Control of a pumping system

Jean-Marc ROUSSEL
jean-marc.roussel@lurpa.ens-cachan.fr

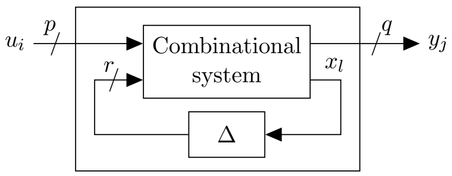
LURPA,ENS Cachan
61, avenue du Prsident Wilson
94235 CACHAN Cedex

April 20, 2012

1 Introduction

This document presents a case study made according to the algebraic synthesis method developed in LURPA. We propose to obtain the control law to implement into a Programmable Logic Controller (PLC) from its specifications given in natural language, by solving a Boolean equations system of switching functions.

We suppose that the expected control law can be expressed with recurrent Boolean equations as presented Figure 1. This generic model has p Boolean inputs (u_i), q Boolean outputs (y_j) and r Boolean state variables (x_l). These inputs and outputs correspond to the inputs and outputs of the controller for which the control laws must be designed. The state variables, used to express the sequential behavior, will be represented with internal variables of the controller.



$$\begin{cases} y_j[k] = F_j(u_1[k], \cdots, u_p[k], x_1[k-1], \cdots, x_r[k-1]) \\ x_l[k] = F_{q+l}(u_1[k], \cdots, u_p[k], x_1[k-1], \cdots, x_r[k-1]) \end{cases}$$

Figure 1: Generic model of sequential systems expressed with recurrent Boolean equations

The behavior of this model can be fully defined according to the definition of $(q+r)$ switching functions of $(p+r)$ variables. Even if this representation is very compact (the r Boolean state variables allow the representation of 2^r different states), the construction by hands of these switching functions has always been a very tedious and error-prone task [Huf54]: the model presented Figure. 1, admits 2^p inputs combinations, can send 2^q outputs combinations and can express $(2^{2^{(p+r)}})^{(q+r)}$ sequential behaviors.

Nevertheless, thanks to recent mathematical results obtained for Boolean algebras [Rud01], [Bro03], the automatic algebraic synthesis of switching functions is now possible. We propose to obtain the control law to implement into a PLC by solving a Boolean equations system of switching functions. Details of the proposed method can be found in [HRL08a] [HRL08b] [Hie09].

To avoid tedious symbolic calculus and to help the designer during the different steps of this synthesis method, a prototype software tool has been developed in Python. This tool[1] performs all the computations required for inconsistencies detection and control laws generation. This enables the designer to focus only on application-related issues. For ergonomic reasons, complementary works were also developed in order be able to represent the synthesized control law with a state model.

[1]Case studies are available: http://www.lurpa.ens-cachan.fr/isa/asc/case_studies.html

1.1 Notations used

To avoid confusion between Boolean variables and Boolean functions of Boolean variables, each Boolean variable b_i is denoted as ${}_bb_i$. The set of the two Boolean values ${}_b0$ and ${}_b1$ is denoted as: $B = \{{}_b0, {}_b1\}$. The classical two-element Boolean Algebra is $(\{{}_b0, {}_b1\}, \vee, \wedge, \neg, {}_b0, {}_b1)$.

Let $F_n(B)$ be the set of the 2^{2^n} n-variable switching functions. The Boolean Algebra of n-variable switching functions is $(F_n(B), +, \cdot, \bar{}, 0, 1)$:

- 0 and 1 are the 2 constant functions:

$$0: \begin{array}{rl} B^n & \to B \\ ({}_bb_1, \ldots, {}_bb_n) & \mapsto {}_b0 \end{array} \qquad 1: \begin{array}{rl} B^n & \to B \\ ({}_bb_1, \ldots, {}_bb_n) & \mapsto {}_b1 \end{array}$$

- $+, \cdot, \bar{}$ are three closed operations defined as follows:

$$\text{Op. } +: \begin{array}{rl} F_n(B)^2 & \to F_n(B) \\ (f,g) & \mapsto f+g \end{array} \qquad \text{Op. } \cdot: \begin{array}{rl} F_n(B)^2 & \to F_n(B) \\ (f,g) & \mapsto f \cdot g \end{array} \qquad \text{Op. } \bar{}: \begin{array}{rl} F_n(B) & \to F_n(B) \\ f & \mapsto \overline{f} \end{array}$$

$$\begin{aligned} &\text{where } \forall ({}_bb_1, \ldots, {}_bb_n) \in B^n, \\ &(f+g)({}_bb_1, \ldots, {}_bb_n) = f({}_bb_1, \ldots, {}_bb_n) \vee g({}_bb_1, \ldots, {}_bb_n) \\ &(f \cdot g)({}_bb_1, \ldots, {}_bb_n) = f({}_bb_1, \ldots, {}_bb_n) \wedge g({}_bb_1, \ldots, {}_bb_n) \\ &\overline{f}({}_bb_1, \ldots, {}_bb_n) = \neg f({}_bb_1, \ldots, {}_bb_n) \end{aligned}$$

$F_n(B)$ can be equipped with a partial order relation, called *Inclusion-Relation* defined as follows:

$$x \leq y \quad \Leftrightarrow \quad x \cdot y = x$$

2 Control system specifications

The studied system is the controller of a water distribution system composed of two pumps which are working in redundancy. The water distribution is made when it is necessary according to the possible failures of elements (the pumps and the distributing system).

2.1 Inputs and outputs of the controller

The Boolean inputs and outputs of this controller are given in Fig. 2. Each pump is controlled thanks to a Boolean output (‘p1’ and ‘p2’). The controller is informed of water distribution requests thanks to the input ‘req’. Inputs ‘f1’ and ‘f2’ inform the controller of a failure of the pumps and ‘gf’ of the global failure of the installation. The input ‘Pr’ precises which pump has priority.

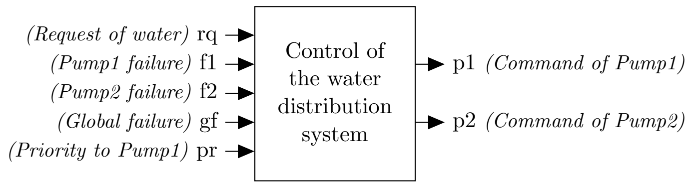


Figure 2: Inputs and outputs of the controller to design

2.2 Expected behavior

The expected behavior of the control system regarding the application requirements can be expressed by the set of assertions given hereafter:

- The two pumps never operate simultaneously.
- A pump cannot operate if it is out of order.
- When a global failure is detected, no pump can operate.
- Pumps can operate only if a water distribution request is present.
- Pumps operate when it is possible. Priority is given according to ‘pr’ (Pump1 has priority when ‘pr’ is true).
- In order to reduce the wear of the pump, it is necessary to restrict the number of starting of the pumps.

2.3 Control laws to design

Our approach does not permit to identify automatically which state variables must be used. They are given by the designer according to its interpretation of the specification.

For the water distribution system, we propose to use two state variables: one for each output. According to this choice, we have only two switching functions to synthesize. However, each function is a 7-variable switching functions as the control laws has 5 inputs and 2 state variables. The generic form of the control law we want to design is:

$$\begin{cases} \mathrm{p1}[k] = \mathrm{P1}\ (\mathrm{rq}[k], \mathrm{f1}[k], \mathrm{f2}[k]\ \mathrm{gf}[k], \mathrm{pr}[k], \mathrm{p1}[k-1], \mathrm{p2}[k-1]) \\ \mathrm{p2}[k] = \mathrm{P2}\ (\mathrm{rq}[k], \mathrm{f1}[k], \mathrm{f2}[k]\ \mathrm{gf}[k], \mathrm{pr}[k], \mathrm{p1}[k-1], \mathrm{p2}[k-1]) \\ \mathrm{p1}[0] = {}_b0 \\ \mathrm{p2}[0] = {}_b0 \end{cases}$$

This simple model permits to express 2^{256} $((2^{2^7})^2)$ different control laws. We propose to find the control law which satisfies the expected behavior given Section 2.2 by solving a Boolean equations system of 7-variable switching functions according to optimization criteria.

3 Algebraic synthesis of the control laws

The first step of the proposed method consists to formalize the expected behavior with relations between formula of 7-variable switching functions. For this case study, we have 7 specific switching functions[2]:

- The 5 switching functions (Rq, F1, F2, GF, and Pr) which characterize the behavior of the inputs of the controller and are defined as follows:

$$\begin{aligned} \mathrm{Rq}: \quad & B^7 \to B \\ & (\mathrm{rq}[k], \ldots, \mathrm{p2}[k-1]) \mapsto \mathrm{rq}[k] \end{aligned}$$

- The 2 switching functions (${}_p\mathrm{P1}$ and ${}_p\mathrm{P2}$) which characterize the previous behavior of the state variables of the controller and are defined as follows:

$$\begin{aligned} {}_p\mathrm{P1}: \quad & B^7 \to B \\ & (\mathrm{rq}[k], \ldots, \mathrm{p2}[k-1]) \mapsto \mathrm{p1}[k-1] \end{aligned}$$

In our case, only 2 switching functions must be designed (P1 and P2). They represent the unknowns or our problem.

Remark: As P1 and ${}_p\mathrm{P1}$ represent the behavior of Pump1 at respectively times $[k]$ and $[k-1]$, the starting of Pump1 corresponds to $(\mathrm{P1} \cdot \overline{{}_p\mathrm{P1}})$ and the stopping of Pump1 corresponds to $(\overline{\mathrm{P1}} \cdot {}_p\mathrm{P1})$.

3.1 Formalization of requirements

Assertions describing the expected behavior of control systems in natural language can be translated into formal statements thanks to the relations Equality and Inclusion.

- The two pumps never operate simultaneously.

$$\mathrm{P1} \cdot \mathrm{P2} = 0$$

- A pump cannot operate if it is out of order.
  - If Pump1 is out of order (F1), Pump1 can not operate:

$$\mathrm{F1} \leq \overline{\mathrm{P1}}$$

  - If Pump2 is out of order (F2), Pump2 can not operate:

$$\mathrm{F2} \leq \overline{\mathrm{P2}}$$

- When a global failure is detected (GF), no pump can operate.

$$\mathrm{GF} \leq (\overline{\mathrm{P1}} \cdot \overline{\mathrm{P2}})$$

- Pumps can operate only if a water distribution request (Rq) is present.

$$(\mathrm{P1} + \mathrm{P2}) \leq \mathrm{Rq}$$

For the last two assertions, the formal statements we propose are three optimization criteria expected for P1 and P2:

- Pumps operate when it is possible.

$$\text{Maximization of: } (\mathrm{P1} + \mathrm{P2})$$

[2]These functions are the 7 projection-functions of $F_7(B)$.

- Priority is given according to 'pr' (Pump1 has priority when 'pr' is true).

$$\text{Maximization of: } ((\mathrm{Pr} \cdot \mathrm{P1}) + (\overline{\mathrm{Pr}} \cdot \mathrm{P2}))$$

- It is necessary to restrict the number of starting of the pumps.

$$\text{Minimization of: } ((\mathrm{P1} \cdot \overline{{}_p\mathrm{P1}}) + (\mathrm{P2} \cdot \overline{{}_p\mathrm{P2}}))$$

Thanks to mathematical results obtained in [Ler11], we are able to obtain automatically the parametric form of the k-tuples solutions of $F_n(B)$ which satisfy not only a given equation ($\mathrm{Eq}(\mathrm{X}_k, \mathrm{Pj}) = 0$) of boolean functions but also maximize (or minimize) a boolean formula of these boolean functions ($\mathcal{F}_\mathrm{C}(\mathrm{X}_k, \mathrm{Pj})$) corresponding to the desired optimization criterion.

3.2 Consistency checking

The result of the formalization of all the requirements is composed of a set of relations for which the solutions must be selected according to three criteria:

$$\left[\begin{array}{l} \textbf{Requirements:} \\ \begin{cases} \mathrm{P1} \cdot \mathrm{P2} = 0 \\ \mathrm{F1} \leq \overline{\mathrm{P1}} \\ \mathrm{F2} \leq \overline{\mathrm{P2}} \\ \mathrm{GF} \leq (\overline{\mathrm{P1}} \cdot \overline{\mathrm{P2}}) \\ (\mathrm{P1} + \mathrm{P2}) \leq \mathrm{Rq} \end{cases} \\ \textbf{Optimization criteria:} \\ \begin{cases} \text{Maximization of: } (\mathrm{P1} + \mathrm{P2}) \\ \text{Maximization of: } ((\mathrm{Pr} \cdot \mathrm{P1}) + (\overline{\mathrm{Pr}} \cdot \mathrm{P2})) \\ \text{Minimization of: } ((\mathrm{P1} \cdot \overline{{}_p\mathrm{P1}}) + (\mathrm{P2} \cdot \overline{{}_p\mathrm{P2}})) \end{cases} \end{array}\right.$$

For this problem, the method we propose permits to prove that the proposed criteria cannot be treated simultaneously as some of them are antagonist (it is not possible to maximize the water distribution if the pumps never start). It is necessary to order given criteria according to their priorities (from maximal priority to minimal).

$$\left[\begin{array}{l} \textbf{Requirements:} \\ \begin{cases} \mathrm{P1} \cdot \mathrm{P2} = 0 \\ \mathrm{F1} \leq \overline{\mathrm{P1}} \\ \mathrm{F2} \leq \overline{\mathrm{P2}} \\ \mathrm{GF} \leq (\overline{\mathrm{P1}} \cdot \overline{\mathrm{P2}}) \\ (\mathrm{P1} + \mathrm{P2}) \leq \mathrm{Rq} \end{cases} \\ \textbf{Ordered Optimization criteria:} \\ \begin{cases} \textbf{1:} \quad \text{Maximization of: } (\mathrm{P1} + \mathrm{P2}) \\ \textbf{2:} \quad \text{Minimization of: } ((\mathrm{P1} \cdot \overline{{}_p\mathrm{P1}}) + (\mathrm{P2} \cdot \overline{{}_p\mathrm{P2}})) \\ \textbf{3:} \quad \text{Maximization of: } ((\mathrm{Pr} \cdot \mathrm{P1}) + (\overline{\mathrm{Pr}} \cdot \mathrm{P2})) \end{cases} \end{array}\right.$$

As the given requirements are consistent, this new version of the problem admits one or more solutions.

3.3 Equation solving

Thanks to mathematical results obtained in [Ler11], we are able to obtain automatically the solution of this problem. This solution is:

$$\begin{cases} \mathrm{P1} = \mathrm{Rq} \cdot \overline{\mathrm{GF}} \cdot \overline{\mathrm{F1}} \cdot (\mathrm{F2} + \mathrm{Pr} \cdot ({}_p\mathrm{P1} + \overline{{}_p\mathrm{P2}}) + {}_p\mathrm{P1} \cdot \overline{{}_p\mathrm{P2}}) \\ \mathrm{P2} = \mathrm{Rq} \cdot \overline{\mathrm{GF}} \cdot \overline{\mathrm{F2}} \cdot (\mathrm{F1} + \overline{\mathrm{Pr}} \cdot ({}_p\mathrm{P2} + \overline{{}_p\mathrm{P1}}) + {}_p\mathrm{P2} \cdot \overline{{}_p\mathrm{P1}}) \end{cases}$$

4 Obtained contol laws

4.1 Representation with recurrent Boolean equations

The control laws presented hereafter was obtained by translating the expression of the unknowns according to the projection-functions into relations between recurrent Boolean equations.

$$\begin{cases} \mathrm{p1}[k] = \mathrm{rq}[k] \cdot \overline{\mathrm{gf}[k]} \cdot \overline{\mathrm{f1}[k]} \cdot (\mathrm{f2}[k] + \mathrm{pr}[k] \cdot (\mathrm{p1}[k-1] + \overline{\mathrm{p2}[k-1]}) + \mathrm{p1}[k-1] \cdot \overline{\mathrm{p2}[k-1]})) \\ \mathrm{p2}[k] = \mathrm{rq}[k] \cdot \overline{\mathrm{gf}[k]} \cdot \overline{\mathrm{f2}[k]} \cdot (\mathrm{f1}[k] + \overline{\mathrm{pr}[k]} \cdot (\mathrm{p2}[k-1] + \overline{\mathrm{p1}[k-1]}) + \mathrm{p2}[k-1] \cdot \overline{\mathrm{p1}[k-1]}) \\ \mathrm{p1}[0] =_b 0 \\ \mathrm{p2}[0] =_b 0 \end{cases}$$

This control law was implemented into a PLC with the Ladder Diagram language [IEC03]. The code is composed of only four rungs (Figure 3).

```
Rung 1: Command of Pump1
|  rq    gf    f1       f2               p1    |
+--| |--|/|--|/|--+--| |----------+--( )--+
|                 |  pr       pp1   |        |
|                 +--| |--+--| |--+        |
|                 |       |  pp2   |        |
|                 |       +--|/|--+        |
|                 |  pp1      pp2   |        |
|                 +--| |-----|/|--+        |
|                                            |

Rung 2: Command of Pump2
|  rq    gf    f2       f1               p2    |
+--| |--|/|--|/|--+--| |----------+--( )--+
|                 |  pr       pp2   |        |
|                 +--|/|--+--| |--+        |
|                 |       |  pp1   |        |
|                 |       +--|/|--+        |
|                 |  pp2      pp1   |        |
|                 +--| |-----|/|--+        |
|                                            |

Rung 3: Update previous value of Pump1
|  p1                                  pp1  |
+--| |-------------------------------( )--+
|                                           |

Rung 4: Update previous value of Pump2
|  p2                                  pp2  |
+--| |-------------------------------( )--+
|                                           |
```

Figure 3: Ladder Diagram of the code to implement into the PLC

4.2 Representation with a state model

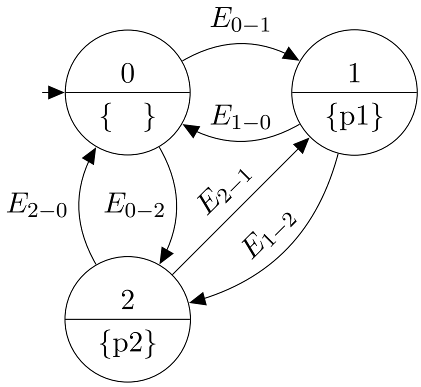
If recurrent Boolean equations are well-adapted for an implementation, the representation of the control law with a state model simplifies the work of the designer. For this control law, the equivalent state model (automatically built thanks to [Gui11]) is composed of three states only (Figure 4). Each state is defined according to the set of emitted outputs. The six transition conditions are a Boolean expression of the inputs. By construction, this state model satisfies all the requirements given Section 2.2.

5 Conclusions

For this case study, the method we propose has allowed to find the control law for the pumping system. The use of optimization criteria has simplified greatly the formalization of the requirements.

References

[Bro03] Frank Markham Brown. *Boolean Reasoning: The Logic of Boolean Equations*. Dover Publications, 2003.



$$\begin{cases} E_{0-1} = \mathrm{rq} \wedge \neg \mathrm{gf} \wedge \neg \mathrm{f1} \wedge (\mathrm{f2} \vee \mathrm{pr}) \\ E_{0-2} = \mathrm{rq} \wedge \neg \mathrm{gf} \wedge \neg \mathrm{f2} \wedge (\mathrm{f1} \vee \neg \mathrm{pr}) \\ E_{1-0} = \neg \mathrm{rq} \vee \mathrm{gf} \vee \mathrm{f1} \wedge \mathrm{f2} \\ E_{1-2} = \mathrm{rq} \wedge \neg \mathrm{gf} \wedge \mathrm{f1} \wedge \neg \mathrm{f2} \\ E_{2-0} = \neg \mathrm{rq} \vee \mathrm{gf} \vee \mathrm{f1} \wedge \mathrm{f2} \\ E_{2-1} = \mathrm{rq} \wedge \neg \mathrm{gf} \wedge \mathrm{f2} \wedge \neg \mathrm{f1} \end{cases}$$

Figure 4: State model of the obtained control law

[Gui11] Anaïs Guignard. Symbolic generation of the automaton representing an algebraic description of a logic system. Master's thesis, ENS Cachan, July 2011.

[Hie09] Yann Hietter. *Synthèse algébrique de lois de commande pour les systèmes à évènements discrets logiques*. PhD thesis, ENS Cachan, May 2009.

[HRL08a] Yann Hietter, Jean-Marc Roussel, and Jean-Jacques Lesage. Algebraic synthesis of dependable logic controllers. In *Proceedings of 17th IFAC World Congress, 2008*, pages pp. 4132–4137, Seoul, South Korea, July 2008.

[HRL08b] Yann Hietter, Jean-Marc Roussel, and Jean-Jacques Lesage. Algebraic synthesis of transition conditions of a state model. In *Proceedings of 9th International Workshop On Discrete Event Systems (WODES'08)*, pages 187–192, Göteborg, Sweden, May 2008.

[Huf54] D. A. Huffman. The synthesis of sequential switching circuits. *J. of the Franklin Institute*, 257(3-4):161–190 and 275–303, 1954.

[IEC03] IEC 61131-3. *IEC 61131-3 Standard: Programmable controllers - Part 3: Programming languages.* International Electrotechnical Commission, 2 edition, 2003.

[Ler11] Hélène Leroux. Algebraic synthesis of logical controllers with optimization criteria. Master's thesis, ENS Cachan, July 2011.

[Rud01] Sergiu Rudeanu. *Lattice Functions and Equations (Discrete Mathematics and Theoretical Computer Science)*. Springer, 2001.