Available online at www.sciencedirect.com

ScienceDirect

IFAC-PapersOnLine 49-12 (2016) 402–407

CSP solver for Safe PLC Controller: Application to manufacturing systems

R. PICHARD*, N. BEN RABAH*, V. CARRE-MENETRIER* and B. RIERA*

** CReSTIC (EA3804), UFR Sciences Exactes et Naturelles, Reims University (URCA), Moulin de la Housse, BP 1039, 51687 Reims - France (bernard.riera@univ-reims.fr).*

Abstract: This paper presents an original approach of safe control synthesis for manufacturing systems controlled by Programmable Logic Controller (PLC) based on the use of a CSP (constraint satisfaction problem) solver. In this work, manufacturing systems are considered as Discrete Event Systems (DES) with logical Inputs (sensors) and logical Outputs (actuators). The proposed approach separates the functional control part from the safety control part. The methodology is based on the use of safety constraints in order to get from a CSP solver all the safe outputs vectors at each PLC scan time. The safe outputs vector is selected by choosing the one which minimizes the Hamming distance with the functional outputs vector. The approach is illustrated with a sorting boxes simulated process using the ITS PLC software from the Real Games Company (www.realgames.pt).

***Keywords**:* Discrete-Event Systems, CSP, Control, Safety, Programmable Logic Controllers, Manufacturing Systems.

1. INTRODUCTION

This paper presents an original approach of control synthesis for manufacturing systems controlled by PLC (Programmable Logic Controller). In this work, manufacturing systems are considered as Discrete Event Systems (DES) (Cassandras *et al.,* 1999) with logical Inputs (sensors) and logical Outputs (actuators). This is an extension of the research work that the CReSTIC (Research Center in Information and Communication Science and Technologies) has led for several years on the definition and design of guard conditions (also called constraints) placed at the end of the PLC program which act as a logic filter in order to be robust to control errors. These safety constraints can be formally checked off line by using a model checker (Marangé *et al.,* 2010). This idea has been extended to propose a safe control design pattern based on safety logical constraints. This approach, which separates the functional control part from the safety control part, is easy to implement and involve a new way to design the controller. The methodology is based on the use of safety constraints in order to get the most permissive safe controller allowed by the safety constraints set. This controller is then constrained by functional constraints while respecting the safety constraints (Riera *et al.,* 2014, 2015). In this paper, we propose a new approach for the control synthesis algorithm. The idea is to use at each PLC scan time a CSP (Constraint Satisfaction Problem) solver to get the set of all outputs vectors respecting the set of safety constraints and to select the one which is the closest in the sense of Hamming distance of the functional outputs vector. Hence, the controller continues to work with safe outputs values. This approach to PLC programming makes safety a priority and allows for a controller to create a safe environment where functional and safety aspects are clearly separated. Compared to the algorithm already proposed, the approach using a CSP solver does not require to define priorities between outputs when a combined safety constraint is violated. The first part of the paper is dedicated to the concept of robust logic filter to control errors. In the second part, the definition and mathematical formalism used for the safety guards are detailed. The third part presents the control algorithm using a CSP solver. At least, the approach is illustrated by using one example: a virtual sorting system using the ITS PLC software from the Real Games Company (www.realgames.pt). Today, PLC does not include CSP solver. To test the idea, a soft PLC written in IronPython and using logilab-constraint, an open source constraint solver (https://www.logilab.org/project/logilab-constraint) written in pure Python controller, has been designed. This example shows the interest in terms of simplicity and efficiency of this original control synthesis method. It seems to be the first time that a CSP solver is used in real time as a part of a PLC program to get a safe controller.

2. LOGICAL GUARDS FOR SAFE PLC PROGRAM

Since a PLC is a dedicated controller it will only process this one program over and over again. One cycle through the program is called a scan time and involves reading the inputs (i) from the other modules (input scan), executing the logic based on these inputs (logic scan) and then updated the outputs (o) accordingly (output scan). The memory in the CPU stores the program while also holding the status of the I/O and providing a means to store values. A controller at each PLC scan time has to compute the outputs values (controllable variables) based on inputs (uncontrollable variables) and internal memories. The use of a memory map

2405-8963 © 2016, IFAC (International Federation of Automatic Control) Hosting by Elsevier Ltd. All rights reserved.
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2016.07.638

enables to guarantee that all the calculations are performed with inputs values which are not modified during a PLC scan time. Outputs update is performed with the last outputs calculation in the PLC program. These three basic stages of operations (input scan, logic solve and output scan are repeated at each scan time (cf. figure 1).

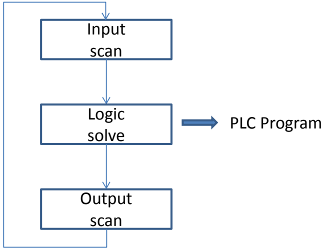

Fig. 1. PLC operation sequence

The idea proposed by (Marangé *et al.,* 2010) is to place a logic filter between the logic solve and the output scan. The goal of this filter is to detect and compensate control errors (cf. figure 2).

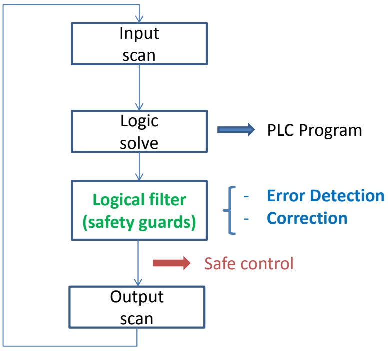

Fig. 2. Principle of the logic filter

Three use cases can be thought of doing with the logic filter: safe blocking, supervisor and controller. In the first case, when a safety constraint is violated, the controller is frozen in a safe state which is supposed known. The supervisory approach consists in correcting the control errors without blocking the controller. This enables for instance to safe existing PLC program without changing the code. The controller approach is similar to the supervisor approach. The main difference is that in the design of the controller, it is taken into account by the designer that the safety part is managed by the safety constraints. Hence, there is a separation between functional and safety aspects of the controller. In addition, even if the functional part is badly defined, the system remains safe (Riera *et al.*, 2015). Contrary to the supervisor approach, the fact to violate a safety constraint can be seen as normal (cf. figure 3).

This approach modifies the way to design a PLC program but presents several advantages (tasks synchronization, management of running modes, connection to a Manufacturing Execution System …). Control design based on logical constraints involve 2 main difficulties:

1) Constraints definition and validation which are not going. We suppose in this paper that the designer has got a correct set of safety constraints.

2) The proposal of a control algorithm which defines, when one or several constraints are violated, a safe outputs vector compliant with all the safety constraints.

We have already proposed an algorithm to compute at each PLC scan time a safe outputs vector (Riera *et al.*, 2015). The idea in this paper is to propose a new approach based on CSP to perform the detection and the correction stages. The main advantages of this new approach come from the fact that it is simpler than the previous algorithm because it is not necessary to define priority between outputs when a combined safety constraint is violated. The idea is to use at each PLC scan time a CSP solver to get the set of all outputs vectors respecting the set of safety constraints and to select the one which is the closest in the sense of Hamming distance of the functional outputs vector. Indeed, the functional part of the PLC program aims at reaching the production goals. The idea is to select the safe outputs vector which is the most similar to the functional outputs vector.

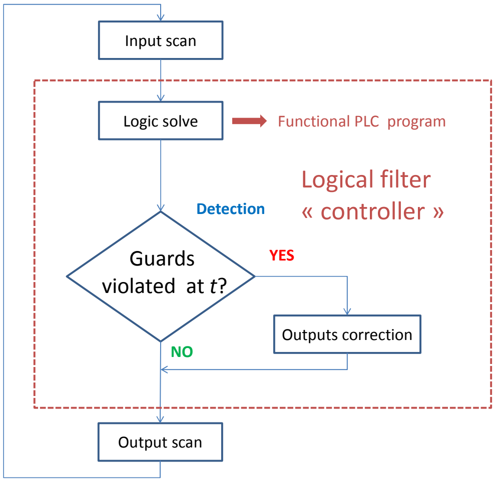

Fig. 3. Logic filter as a controller

2. BOOLEAN SAFETY CONSTRAINTS FORMALISM

The notations used in the following paper are:

- *t*: current scan time (from PLC point of view), *t-1* previous PLC scan time.
- $o_k = o_k(t)$: logical variable corresponding to the value of k^{th} PLC boolean output (actuator) at *t*. Outputs at *t* are considered as the one and only variables that can be controlled (write variables) at each PLC scan time. All other PLC variables (inputs, previous outputs …) are uncontrollable (read only variables).
- $o_k^* = o_k(t-1)$: logical variable corresponding to the value of k^{th} PLC boolean output (actuator) at time *t-1* (previous PLC scan time).
- “.”, “+”, “$\oplus$” “$\overline{\ }$” are respectively the logical operators AND, OR, XOR and NOT.
- 0 means *False* and 1 means *True*.
- $\uparrow x$ and $\downarrow x$ means respectively rising edge and falling edge of boolean variable x (in the PLC, $\uparrow x = \overline{x^*}.x, \downarrow x = x^*.\bar{x}$).
- $\sum$ and $\prod$ are respectively the logical sum (OR) and the logical product (AND) of logical variables.
- $\sum\prod$ is a logical polynomial (sum of products expression also called SIGMA-PI).
- O: set of output variables at *t*.
- Y: set of uncontrollable variables at *t, t-1, t-2* ...