Verifying an ATM switch with Formal Methods

A. Février, E. Najm, N. Prost, F. Robles
ENST Paris
Département Réseaux
46, Rue Barrault
75013 - Paris
FRANCE
Tel: +33 (1) 45 81 73 52
Sec: +33 (1) 45 81 70 85
Fax: +33 (1) 45 89 16 64
{fevrier, najm, prost, robles}@res.enst.fr

September 1994

Abstract

In this paper we exhibit a formal specification and verification of an ATM switching fabric with the formal specification language LOTOS and the Caesar / Aldebaran verification tools. Switching fabrics have been proved to behave correctly using graph theory and other mathematical theories. We provide for an alternative proof method motivated by its generality and its independence of the specific design used in the fabric.

Keywords: FDT, LOTOS, ATM, Batcher, Banyan, Caesar, Aldebaran

1 Problem Statement

The asynchronous transfer mode (ATM) has been chosen by ITU-T (formally CCITT) as the switching and multiplexing technique for the broadband access to ISDN. Several architectural designs of high speed switching fabrics exist (e.g. CNET's Prelude, Bellcore's Moonshine, Alcatel's Roxanne, AT&T's starlite, and others). Theoritical work has already been done on switching fabrics but we aim to present a new framework for proving properties on such components.

2 Informal Description

In this paper we exhibit a formal specification and verification of an ATM switching fabric with the formal specification language LOTOS[ISO88] and the Caesar [Gar90]/ Aldebaran [FGM+92, Fer89] verification tools. We formally specify and partially verify an ATM switching fabric inspired from the Batcher Banyan multistage interconnection network and more specifically from the starlite switching fabric. We choose a switching fabric having appropriate size and functions that allow an easy and simple illustration of our methodology. The ATM switching fabric we consider (figure: 1) is composed of five functions: filter, routing table, batcher, trap, Banyan. The Filter function allows only valid cells to enter the switch (by rejecting empty cells for instance), the Routing Table function performs header translation and the routing tag addition to the cell, the Batcher function sorts cells according to their

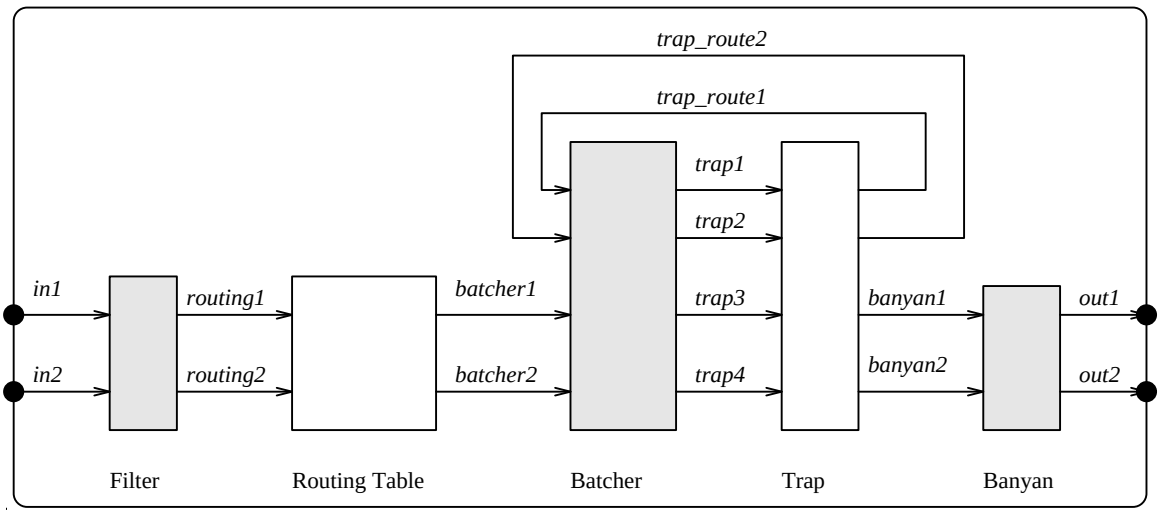

Figure 1: the starlite switch

destination address, the Trap function avoids output conflicts by recirculating (all but one) cells contending for the same output and the Banyan function makes the switching of cells according to their destination address.

We choose LOTOS as the formal description language. Because of the parallelism concept (and the underlying notion of synchronism) LOTOS appears to be more appropriate for our purpose than other formal languages like e.g. SDL or Estelle. Using the LOTOS composition operators, we provide a specification whereby the five functions of the fabric are reflected in the architecture of the specification. The verification method we use consists in formally comparing two specifications of the switching fabric. The first specification is abstract and describes the intended, externally observable behavior of the switch. The second specification reflects the joint operation of the subcomponents that form the switch. We establish that these two specifications are equivalent. Using this method we prove that cells are correctly routed.

After the proof of an equivalence property, we plan to proof a correctness property. We formalize the description of the last function (Banyan) in order to prove that there is no internal contention. We use the Caesar tool to prove that there is no wrong transition.

3 Specification of the switch

We specify first the ATM switch. We use a 2X2 switch. The extension to a nXn switch is straightforward.

When we start specifying the switch we have to work on a simple version to ease the building of the framework. Moreover, it is easier to present a reduced version which allows easy understanding and removes nothing to the validity of the framework.

3.1 The ATM switch: Abstract Specification

We specify the behavior of the switch. The specification is as simple as possible. We plan to use a 2X2 switch to show the method (figure: 2). The method is still applicable for a nXn switch.

The switch has two input gates, two output gates and two memory cells to lower the contention.

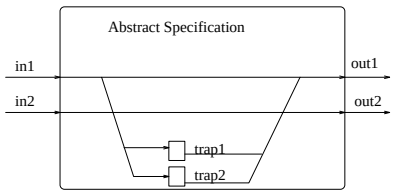

Figure 2: The Abstract Specification

The behavior is then simple. The specification waits for two input cells, then exit two output cells on its exit gates occasionnaly keeping in its memory one or two cells to decrease contention.

There should be no contention in any function of the switch. Nevertheless if all input cells were to go through the same exit port then contention would occur. Indeed the probability of such a case is really low and can be considered as null.

3.2 The ATM switch: Concrete Specification

We consider the switch as described in figure 1.

It has five subcomponents:

1. Filter: This process discards non-authorized cells.
2. Table: This process analyses the header of incoming cells and generates self routing cells.
3. Batcher: This process receives incoming self routing cells from all input lines and delivers them sorted according to their tag value (say the output port) to its output lines.
4. Trap: This process receives incoming self routing cells from all input lines and filters them so as to give to the Banyan block only self routing cells with different tag values.
5. Banyan: This process switch self routing cells to the output port associated to their tag value.

3.3 Proof

We use for the two specifications a cell generator which sends any cell to the specifications of the switch. The generators for both specifications are the same. We then use Caesar / Aldebaran to produce the automata. Then we compare them to prove that there are observationaly identical. As there are hidden gates in the concrete specification we could not run for a strong equivalence.

4 The Banyan

4.1 Description

We specify the Banyan because it is the only place in the switch where a contention may takes place. We use a 4x4 Banyan to show a more interesting case of internal composition.

The Banyan is built using switching elements (figure: 3) which take two input cells and output them in a correct order with respect to the most significant bit of their tag value. We put these elements according to the regular sketch of the Banyan (figure: 4). Then we add a converter which removes the tag of the cells before their output.

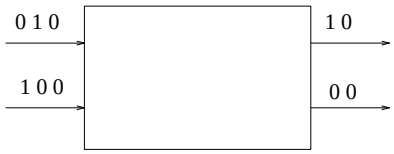

Figure 3: The Switching Element

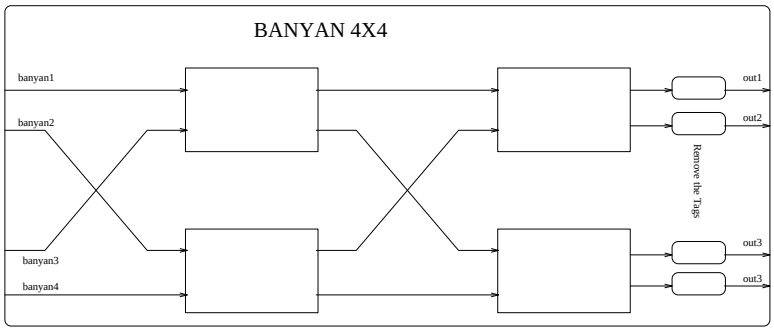

Figure 4: The Banyan

4.2 No Contention

With the specification of the Banyan described previously, we plan to prove that:

1. there is no internal contention,
2. the packets are correctly routed.

We make the assumptions that at the entry the packets are correctly ordered w.r.t. their exit gate and that they are on the topmost gates if there are empty cells.

We use two other processes (figure: 5):

1. a generator which sends to the Banyan every combination of inputs allowed,
2. a tester which scans the output and raises an error if a cell exits on a wrong gate.

Furthermore, the switching element is modified to add an "error" gate if two cells would be sent to the same output (figure: 6).

Then we use Caesar which produces an automaton in which there is no "error" transition. This proves the correctness of the Banyan.

the automaton produced by a tool developped by Jeron and Jard [JJ94] by the sending of three cells to be sent on gates {1, 3, 4} is displayed on figure 7

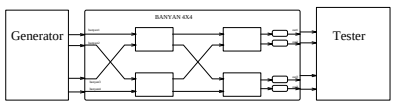

Figure 5: The analysis of the Banyan

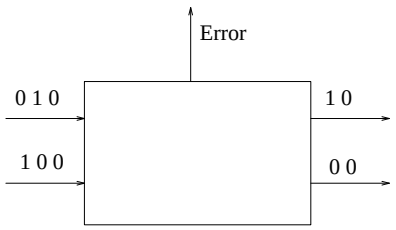

Figure 6: The Switching Element with error detection

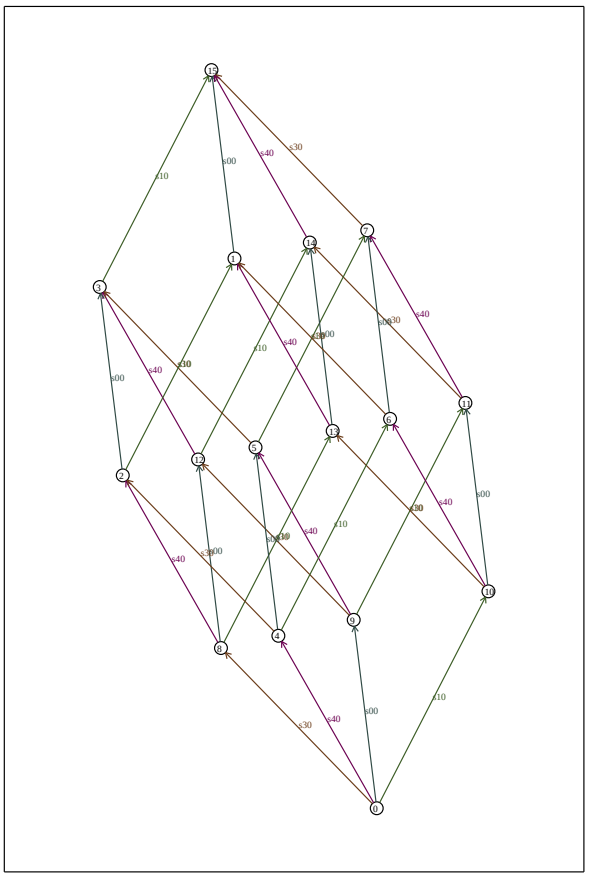

Figure 7: An automaton produced by Aldebaran

5 Comments on Results and further work

We used the Caesar / Aldebaran tools to prove some properties of an ATM switch. We showed that two specifications were observationnaly equivalent so we could use either one depending of the level of refinement we wanted to have. We could use the abstract specification as a component of an ATM network, and the concrete one to prove some properties on the switch.

Such equivalences can be used to first specify a global system, then refine the specification or design of each component, preserving the validity of any proof already done.

We plan to extend this paper to include proof of validity of the simplifications done, and to show the complexity of the algorithms used.

6 Conclusion

We presented general methods to do some proofs using FDTs. We may extend the method to any number. Moreover, the framework is still applicable if a small change was made on the specification, whereas a more mathematical proof would have to be redone in whole.

References

[Fer89] Jean-Claude Fernandez. Aldebaran: A tool for verification of communicating processes. Rapport SPECTRE C14, Laboratoire de Génie Informatique — Institut IMAG, Grenoble, September 1989.

[FGM+92] Jean-Claude Fernandez, Hubert Garavel, Laurent Mounier, Anne Rasse, Carlos Rodríguez, and Joseph Sifakis. A toolbox for the verification of lotos programs. In Lori A. Clarke, editor, *Proceedings of the 14th International Conference on Software Engineering ICSE'14 (Melbourne, Australia)*, pages 246–259, New-York, May 1992. ACM.

[Gar90] Hubert Garavel. Cæsar reference manual. Rapport SPECTRE C18, Laboratoire de Génie Informatique — Institut IMAG, Grenoble, November 1990.

[Hor93] Ellis Horwood. *M. de Prycker*, chapter 4. Asynchronous Transfer Mode: Solution for B-ISDN, 1993.

[ISO88] ISO. Lotos — a formal description technique based on the temporal ordering of observational behaviour. International Standard 8807, International Organization for Standardization — Information Processing Systems — Open Systems Interconnection, Genève, September 1988.

[JJ94] T. Jéron and C. Jard. 3d layout of reachability graphs of communicating processes. Submitted, 1994.

[Pal93] A Paltaving. Non blocking architectures for atm switching. In *IEEE Communication Magazine*, 1993.

[Tob90] F.A. Tobagi. Fast packet switching architecture for broadband integtrated services digital networks. In *IEEE*, volume 78(1), pages 133–167, 1990.

[Wit93] Witte. Architectures for atm switching systems. In *IEEE communication Magazine*, February 1993.