Symbolic Algebraic Discrete Systems Theory – Applied to a Fighter Aircraft*

Jonas Plantin Johan Gunnarsson Roger Germundsson

Department of Electrical Engineering

Linköping University

S-581 83 Linköping, Sweden

{plantin, roger, johan}@isy.liu.se

<http://www.control.isy.liu.se>

Submitted to the 34th Conference on Decision and Control.

1995

Pages: 1863–1864

Abstract

Symbolic algebraic modeling and analysis techniques for DEDS are applied to the landing gear subsystem in the new Swedish fighter aircraft, JAS 39 Gripen. Our methods are based on polynomials over finite fields. Polynomials are used to represent the basic dynamic equations for the processes (controller and plant) as well as static properties of these. Temporal algebra (or temporal logic) is used to represent specifications of system behavior. We use this approach to model the landing gear controller from the complete implementation in Pascal. We also provide temporal algebra interpretations of the specifications made available to us. Finally we perform a number of symbolic analyses on the complete process (controller and plant). This project is a first demonstration of possible uses of these methods and tools and it shows that these methods and tools scale to problems of a non trivial size, i.e. of the size found in complex system designs such as the JAS 39.

1 Introduction

The interest in discrete event systems (DEDS) has increased during the last years, due to the lack of methods and tools that are capable to handle the complexity of problems and tasks present in industry today. To explore the usefulness of symbolic and algebraic methods, we use polynomials over finite fields (see section 2) applied to DEDS with industrial sized complexity: The landing gear controller (LGC) of the Swedish fighter aircraft JAS 39 Gripen.

*This work was supported by the Swedish Research Council for Engineering Sciences (TFR) and the Swedish National Board for Industrial and Technical Development (NUTEK), which is gratefully acknowledged.

The purpose of the LGC is to perform maneuvers of the landing gears and the corresponding doors which enclose the gears in retracted position. The controller is a software process that interacts with 5 binary actuators, 30 binary landing gear sensors, 2 binary pilot signals, and 5 integer mode signals from other subsystems in the aircraft. The only formal description of the controller is a 1200 line Pascal code.

This paper gives an overview of the project of doing static and dynamic analysis on the behavior of the LGC. This was made possible by modeling the LGC by a polynomial, i.e. compiling the Pascal implementation of the LGC to a polynomial relation. For a complete description of this project see [1, 3, 4, 2].

2 The Polynomial Framework

Quantities and relations in DEDS are of a finite nature and can therefore be represented by finite relations. These relations are in turn represented mathematically by polynomials over finite fields $\mathbb{F}_q[Z]$, i.e. polynomials of variables in the set Z with coefficients from a finite field $\mathbb{F}_q$. By further restricting the class of polynomials we construct a quotient polynomial ring (see [1]) that gives a one to one correspondence between polynomials and relations as well as a compact representation of the relations.

The computational framework used for manipulating polynomials is based on *binary decision diagrams* (BDD), which give a powerful representation as well as fast computations which allow us to manipulate rather complex systems.

3 Modeling

As mentioned in the introduction we build a polynomial model from the Pascal code. The polynomial model is denoted $M(z, z^+)$, where z and z^+ are the system variables¹ for present and next time instant respectively.

The Pascal code, representing the LGC, is executed once every sample, and the code represents a state space form of the LGC. Thus we need to analyze the code to determine what variables are inputs and outputs of the entire program. Variables that are both output and input variables have to be state variables. Other topics in the global analysis of the code are temporary variables and timers. The maximum range of the integer variables is determined to $0, 1, \dots, 15$ which makes it possible to represent each integer variable by four Boolean variables.

The translation from Pascal to Boolean expressions² follows the control flow graph of the program. The value of each program expression is determined by the current values of symbols and the actual program expression, i.e. the compilation function is of the form:

$$\Phi : Pascal \times State \rightarrow State$$

We store the current state of the program as a symbol table of the form:

$$\sigma = \{v_1 \mapsto e_1, \dots, v_n \mapsto e_n\}$$

¹Input, state and output variables.

²Boolean expressions are essentially polynomials over the field $\mathbb{F}_2$.

where each v_i is a variable or symbol and each e_i is a Boolean expression of input variables or the symbol $\perp$ indicating undefined values. The symbol table σ is initiated by variables that acts as place holders for the input, and by $\perp$ for the output variables. The symbol table is then updated by traversing the control flow graph of the Pascal code.

Suppose we have the Pascal expression

$$pe = \left(\begin{array}{l} \text{IF } q \text{ THEN} \\ \quad y1 := c \\ \text{ELSE} \\ \quad \text{BEGIN} \\ \quad \quad y1 := d; \\ \quad \quad y2 := e \\ \quad \text{END;} \end{array} \right)$$

with the initial symbol table

$$\sigma = \{q \mapsto q^*, c \mapsto c^*, d \mapsto d^*, e \mapsto e^*, \\ y1 \mapsto y1^*, y2 \mapsto y2^*\}$$

we will get

$$\sigma^+ = \Phi(pe)\sigma = \{q \mapsto q^*, c \mapsto c^*, d \mapsto d^*, \\ e \mapsto e^*, y1 \mapsto (q^* \wedge c^*) \vee (\neg q^* \wedge d^*), \\ y2 \mapsto (q^* \wedge y2^*) \vee (\neg q^* \wedge e^*)\}$$

The final Boolean relation is computed from the final symbol table

$$\sigma_{final} = \{x^+ \mapsto f(x, u), y \mapsto g(x, u)\} \\ M(z, z^+) = x^+ \leftrightarrow f(x, u) \wedge y \leftrightarrow g(x, u)$$

where $z = [x, y, u]$.

The resulting relation for the LGC has 26 state variables and the relation $M(z, z^+)$ has 105 variables altogether. The size of the relation is approximately 320 000 nodes as a BDD and takes approximately 35 minutes to compute on a regular workstation.

4 Analysis

We use the relation $M(z, z^+)$ to analyze the LGC behavior in a number of ways. First we compute the set of reachable states in the LGC. This set is represented algebraically by a relation $R(x)$. The number of reachable states turns out to be 10 015 which is far below the possible number which is $2^{26} \approx 10^8$. We can restrict the original relation as

$$\hat{M}(z, z^+) = R(x) \wedge M(z, z^+) \wedge R(x^+)$$

which gives a significantly simpler relation.

The static analysis of $\hat{M}(z, z^+)$ is performed by adding constraints $P(u)$ to the inputs of the LGC, and then analyze what effect this gives to the outputs.

Results on dynamic closed loop analysis is not available yet. However we use the same tools as to compute the set of reachable states. The specifications of the behavior are represented by temporal logic expressions, used together with the model to compute e.g. the set of behaviors that might reach a forbidden state in the future.

References

- [1] Roger Germundsson. *Symbolic Systems - Theory, Computation and Applications*. PhD thesis, Linköping University, September 1995.
- [2] Roger Germundsson, Johan Gunnarsson, and Jonas Plantin. Symbolic algebraic discrete systems - applied to the JAS 39 fighter aircraft. Technical Report LiTH-ISY-R-1718, Department of Electrical Engineering, Linköping University, S-581 83 Linköping, Sweden, December 1994. Available through ftp at <ftp://ftp.control.isy.liu.se/pub/Reports/1995/1718.ps.Z>.
- [3] Johan Gunnarsson. On modeling of discrete event dynamic systems, using symbolic algebraic methods. Technical Report LiU-TEK-LIC-1995:34, Dept. of Electrical Engineering, Linköping University, S-581 83 Linköping, Sweden, June 1995. Available through WWW at <ftp://ftp.control.isy.liu.se/pub/Reports/LicentiateThesis/Lic502.ps.Z>.
- [4] Jonas Plantin. Algebraic methods for verification and control of discrete event dynamic systems. Technical Report LiU-TEK-LIC-1995:33, Dept. of Electrical Engineering, Linköping University, June 1995.