Aldébaran: Users's manual

Jean-Claude Fernandez and Laurent Mounier
IMAG–LGI
BP53X
38041 GRENOBLE Cedex
Tel : 76 51 49 15
e-mail `fernand@imag.imag.fr`

June 29, 1990

1 Introduction

Aldébaran [2] is a system for verifying communicating systems, represented by transition systems whose arcs are labelled by action names. It allows the reduction and the comparison of labeled transition system with respect to equivalence relations such as bisimulation equivalence [6], observational equivalence and safety equivalence [8]. It also allows compositions of labeled transition systems to be treated by different strategies of reductions. Aldébaran is written in C and runs on UNIX system. At present, the limit of the size of a labeled transition system on a SUN 3/60 with 50 Mega-bytes of memory, is one million transitions, because each transition is represented in twenty bytes. The reduction algorithms are based on Paige & Tarjan algorithm [7], that solves the relational coarsest partition problem in $O(m \log n)$ time, where m is the number of transitions and n is the number of states. This allows to reduce labeled transition systems with hundred thousands of transitions in some minutes. It has a simple input format which is a list of triples representing the transition relation. Aldébaran may be interfaced with other systems which manipulate labeled transition systems. For instance, Aldébaran is interfaced with a LOTOS compiler [4] and a common object code produced by LUSTRE and ESTEREL compilers [1]. In section 2, we briefly describe the "silent" use of Aldébaran. For a more complete description of Aldébaran, see [3] or [2]. In section 3, we present performances of Aldébaran on an example of verification: a scheduling problem [6].

2 Description of Aldébaran commands

NAME aldebaran - minimize or compare labeled transition systems with respect to an equivalence relation.

SYNOPSIS **aldebaran** [options] name_1 ...name_i ...

DESCRIPTION Aldebaran accepts two input file format: ASCII file and binary file (option **-bin**). Files whose names end with ".gra" are taken to be binary files whereas files whose names end with ".aut" are taken to be ASCII files. Each file contains a labeled transition system which is set up with a *descriptor* and a set of *edges*. A descriptor (resp. an edge) is represented as:

$$(\mathit{initial\_state}\,,\ \mathit{number\_of\_transitions}\,,\ \mathit{number\_of\_states})$$
$$(\mathit{state}\,,\ \mathit{name}\,,\ \mathit{state})$$

The following options are processed by Aldébaran.

-help help.

-bin Indicate binary files.

-inv Inverse representation for labeled transition system (default: direct representation).

-stat Some statistics are displayed.

-step Transformations of labeled transition systems in order to obtain normal form are interactive.

-fly The equality of the labelled transition systems, contained in name_1 and name_2, with respect to a weak bisimulation equivalence (tau* a) is tested. The result TRUE or FALSE is displayed.

-bmin

-omin

-amin

-smin Each labelled transition system, contained in name_i, for $i = 1, n$, is minimized with respect to bisimulation (resp. observational, acceptance models and safety) equivalence. The result of minimization is displayed.

-bequ

-oequ

-aequ

-sequ The equality of the labelled transition systems, contained in name_1 and name_2, with respect to bisimulation (resp. observational, acceptance models and safety) equivalence is tested. The result TRUE or FALSE is displayed.

-bcla

-ocla

-acla

-scla For each labelled transition system, contained in name_i, for $i = 1, n$, equivalence classes with respect to bisimulation (resp. observational, acceptance models and safety) equivalence are displayed.

REMARK The last twelve options inhibit the interactive use of Aldébaran.

3 Example

3.1 Scheduler

We give an example of reduction carried out by Aldébaran. The reduction is based on observational equivalence. Reduction with respect to observational equivalence consists of transforming the labeled transition system by computing transitive closure of the transition relation labelled by τ and finding the coarsest partition with respect to the transition relation and the universal partition. The example is Milner's problem of scheduling (see [6], page 33). This example is interesting for evaluation purposes because the numbers of states, transitions and equivalence classes grow in the same proportion. We give two specifications in Lotos [4]. We consider a ring of n elementary identical components, called *cyclers*. A cycler specification in Lotos is:

```
process CYCLER[gi, ai, bi, gi+1 ] : noexit :=
    gi ; ai ;
          ((  bi ; gi+1 ; CYCLER[ gi, ai, bi, gi+1])
              []
             ( gi+1 ; bi ; CYCLER[ gi, ai, bi, gi+1]))
endproc
```

A cycler should cycle endlessly as follows: (i) Be enabled by predecessor at gi, (ii) Receive initiation request at ai (iii) Receive termination signal at bi and enable successor at $gi + 1$ in either order. We give two specifications of scheduler: the first one is such that the ai and bi are visible ; in the second one only the ai are visibles. (This last specification expresses that the scheduler is observationally equivalent to $(a_1...a_n)^\omega$). In both cases, we give a table that summarizes the time (in seconds) spent to find the coarsest partition compatible with the transition relation and the universal partition.

3.2 First specification

```
specification SCHEDULER [a1, ..., an, b1, ..., bn ] : noexit behaviour
 hide g1, ..., gn in
 (cycler[g1, a1, b1, g2]
       |[g1, g2]|
     (
       ...
      cycler[gi, ai, bi, gi+1]
       |[gi+1]|
       ...
         (cycler[gn, an, bn, g1] ||| g1; stop)
```

```
      ...
    ))
   where library cycler endlib
endspec
```

numbers of cyclers	number of states	number of transitions	number of classes	time
2	13	35	9	0.017s
3	37	139	25	0.05s
4	97	453	65	0.26s
5	241	1321	161	0.88s
6	577	3595	385	2.6s
7	1345	9339	897	7.28
8	3073	23465	2049	20.5s
9	6913	57687	4663	56.3s
10	15361	138111	10241	159.8s

3.3 Second specification

```
 specification SCHEDULER [a1, ..., an] : noexit behaviour
  hide g1, ..., gn, b1, ..., bn in
  (cycler[g1, a1, b1, g2]
        |[g1, g2]|
      (
        ...
       cycler[gi, ai, bi, gi+1]
        |[gi+1]|
        ...
          (cycler[gn, an, bn, g1] ||| g1; stop)
        ...
      ))
   where library cycler endlib
endspec
```

numbers of cyclers	number of states	number of transitions	number of classes	time
2	13	35	3	0.01s
3	37	325	4	0.05s
4	97	1465	5	0.15s
5	241	5851	6	0.6s
6	577	21853	7	1.9s
7	1345	78247	8	6.9s
8	3073	272209	9	24s
9	6913	927451	10	80s

Notice that in both cases, time increases quasi linearly with the number of transitions.

3.4 Other Results

In this section, we give the results obtained by applying the decision procedure "on the fly" (produce of labeled transition systems) to labeled transition systems.
The labeled transition systems are previously generated from LOTOS specifications by using CÆSAR ([4,9]) and stored, and consequently the results can be compared with the classical verification procedure.
Two examples are described here: the first one is the scheduler described by Milner and the second one is an alternating bit protocol called Datalink protocol [10].
The following notations are used:

- n_i and m_i denote the number of states and transitions of the two labeled transition systems ($i = 1, 2$).
- n denotes the number of states of the product which have been effectively analyzed.
- $t1$ is the time needed by the classical decision procedure of Aldébaran.
- $t2$ is the time need by the decision procedure "on the fly".

The times given here are elapsed times, obtained on a SUN 3-80 Workstation. Only the verification phase is taken into account.

3.4.1 Milner's scheduler

The results are given for different values of N.

N	n1	m1	n2	m2	n	t1	t2
7	1345	5377	7	7	449	0:14	0:11
8	3073	13825	8	8	1025	0:46	0:34
9	6913	34561	9	9	2305	2:50	1:44
10	15361	84481	10	10	5121	13:07	5:25

3.4.2 Datalink protocol

The Datalink protocol is an example of an alternating bit protocol. The LOTOS specification provided to CÆSAR is described in [10].

By varying the number of the different messages which can be transmitted (noted N), labeled transition systems of various sizes can be obtained. However, for $N > 40$, the memory required by the classical decision procedure of Aldébaran becomes too large, and consequently the verification can no longer be performed with this procedure.

N	n1	m1	n2	m2	n	t1	t2
20	7241	10560	41	440	1661	0:24	0:19
30	15661	23040	60	930	3691	0:57	0:55
40	27281	40320	80	1640	6521	2:07	1:45
50	42101	62400	101	2600	10151	—	2:27
60	60121	89280	121	3720	14581	—	3:42
70	81341	120960	140	4970	19811	—	6:42
80	105761	157440	161	6560	25841	—	9:23

References

[1] P. Couronné, J.A. Plaice, and J.B. Saint. The lustre esterel portable format. *unpublished*, 1986.

[2] J. C. Fernandez. *Aldébaran, Un système de vérification par réduction de processus communicants*. PhD thesis, Université de Grenoble, 1988.

[3] J. C. Fernandez. *Aldébaran: User's Manual*. Technical Report, LGI-IMAG Grenoble, 1988.

[4] H. Garavel. *Compilation et vérification de programmes LOTOS*. PhD thesis, Université Joseph Fourier de Grenoble, 1989.

[5] S. Graf and J. Sifakis. *Readiness Semantics for Regular Processes with Silent Action*. Technical Report Projet Cesar RT-3, LGI-IMAG Grenoble, 1986.

[6] R. Milner. A calculus of communication systems. In *LNCS 92*, Springer Verlag, 1980.

[7] R. Paige and R. Tarjan. Three partition refinement algorithms. *SIAM J. Comput., No. 6*, 16, 1987.

[8] C. Rodriguez. *Spécification et validation de systèmes en XESAR*. PhD thesis, Institut National Polytechnique de Grenoble, 1988.

[9] Hubert Garavel and Joseph Sifakis. Compilation and verification of lotos specifications. In L. Logrippo, R. L. Probert, and H. Ural, editors, *Proceedings of the 10th International Symposium on Protocol Specification, Testing and Verification (Ottawa)*, IFIP, Amsterdam, June 1990.

[10] Juan Quemada, Santiago Pavón, and Angel Fernández. Transforming lotos specifications with lola: the parametrized expansion. In Kenneth J. Turner, editor, *Proceedings of the 1st International Conference on Formal Description Techniques FORTE'88 (Stirling, Scotland)*, pages 45–54, North-Holland, Amsterdam, September 1988.