In general, I propose to continue to look for quantitative rules-of-thumb for the performance dependency on the degree of connectivity. These rules should clearly show that often a system's overall performance is determined by its slower operations. To optimize a system, it is important to test how sensitive it is to various operations. Those that have the largest effect on overall system throughput should be tuned.

4. References

- [1] Baez, J. *Renormalization Made Easy*, URL: <http://math.ucr.edu/home/baez/renormalization.html>, 1999.
- [2] Forster, M. *Percolation: An Easy Example of Renormalization*, 2003.
- [3] Greenstreet, M.R. and de Alwis, B. How to Achieve Worst-Case Performance. In *Proceedings of the 7th International Symposium on Asynchronous Circuits and Systems*, pages 206-216, 2001.
- [4] Grimmett, G. R. *Percolation*, URL: <http://www.statslab.cam.ac.uk/~grg/papers/Usopts>
- [5] Grimmett, G. R. *Percolation*, chapter 1. Springer, second edition, 1999. Draft is available at <http://www.statslab.cam.ac.uk/~grg/papers/perc/>
- [6] Hu, C.K., Chert, C.N., and Wu, F. *Journal of Statistical Physics*, Y82:1199, 1996.
- [7] Kesten, H. The critical probability of bond percolation on the square lattice equals $\frac{1}{2}$. *Mathematical Physics*, 44:41-59, 1980.
- [8] Newman, C.M. and Schulman, L.S. *Journal of Statistical Physics*, 26:613, 1981.
- [9] Pang, P. B., and Greenstreet, M.R. Self-timed meshes are faster than synchronous. In *Proceedings of the Third International Symposium on Advanced Research in Asynchronous Circuits and Systems*, pages 30-39. IEEE, 1997.
- [10] Stauffer, D. Minireview: New results for old percolation. *Physica A*, 242(1-2):1-7, 1997.
- [11] Stauffer, D. and Aharony, A. *Introduction to Percolation Theory*. London: Taylor & Francis, second edition, 1992.
- [12] Reynolds, P.J., Stanley, H.E., and Klein, W. *Journal of Physics*, C10:LI67, 1977.