



**HAL**  
open science

## Discrete-Event systems: Symbolic representations

Yannick Pencolé

► **To cite this version:**

Yannick Pencolé. Discrete-Event systems: Symbolic representations. Doctoral. Module T7. Discrete-Event systems, Girona, Spain. 2022, pp.41. hal-04775887

**HAL Id: hal-04775887**

**<https://laas.hal.science/hal-04775887v1>**

Submitted on 10 Nov 2024

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

---


# Module T7. Discrete-Event systems

## Symbolic representations

---

Yannick Pencolé  
7th July 2022  
LAAS-CNRS, France





---

---

# Preliminaries about propositional logic

# Preliminaries about propositional logics : syntax

- Propositional formulas  $Forms(\mathcal{V})$  over  $\mathcal{V}$  :
  - ▶ Constant  $\top$  (is true, tautology),  $\perp$  (is false, contradiction)
  - ▶ Propositional variable  $v$  from  $\mathcal{V}$
  - ▶ By induction : let  $\Phi, \Phi'$  be two propositional formulas :
    - $\neg\Phi$  (negation)
    - $\Phi \wedge \Phi'$  (conjunction)
    - $\Phi \vee \Phi'$  (disjunction)
    - $\Phi \rightarrow \Phi'$  (implication)
    - $\Phi \leftrightarrow \Phi'$  (equivalence)

Some formulas of  $Forms(\mathcal{V})$  with  $\mathcal{V} = \{a, b, c\}$

$$\begin{array}{ccc} a & \top & a \wedge (b \vee c) \\ \perp \vee a & a \leftrightarrow \neg(b \rightarrow c) & \neg\neg(\perp \vee c) \end{array}$$



---

## Preliminaries about propositional logics : semantics

---

- Propositional variables  $v \in \mathcal{V}$  : either true  $T$  or false  $F$
- Assignment  $\alpha : \mathcal{V} \rightarrow \{T, F\}$

$$\alpha(a) = T, \alpha(b) = T, \alpha(c) = F$$

- Semantics of a formula  $\Phi$  over assignment  $\alpha$

$$\alpha(b \vee c) = T, \alpha(\top) = T, \alpha(b \wedge c) = F, \alpha(\neg(c \rightarrow a)) = F$$

- $\alpha$  is a **model** of  $\Phi$  if  $\alpha(\Phi) = T$  (denoted  $\alpha \models \Phi$ )

$\alpha$  is a model of  $(b \vee c)$  but not a model of  $(b \wedge c) = F$ .



---

# Preliminaries about propositional logics : satisfiability

---

- $\Phi$  is **satisfiable** if it has at least a model  $\alpha$ .
- $\Phi$  is **inconsistent/unsatisfiable** if it has **no model**.
- $\Phi$  is **valid** if every  $\alpha$  is a model.

- $b \wedge c$  is satisfiable
- $c \rightarrow c$  is valid
- $\neg(c \rightarrow c)$  is inconsistent

**SAT : Checking satisfiability of  $\Phi$  is a NP-complete problem**

---



# Symbolic diagnosis methods : motivations

---

Main issue about diagnosis in DES :

- The number of states in a DES is **exponential** in the number of components in the DES.
- **Combinatorial explosion problem**

One way to go : the symbolic way

- An implicit representation that aims at avoiding the enumeration of states
- A state holds a set of properties.
- A belief state holds the properties that are common to the states it contains
- Used for diagnosis and diagnosability analysis



---

# Symbolic encoding of set of objects


---

Consider a set  $\mathcal{O}_m$  of  $m = 2^n$  objects ( $n$  integer).

- How to design a logic that maps any object of  $\mathcal{O}_m$  into a logical formula?



---



## Symbolic encoding of a set of objects : one solution

---

1. Consider the propositional variables  $v_1, \dots, v_n$
2. Assign a unique index  $i$  to every object  $\mathcal{O}_m$

$$\mathcal{O}_m = \{o_0, o_1, o_2, \dots, o_{m-1}\}$$

3. Encode index  $i$  as a Boolean value

$$\mathcal{O}_m = \{o_{\langle FF\dots F \rangle}, o_{\langle FF\dots FT \rangle}, \dots, o_{\langle TT\dots TT \rangle}\}$$

4. **A formula  $\Phi_{o_i}$  represents object  $o_i$  iff**
  - ▶  $\Phi_{o_i}$  has one and only one model  $\alpha$
  - ▶ **If  $i = \langle b_1, \dots, b_n \rangle$  then  $\alpha(v_j) = b_j$  for  $j \in \{1, \dots, n\}$ .**





---

# Symbolic encoding of a set of objects : example

---

- Objects :  $\mathcal{O} = \{carrot, potato, cabbage, cucumber, orange, lemon, strawberry, apple\}$
- Propositional variables :  $v_1, v_2, v_3$
- carrot : index 0 =  $\langle FFF \rangle \rightsquigarrow \Phi_{carrot} = \neg v_1 \wedge \neg v_2 \wedge \neg v_3$
- potato : index 1 =  $\langle FFT \rangle \rightsquigarrow \Phi_{potato} = \neg v_1 \wedge \neg v_2 \wedge v_3$
- ...
- apple : index 7 =  $\langle TTT \rangle \rightsquigarrow \Phi_{apple} = v_1 \wedge v_2 \wedge v_3$

## Symbolic manipulation of sets : union $\vee$

- $fruits = \{orange, lemon, strawberry, apple\} \subset \mathcal{O}$

$$\begin{aligned}\Phi_{fruits} &= \Phi_{orange} \vee \Phi_{lemon} \vee \Phi_{strawberry} \vee \Phi_{apple} \\ &\equiv (v_1 \wedge \neg v_2 \wedge \neg v_3) \vee (v_1 \wedge \neg v_2 \wedge v_3) \\ &\quad \vee (v_1 \wedge v_2 \wedge \neg v_3) \vee (v_1 \wedge v_2 \wedge v_3) \\ &\equiv v_1 \wedge ((\neg v_2 \wedge \neg v_3) \vee (\neg v_2 \wedge v_3) \vee (v_2 \wedge \neg v_3) \vee (v_2 \wedge v_3)) \\ &\equiv v_1 \wedge \top \equiv v_1\end{aligned}$$

The set of fruits is symbolically represented by :  $v_1$ . (quite compact, isn't it?)

## Symbolic manipulation of sets : intersection $\wedge$

- $yellow = \{potato, lemon\} \subset \mathcal{O}$

$$\begin{aligned}\Phi_{yellow} &= \Phi_{potato} \vee \Phi_{lemon} \\ &\equiv (\neg v_1 \wedge \neg v_2 \wedge v_3) \vee (v_1 \wedge \neg v_2 \wedge v_3) \\ &\equiv \neg v_2 \wedge v_3\end{aligned}$$


Yellow fruits?

$$\begin{aligned}\Phi_{yellow} \wedge \Phi_{fruits} &= (\neg v_2 \wedge v_3) \wedge (v_1) \\ &\equiv v_1 \wedge \neg v_2 \wedge v_3 \equiv \Phi_{lemon}\end{aligned}$$

The only yellow fruit is the lemon.


---

---



# Symbolic diagnosis

---



# Symbolic encoding of a discrete event system : one solution

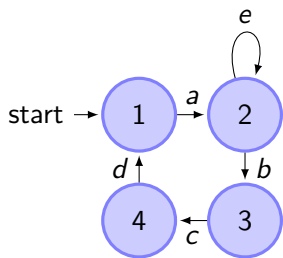
---

- Transition system/Automaton  $A = \langle Q, \mathcal{T}, \mathcal{E}, q_{init} \rangle$
- Three sets to encode : source states, event labels, target states
- Three subsets of propositional variables  $\mathcal{V} = \mathcal{V}_Q \cup \mathcal{V}_{\mathcal{E}} \cup \mathcal{V}'_Q$
- How to represent a transition  $q \xrightarrow{e} q' = \langle q, e, q' \rangle \in \mathcal{T}$  in logic :
  - ▶  $\Phi_q$  : formula that encodes state  $q \in Q$  over  $\mathcal{V}_Q$
  - ▶  $\Phi_{q'}$  : formula that encodes state  $q' \in Q$  over  $\mathcal{V}'_Q$
  - ▶  $\Phi_e$  : formula that encodes an event label  $e \in \mathcal{E}$  over  $\mathcal{V}_{\mathcal{E}}$
  - ▶ transition  $q \xrightarrow{e} q'$  is then represented by the formula :

$$\Phi_q \wedge \Phi_e \wedge \Phi_{q'}$$



## Example : symbolic automaton



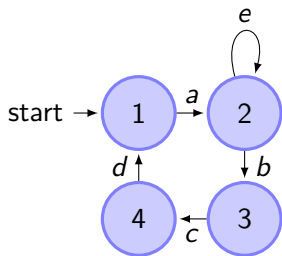
4 states :

1. sources : 2 Boolean variables  $A, B$
2. targets : 2 Boolean variables  $A', B'$

Encoding :

- state 1 : source  $\neg A \wedge \neg B$  / target  $\neg A' \wedge \neg B'$
- state 2 source  $A \wedge \neg B$  / target  $A' \wedge \neg B'$
- state 3 source :  $\neg A \wedge B$  / target  $\neg A' \wedge B'$
- state 4 source :  $A \wedge B$  / target  $A' \wedge B'$

## Example : symbolic automaton



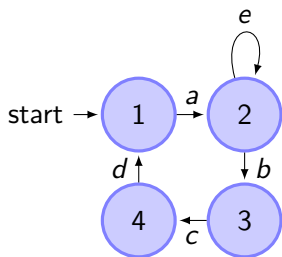
5 events : 3 Boolean variables  $e_1, e_2, e_3$

Encoding :

- $\Phi_a : \neg e_1 \wedge \neg e_2 \wedge \neg e_3$
- $\Phi_b : e_1 \wedge \neg e_2 \wedge \neg e_3$
- $\Phi_c : \neg e_1 \wedge e_2 \wedge \neg e_3$
- $\Phi_d : e_1 \wedge e_2 \wedge \neg e_3$
- $\Phi_e : \neg e_1 \wedge \neg e_2 \wedge e_3$
- 'NotAnEvent' :  $(e_1 \vee e_2) \wedge e_3$



## Example : symbolic automaton



Encoding of the transitions :

- $1 \xrightarrow{a} 2$  : source encoding (1) and event encoding and target encoding(2)

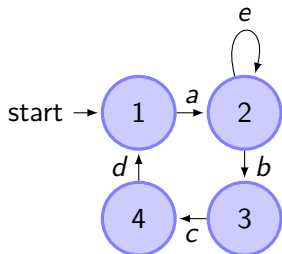
$$\neg A \wedge \neg B \wedge \neg e_1 \wedge \neg e_2 \wedge \neg e_3 \wedge A' \wedge \neg B'$$

- $2 \xrightarrow{e} 2$

$$A \wedge \neg B \wedge \neg e_1 \wedge \neg e_2 \wedge e_3 \wedge A' \wedge \neg B'$$

- etc

## Example : symbolic automaton

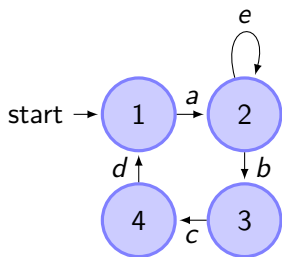


Complete encoding of the automaton :

$\Phi_{aut} \equiv$

$$\begin{aligned} & ((\neg A \wedge \neg B \wedge \neg e_1 \wedge \neg e_2 \wedge \neg e_3 \wedge A' \wedge \neg B') \vee \\ & (A \wedge \neg B \wedge \neg e_1 \wedge \neg e_2 \wedge e_3 \wedge A' \wedge \neg B') \vee \\ & (A \wedge \neg B \wedge e_1 \wedge \neg e_2 \wedge \neg e_3 \wedge \neg A' \wedge B') \vee \\ & (\neg A \wedge B \wedge \neg e_1 \wedge e_2 \wedge \neg e_3 \wedge A' \wedge B') \vee \\ & (A \wedge B \wedge e_1 \wedge e_2 \wedge \neg e_3 \wedge \neg A' \wedge \neg B')) \\ & \wedge \neg((e_1 \vee e_2) \wedge e_3) \end{aligned}$$

# Symbolic operations on automata : transition sets



- How to get the set of transitions labeled by event  $a$ ?


$$\Phi_{aut} \wedge \Phi_a$$

- How to get the set of transitions starting from state  $q_1$ ?

$$\Phi_{aut} \wedge \Phi_{q_1}$$

- How to get the set of transitions starting from state  $q_1$  labelled with  $a$ ?

$$\Phi_{aut} \wedge \Phi_{q_1} \wedge \Phi_a$$



---

# Symbolic operations on automata : firing transitions ?

---

- Requires two specific symbolic operations
  1. Existential abstraction
  2. Variable substitutions

# Symbolic operations : existential abstraction

- Symbolic operation that removes variables from a formula  $\Phi$
- Let  $\mathcal{A} = \{v_{i_1}, \dots, v_{i_n}\} \subset \mathcal{V}$  be  $n$  variables to abstract
- Let  $\alpha \in \mathbb{B}^n$  be a possible assignment of variables in  $\mathcal{A}$  (ex :  $\alpha = \{v_{i_1} \leftarrow T, v_{i_2} \leftarrow F \dots\}$ )
- The restriction of  $\Phi$  by  $\alpha$  :

$$\Phi|_{\alpha} = \Phi[v_{i_1} \leftarrow \alpha(v_{i_1}), v_{i_2} \leftarrow \alpha(v_{i_2}), \dots]$$

- Existential abstraction :

$$exAbst(\Phi, \mathcal{A}) = \bigvee_{\alpha \in \mathbb{B}^n} \Phi|_{\alpha}$$

# Symbolic operations : existential abstraction an example

$$\Phi = (v_1 \wedge v_2) \vee (v_2 \wedge \neg v_3)$$

Existential abstraction of  $\mathcal{A} = \{v_2\}$  :

1. Assignment  $\alpha_1 : \alpha_1(v_2) = T$ 
  - ▶  $\Phi|_{\alpha_1} = (v_1 \wedge T) \vee (T \wedge \neg v_3) \equiv v_1 \vee \neg v_3$
2. Assignment  $\alpha_2 : \alpha_2(v_2) = F$ 
  - ▶  $\Phi|_{\alpha_2} = (v_1 \wedge \perp) \vee (\perp \wedge \neg v_3) \equiv \perp$

$$\begin{aligned} \text{exAbst}(\Phi, \mathcal{A}) &= \Phi|_{\alpha_1} \vee \Phi|_{\alpha_2} \equiv (v_1 \vee \neg v_3) \vee (\perp) \\ &\equiv v_1 \vee \neg v_3 \end{aligned}$$



---

## Symbolic operations : variable substitutions

---


- Let  $\Phi$  be a formula defined over a set of propositional variables  $\mathcal{V}$
- Let  $v, v'$  be two propositional variables ( $v, v'$  can be in  $\mathcal{V}$  or not)
- The substitution of  $v$  by  $v'$  in  $\Phi$  :

$$\Phi[v|v'] = \text{subst}(\Phi, v, v')$$

is the formula  $\Phi$  where any occurrence of  $v$  is syntactically replaced by  $v'$

- Let  $\mathcal{V} = \{v_1, \dots, v_n\}, \mathcal{V}' = \{v'_1, \dots, v'_n\}$

$$\Phi[\mathcal{V}|\mathcal{V}'] = \Phi[v_1|v'_1][v_2|v'_2] \dots [v_n|v'_n] = \text{subst}(\Phi, \mathcal{V}, \mathcal{V}')$$



---

# Symbolic operations : variable substitutions an example

---

$$\Phi = (v_1 \wedge v_2) \vee (v_2 \wedge \neg v_3)$$

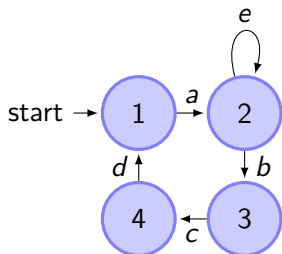
1.  $\Phi[v_1|v_4] = (v_4 \wedge v_2) \vee (v_2 \wedge \neg v_3)$
2.  $\Phi[v_1|v_2] = (v_2 \wedge v_2) \vee (v_2 \wedge \neg v_3) \equiv v_2$
3.  $\Phi[\{v_1, v_2\}|\{v'_1, v'_2\}] = (v'_1 \wedge v'_2) \vee (v'_2 \wedge \neg v_3)$



---

# Symbolic operations on automaton : firing transitions

---



- How to fire  $a$  then  $b$  :

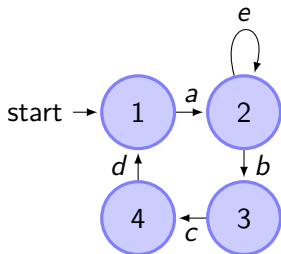
$$\Phi_{aut} \wedge \Phi_a$$

(set of transitions  $a$ )

---

# Symbolic operations on automaton : firing transitions

---

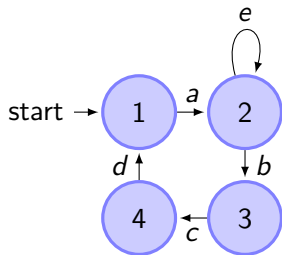


- How to fire  $a$  then  $b$  :

$$exAbst(\Phi_{aut} \wedge \Phi_a, \{A, B, e_1, e_2, e_3\})$$

**set of states  $q'$  such that  $q \xrightarrow{a} q'$   
(target encoding))**

# Symbolic operations on automaton

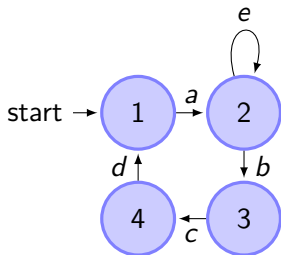


- How to fire  $a$  then  $b$  :

$$\text{subst}(\text{exAbst}(\Phi_{\text{aut}} \wedge \Phi_a, \{A, B, e_1, e_2, e_3\}), \\ \{A', B'\}, \{A, B\})$$

(set of states  $q' \ q \xrightarrow{a} q'$  (source encoding))

# Symbolic operations on automaton



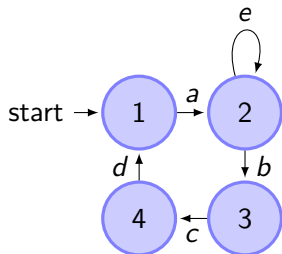
- How to fire  $a$  then  $b$  :

$$\Phi_{aut} \wedge \Phi_b \wedge$$

$$\left( \text{subst}(\text{exAbst}(\Phi_{aut} \wedge \Phi_a, \{A, B, e_1, e_2, e_3\}), \{A', B'\}, \{A, B\}) \right)$$

(set of transitions  $q' \xrightarrow{b} q''$  such that  $q \xrightarrow{a} q'$ )

# Symbolic operations on automaton



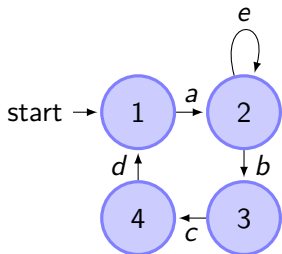
- How to fire  $a$  then  $b$  :

$$exAbs(\Phi_{aut} \wedge \Phi_b \wedge$$

$$(subst(exAbs(\Phi_{aut} \wedge \Phi_a, \{A, B, e_1, e_2, e_3\}), \\ \{A', B'\}, \{A, B\}), \{A, B, e_1, e_2, e_3\}))$$

(set of states  $q''$  such that  
 $q \xrightarrow{a} q' \xrightarrow{b} q''$  (target encoding))

# Symbolic operations on automaton



- How to fire  $a$  then  $b$  :

$subst(exAbst(\Phi_{aut} \wedge \Phi_b \wedge$

$(subst(exAbst(\Phi_{aut} \wedge \Phi_a, \{A, B, e_1, e_2, e_3\}),$   
 $\{A', B'\}, \{A, B\}), \{A, B, e_1, e_2, e_3\}),$

$\{A', B'\}, \{A, B\}))$

(set of states  $q''$  such that

$q \xrightarrow{a} q' \xrightarrow{b} q''$  (source encoding))

# Symbolic operations for diagnosis

- $\Phi_{\mathcal{E}_o}$  represents the observable events. Ex :  $\mathcal{E}_o = \{a, e\}$

$$\Phi_{\mathcal{E}_o} \equiv (\Phi_a \vee \Phi_e) \wedge \neg \text{NotAnEvent}$$

- Belief state after observing an observation from any state of the system

$$\text{exAbst}(\Phi_{\text{aut}} \wedge \Phi_{\mathcal{E}_o}, \{A, B, e_1, e_2, e_3\})$$

- Let  $\Phi_S$  be a set of states, firing all the non-observable transitions at once from  $S$ .

$$\text{exAbst}(\Phi_{\text{aut}} \wedge \neg \Phi_{\mathcal{E}_o} \wedge \Phi_S, \{A, B, e_1, e_2, e_3\})$$

# Symbolic operation Next

Formally

$$\mathcal{S}' = \text{next}_{\mathcal{E}'}(\mathcal{S}) = \{q' \in Q \mid \exists q \in \mathcal{S}, \exists e \in \mathcal{E}', q \xrightarrow{e} q'\}$$

Symbolic computation :

$$\Phi_{\mathcal{S}'} = \text{subst}(\text{exAbst}(\Phi_{\text{aut}} \wedge \Phi_{\mathcal{E}'} \wedge \Phi_{\mathcal{S}}, \mathcal{V}_{\mathcal{Q}} \cup \mathcal{V}_{\mathcal{E}}), \mathcal{V}'_{\mathcal{Q}}, \mathcal{V}_{\mathcal{Q}})$$

Detailed steps of the computation :

1. fire any transitions from  $\mathcal{S}$  labeled with an event of  $\mathcal{E}'$  :  
 $\Phi_{\text{aut}} \wedge \Phi_{\mathcal{E}'} \wedge \Phi_{\mathcal{S}}$
2. keep the target states :  $\text{exAbst}(\dots, \mathcal{V}_{\mathcal{Q}} \cup \mathcal{V}_{\mathcal{E}})$
3. get the source encoding of these targets :  $\text{subst}(\dots, \mathcal{V}'_{\mathcal{Q}}, \mathcal{V}_{\mathcal{Q}})$



# Symbolic silent closure

Formally,

Silent closure of  $S : \{q, \exists q_0 \in S : q_0 \xrightarrow{e_1} \dots \xrightarrow{e_n} q_n = q, e_i \notin \mathcal{E}_o\}$

Symbolic computation :

$$\Phi'_S \leftarrow \Phi_S$$

**Repeat,**

$$\Phi_S \leftarrow \Phi'_S,$$

$$\Phi'_S \leftarrow \text{subst}(\text{exAbst}(\Phi_{\text{aut}} \wedge \neg \Phi_{\mathcal{E}_o} \wedge \Phi_S, \{A, B, e_1, e_2, e_3\}), \\ \{A', B'\}, \{A, B\}) \text{ (next)}$$

**Till**  $\Phi'_S = \Phi_S$ . (fixed point).



---

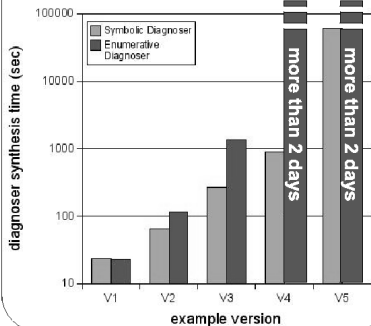
## Some advantages of the symbolic approach

---

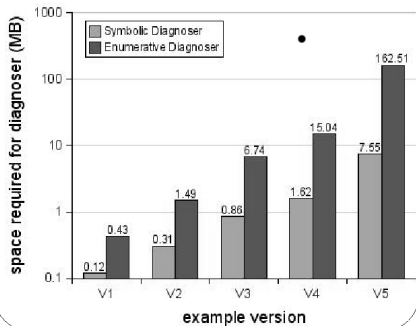
- manipulation of logical formula ( $\wedge \vee$ ) can have polynomial complexity (use of Binary Decision Diagrams or others (Sentential Decision Diagrams, DDNNF)).
- canonical representation of a logic formula (BDD or SDD or DDNNF)
- represents and handles sets of elements in an efficient way ( $A \cap B$  implemented as  $\Phi_A \wedge \Phi_B$ )
- automata composition is just an  $\wedge$ -operator ( $A_1 || A_2$  is roughly implemented as  $\Phi_{A_1} \wedge \Phi_{A_2}$ )

# Symbolic diagnoser : some (old) results

## Computation time for symbolic and enumerative diagnoser

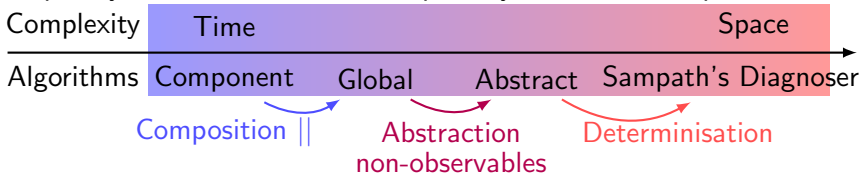


## Size of symbolic and enumerative diagnoser representation



# Spectrum of symbolic algorithms

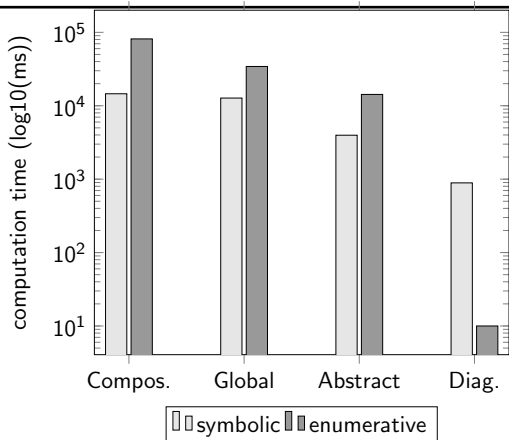
Complexity transfer : from time to space by successive compilations



Some other extensions of these algorithms :

- Decentralised versions.

## Comparative results



Comparison of average time for computing diagnoses on the same model (100 scenarios with 10000 observations each).

---

---



# SAT-based diagnosis

# Diagnosis of DES as Satisfiability problem

- Consider a DES modeled as an automaton  $A$
- Consider a time horizon  $n$  (i.e. the DES performs at most  $n$  operations/transitions)
- One set of state variables per tick :  $\mathcal{V}_Q^1, \dots, \mathcal{V}_Q^{n+1}$
- One set of event variables per tick :  $\mathcal{E}_Q^1, \dots, \mathcal{E}_Q^n$
- A transition  $q \xrightarrow{e} q'$  at tick  $i$  :  $\Phi_{q \xrightarrow{e} q'}^i = \Phi_q^i \wedge \Phi_e^i \wedge \Phi_{q'}^{i+1}$
- No transition at tick  $i$  :  $\Phi_{q \xrightarrow{\emptyset} q}^i = \Phi_q^i \wedge \text{NotAnEvent}^i \wedge \Phi_q^{i+1}$
- One transition at most at tick  $i$  :  $\Phi_{q \xrightarrow{\emptyset} q}^i \oplus \bigoplus_{q \xrightarrow{e} q' \in \mathcal{T}} \Phi_{q \xrightarrow{e} q'}^i$
- A finite run of  $A$  is a conjunction of transition formulas from tick 1 till tick  $n$ .
- Encode the observed run  $\sigma$ .

---

# Diagnosis of DES as Satisfiability problem

---

Finally, you have a formula  $\Phi(A, \sigma, f, n)$

SAT-problems (intuitions) :

1.  $\neg f \wedge \Phi(A, \sigma, f, n)$  SAT? The system may be healthy if yes, certainly not healthy if no.
2. Optimisation problem with SAT :  $\min(f_1 \vee \dots \vee f_k)$  in  $(f_1 \vee \dots \vee f_k) \wedge \Phi(A, \sigma, f, n)$  The returned model is a run consistent with the observations that contains a minimal number of faults.

$k$ -diagnosability can also be solved with SAT.





---

# Summary

---

- Diagnosis of DES : one of the main issue is the combinatorial explosion problem
- Symbolic techniques : encoding automata as logical formula
- Benefit of a factorization effect (knowledge compilation)
- Efficient tools to manipulate logics :
  1. Binary Decision Diagrams
  2. DDNNF, SDD
  3. SAT/SMT solvers

---

# References

---

Book chapter **Discrete-Event Systems Fault Diagnosis** by Alban Grastien and Marina Zanella in Escobet, T., Bregon, A., Pulido, B., Puig, V. (Eds.) *Fault diagnosis of dynamic systems* Springer International Publishing, 2019

## Symbolic diagnosis

1. **A Spectrum of Symbolic On-line Diagnosis Approaches.** Schumann, Pencolé, Thiébaux, AAAI 2007
2. **A decentralised symbolic diagnosis approach.** Schumann, Pencolé, Thiébaux, ECAI 2010
3. **Diagnosis of Discrete Event Systems Using Satisfiability Algorithms : A Theoretical and Empirical Study.** Grastien and Anbulagan IEEE TAC 2013

## Symbolic diagnosability

1. **Symbolic testing of diagnosability.** A. Grastien DX 2009
2. **Symbolic synthesis of observability requirements for diagnosability** B Bittner, M Bozzano, A Cimatti, X Olive, AAAI 2012
3. **Diagnosability of fair transition systems.** B Bittner, M Bozzano, A Cimatti, M Gario, S Tonetta, V Vozarova, AIJ 2022.

## Knowledge compilation

1. **A Knowledge Compilation Map** A. Darwiche and P. Marquis JAIR 2002
2. **Graph-Based Algorithms for Boolean Function Manipulation** R E Bryant 1986
3. **Model-Based Diagnosis with Sentential Decision Diagrams** Torta Shembri DX 2019