Token Passing Networks
Abstract prepared for Permutation Patterns 3

Steve Waton
University of St. Andrews
stevew@mcs.st-andrews.ac.uk

November 29, 2004

1 Introduction

The classifaction and enumeration of stack sortable permutations by Knuth was an inspiration for the study of pattern classes[7]. The work was later generalised by Tarjan who studied networks of stacks and queues which he introduced using the analogy of rail road switch yards. Tarjan showed that every acyclic network generates a closed class and found a lower bound for the shortest basis element in one of the most widely studied networks, several stacks connected in series[10].

In the late nineties Mike Atkinson and others studied finite networks, those that can only hold a finite number of tokens at any time[1, 3, 2]. They answered three critical problems for such networks. Firstly, the decision problem: given a network and a permutation decide whether the permutation can be generated. Secondly, the enumeration problem: how many permutations of each length can a particular network generate. Thirdly, the basis problem: given a particular network find the set of minimal permutations which cannot be generated. A final problem, sometimes called the network problem, given a closed class find a network which generates it or show that no such network exists, remains open.

We consider networks which contain some infinite components, such as queues and stacks, and so can hold infinitely many tokens. The methods used to analyse finite networks rely on a rank encoding of permutations where the maximum rank is bounded by the size of the network, allowing the permutations to be expressed as words over a finite alphabet. Since our networks have no such maximum capacity these methods fail in our setting, however some progress has been made in this area[4, 9, 6].

We give an algorithm to determine whether or not a network can generate every permutation of any length. Such a network is called complete. In doing so we use two results of network composition. The first, on parallel composition, is due to Atkinson and Beals who give a construction for a ungeneratable permutation given two networks composed in parallel together with permutations they cannot generate[5]. The second result, on serial composition, is a corollary of Marcus and Tardos's recent proof of the Stanley-Wilf conjecture[8].

These two results, together with a construction we call path parallel expansion, mean we need only consider strongly connected components of our network to answer the completeness problem. It is then a simple task to show that a network is complete if and only if it contains one of two types of subnetwork, categorised as an infinite loop or a pair of strongly connected stacks.

2 Outline of the result

We begin by defining a token passing network as a directed graph with a specified input node or source which has zero in degree and a specified output node or sink which has zero out degree. The remaining nodes may be labeled as stacks, queues and various deques or left blank.

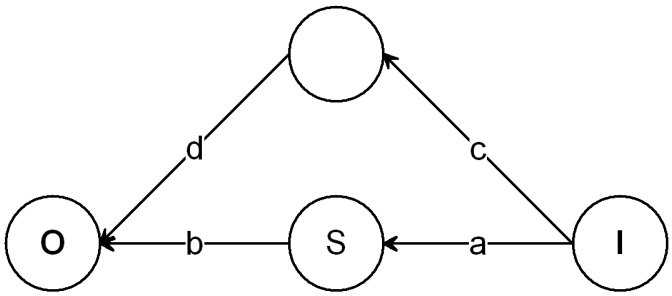

Figure 1: A Typical Network

We next define the codeword alphabet which is the edge set of the graph together with extra symbols to represent tokens moving to and from different ends of any deques. We define a set of states which describe the number tokens stored at each point on the network and a transition function which given a state and a codeword symbol describes the move of the token involved, if such a move is possible.

With these structures in place we can describe permutations in terms of codewords which generate them hence hence prove that a particular permutation can be generated by describing the codeword which generates it.

We immediately have the following results.

- Every acyclic network is complete.
- A parallel composition is complete if and only if one of its parts is complete.
- A serial composition is complete if and only if one of its parts is complete.

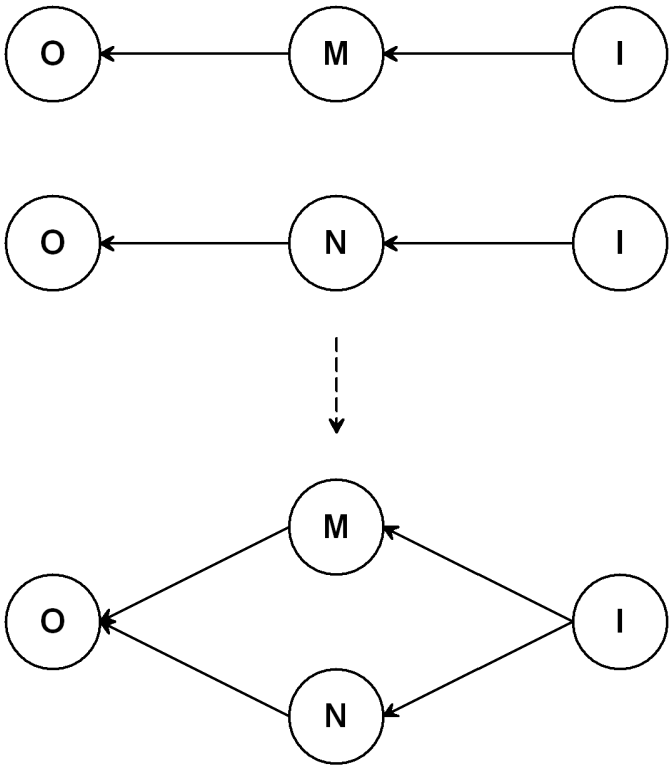

Figure 2: Parallel Composition of two networks.

Next we define the concept of a subnetwork in terms of constructing the original network by inserting parts into the subnetwork.

We prove that a network which contains a complete subnetwork is complete.

We then define two special classes of complete network, which we prove that every complete network must contain as a subnetwork. This is proved by constructing a new network which consists of the parallel composition of all paths which do not contain loops together with any strongly connected components they pass through. We prove that this new network is complete if and only if the network from which it was constructed is complete. This larger network is then complete if and only a path through it is complete, and this is true if and only if one of the strongly connected components is complete. A simple analysis of such components completes our result.

Unfortunately this path parallel expansion of a network does not help us to answer further questions about infinite networks, which seem to require more

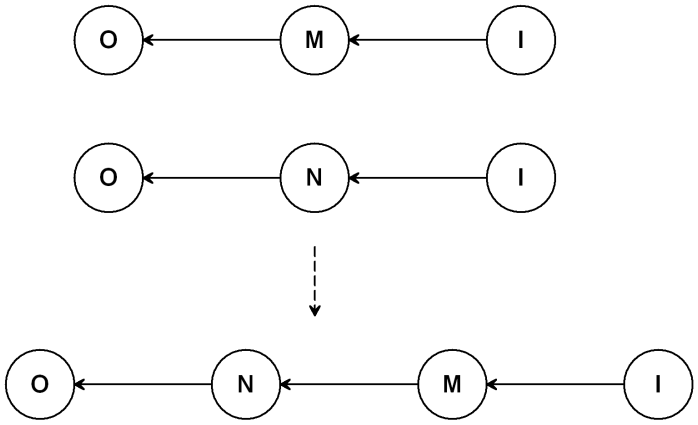

Figure 3: Serial Compositions of two networks.

delicate decompositions. A first conjecture would be that a network which contains an infinitely based subnetwork is either infinitely based or complete.

References

[1] M.H. Albert, M.D. Atkinson, and N. Ruškuc. Regular closed sets of permutations. *Theoretical Computer Science*, 306:85–100, 2003.

[2] M.H. Albert, N. Ruškuc, and S. Linton. On the permutational power of token passing networks. Technical report, Department of Computer Science, University of Otago, 2004.

[3] M.D. Atkinson, M.J. Livesey, and D. Tulley. Permutations generated by token passing in graphs. *Theoretical Computer Science*, 178:103–118, 1997.

[4] M.D. Atkinson, M.M. Murphy, and N. Ruškuc. Sorting with two ordered stacks in series. *Theoretical Computer Science*, 289:205–223, 2002.

[5] Michael D. Atkinson and Robert Beals. Permuting mechanisms and closed classes of permutations. *Australian Computer Science Communications*, 21(3):117–127, 1999.

[6] Miklós Bóna. A survey of stack-sorting disciplines. *Electronic J. Combinatorics*, 9(2):A1, 2003.

[7] D.E. Knuth. *The art of computer programming, volume 1, Fundamental Algorithms*. Addison-Wesley, 1973.

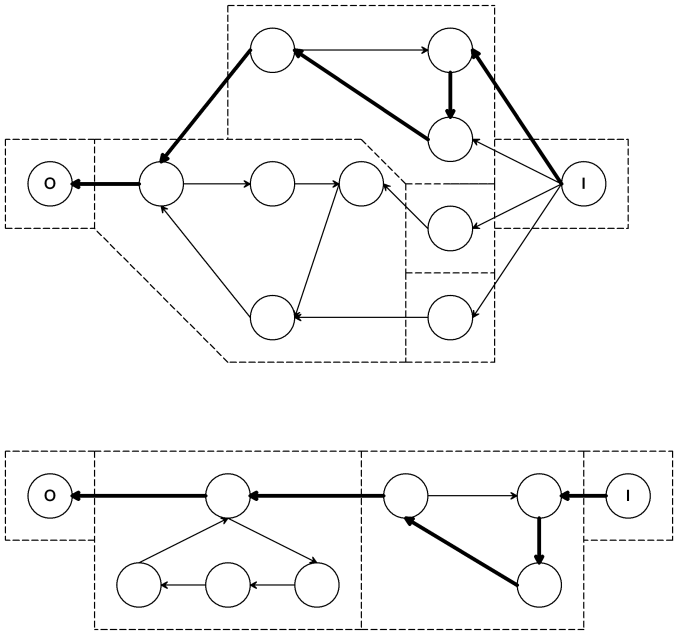

Figure 4: A path and the components it passes through.

[8] A. Marcus and G. Tardos. Excluded permutation matrices and the stanley-wilf conjecture. *Journal of Combinatorial Theory Series A*, 107:153–150, 2004.

[9] M.M. Murphy. *Restricted permutations, antichains, atomic classes and stack sorting*. PhD thesis, University of St. Andrews, 2002.

[10] R.E. Tarjan. Sorting using networks of queues and stacks. *Journal of the ACM*, 19:341–346, 1972.