Application of SMT solvers for evaluation of Real-Time control logic of spacecraft

A A Tyugashev

1Samara State Transport University, Svoobody str. 2V, Samara, Russia, 443066

e-mail: tau797@mail.ru

Abstract. SMT solvers are promising intelligent software successfully applied today in various areas. The paper is describes how the functionality of SMT solver can be utilized for spacecraft's control logic key parameters’ evaluation. Use of SMT in spacecraft control problem domain is based on real-time control calculus developed by A.A. Kalentyev, and the semantic model of real-time control logic. A formal specification of real-time control is feasible or non-feasible on the defined basis of functional processes, this is depends on their parameters’ values, including duration of executed process. We check the feasibility using SMT solver. The example in the paper demonstrates effectiveness call of Z3 SMT solver from specially developed Java application.

1. Introduction

The modern technical object such as airplane, submarine, spacecraft, nuclear power station can be reviewed as ‘system of the systems’ which includes a lot of subsystems, actuators, sensors, devices. All of these devices should co-function in harmonic manner to produce a useful outcome like an orchestra playing symphony. Each instrument must start play at a right time. In orchestra, the control functions are executed by a conductor. In modern complex technical complexes, the control system should provide that functionality. The human could be involved in the process in case of automated control, or not be involved in case of automatic system. Moreover, in according to Ahby’s Law of Requisite Variety [1], “Variety absorbs variety”, so the complexity of control system should be adequate to a complexity of controlled object. Control algorithms must implement the right ‘control logic’, i.e. coordinated functioning of the all units. The ‘coordinated’ word means both a semantic coordination related to physical restrictions and logic of actions and coordination in time. The time characteristics of the control logic should be adequate to the speed of ongoing physical processes associated with the controlled technical complex [2-5].

The very important problem for control logic of complex technical object is evaluation of its parameters and checking if these values are correspond to existing physical and technological constraints. This problem is quite actual both at design stage when the key question is feasibility of the required system and even during operation of the existing technical object when we need to analyze performance, for example. This paper is focused on timing (synchronization) parameters, and use of accessible resources (level of workload/overload). The problem has an additional importance due to its straight connection to dependability/safety issues of spaceflights.

Today, as a rule, the evaluation process is being performed by a human. Unfortunately, the number of parameters of real modern spacecraft must be analyzed, for example, can be very big and exceed the human opportunities.

So, it is a potentially useful approach to apply existing automation tools to provide some assistance to specialists who are responsible for control logic evaluation [6-7]. The very popular and promising
technology today is Satisfiability Modulo Theories (SMT) approach supported by a lot of commercial and free solvers such as ABSolver, Alt-Ergo, Barcelogic, MathSAT, CVC, OpenSMT, Simplify, STeP, Yices, Z3, etc. We can specify the existing constraints using smt-lib formal language, and then get the answer if the system satisfies (sat) the constraints, or not (unsat). The system even can calculate the values of the variables which provide satisfiability.

2. Method
We will consider core staff of the paper in this section. First subsection is devoted to mathematical modeling of Real-Time control Logic, second subsection is about practical ways of utilization of SMT solvers functionality, and third subsection describes developed prototype software tool.

2.1. Mathematical model of Real-Time Control Logic
In previous papers [3,6,7] author proposed the semantic model for real-time control algorithms. The model represents control actions by the set of following:

\[ RTCL = \{< f_i, t_i, \tau_i, l_i > \}, i=1..N \] (1)

\( f_i \) represents an ID of functional process (FP) to be executed, and: \( t_i \) – time of \( f_i \) begin (non-negative integer), \( \tau_i \) – its duration (non-negative integer). \( l_i \) is a ‘logical vector’ defining if the process should be executed. The logical vector is formed by the logical variables with its values: \( \alpha_1=0, \alpha_2=1, \alpha_3=0, \alpha_4=H, \alpha_5=H \). Herewith, 1 and 0 corresponds to True and False, and ‘H’ value means that execution of the process is not depends on value of this logical variable. The presence of logical variables in the model allows specifying a set of options of implementation of the algorithm (including normal and abnormal situations).

Some parameters can be specified by a known constants, some be initially unknown and stated as variables.

The constraints and requirements for the Real-Time control logic can be specified using language of CA formal theory (calculus of real-time control algorithms) proposed by A.A. Kalentyev [3-4]. The extended version of this theory developed by author [6] contains the following operators allowed for use in specification (see Table 1). There are also ‘soft’ bindings: \(<, << \) and \(<>\). Special operator \(<>\) means logical incompatibility of actions, i.e. the processes cannot be found in the same case of execution. This is means that the same logical variables has 1 value in one vector, and 0 in another.

| Name | Mean | Signature |
|------|------|-----------|
| CH   | ‘begin-begin’ synchronization | \((UA_1, UA_2) \rightarrow UA\) |
| CK   | ‘end-end’ synchronization | \((UA_1, UA_2) \rightarrow UA\) |
|     | direct following | \((UA_1, UA_2) \rightarrow UA\) |
| \(H\) | parameterized overlay | \((UA_1, UA_2, integer) \rightarrow UA\) |
| \(3A\) | parameterized following | \((UA_1, UA_2, integer) \rightarrow UA\) |
| \(@\) | absolute time binding | \((UA_1, integer) \rightarrow UA\) |
| \(\Rightarrow\) | qualification by the logical condition | \((condition, UA) \rightarrow UA\) |

These formal calculi are strong associated with algebraic models or real-time control algorithms [6].

We focus on synchronization of functional processes to be executed. The synchronization of two processes can be expressed by following operators: coincidence by begin (named CH from Russian abbreviation), coincidence by end (named CK), direct following (→), time uncrossing (<>), precedence (<), strict precedence (<<), the overlap with the specified shift (H), parameterized following with the specification of the delay (3A).

The sense of operators becomes quite clear after looking at Figures 1-6.
The IV International Conference on Information Technology and Nanotechnology

IOP Publishing

IOP Conf. Series: Journal of Physics: Conf. Series 1096 (2019) 012156
doi:10.1088/1742-6596/1096/1/012156

Figure 1. Coincidence ‘begin-begin’ CH.

Figure 2. Coincidence ‘end-end’ CK.

The operators: <, << and <> expressed ‘soft’ bindings where times of processes’ begins and ends may vary in some intervals.

Figure 3. Direct following.

Figure 4. Parameterized following.

Figure 5. Strict precedence.

Special operator <l> means logical incompatibility of actions, i.e. the processes cannot be found in the same case of execution. This is means that the same logical variable has value 1 in one vector, and 0 in another.
As is was presented in [7], in some cases with the preset values of the model variables, wanted specification of the control actions may be feasible, but the same set of requirements and constraints may be unfeasible with the other values.

Example 1. For the following synchronization requirements:

\[
\begin{align*}
    & f_1 \rightarrow f_2 \\
    & f_1 \rightarrow f_3 \\
    & f_4 \rightarrow f_5 \\
    & f_3 \rightarrow f_4 \\
    & f_2 \rightarrow f_5 
\end{align*}
\]

and values \( \tau_1 = 20 \), \( \tau_2 = 100 \), \( \tau_3 = 200 \), \( \tau_4 = 10 \), \( \tau_5 = 50 \), the specification is not feasible due to impossibility of fulfillment of \( f_2 \rightarrow f_5 \) formulae, this fact can be visually verified in Figure 7.

But if we have the another parameters, for example, \( \tau_1 = 100 \), \( \tau_2 = 150 \), \( \tau_3 = 70 \), \( \tau_4 = 10 \), \( \tau_5 = 50 \), specification becomes feasible (see Figure 8).
The problem of satisfiability of required specification on the defined basis of FPs with the fully known parameters is quite important and interesting. At the same time, even more interesting are two significant issues:

1) For the known specification and partially defined parameters of the functional processes, such as starting time, duration and logical conditions, to check if the specification is satisfiable - in general.

2) Find out the parameters of the FPs, which are satisfy the defined specification – in other words, build the needed algorithm.

The very important point is that the model above can be applied not only for real-time onboard flight control software it was initially developed for, but for representation of any sort of activity/processes performed by human, technical devices, processors, mechanisms, etc. Indeed, the presented model is independent on nature of implementer of functional process. On the other hand, the model allows representing adequately complexity of Real-Time control actions in ‘time space’ and ‘logical space’.

2.2. *Ways of utilization of SMT solvers functionality*

The most common of mathematical objects and constructs such as rational and real numbers, vectors, and matrix are already supported by existing SMT solvers. So, if we will transform requirements to real-time control logic into requirements to these kinds of entities, we will have possibility to utilize functionality of SMT solver.

To do this we may use following way from the formulas and relations between the functional processes to equations and inequalities on the numbers.

**Table 2.** Associations between the control logic requirements and numbers relate in equalities and equations.

| RTCL formulae | entail | comment |
|---------------|--------|---------|
| \( f_i \land H(f_j, \delta) \) | \( t_i + \delta = t_j \) | equation of numbers |
| \( f_i \rightarrow f_j \) | \( t_i + \tau_i = t_j \) | equation of numbers |
| \( 3A(f_i, f_j, \delta) \) | \( t_i + \tau_i + \delta = t_j \) | equation of numbers |
| \( f_i \ll f_j \) | \( t_i < t_j \) | inequality of numbers |
| \( f_i \ll< f_j \) | \( t_i + \tau_i < t_j \) | inequality of numbers |
| \( f_i \lll f_j \) | \( t_i + \tau_i < t_j, V t_j + \tau_j < t_i \) | disjunction of inequalities |

This way allows us to formulate in notions of smt-lib language.

2.3. *Program implementation*

Many of the free accessible SMT solvers provide API for integration with user developed software. Moreover, Z3, for example, can be executed online via Internet. It is allowed us to utilize SMT solver functionality for practical goals in control logic problem domain. Special program prototype was developed to validate applicability of the describing approach.

The prototype written using Java 8 and has intuitive graphical user interface (see Figure 3, in Russian).

Many of the free accessible SMT solvers provide API for integration with user developed software. Moreover, Z3, for example, can be executed online via Internet. It is allowed us to utilize SMT solver
functionality for practical goals in control logic problem domain. Special program prototype was
developed to validate applicability of the describing approach.

The prototype written using Java 8 and has intuitive graphical user interface (see Figure 9, in Russian).

![Figure 9. Screenshot of developed program prototype.](image)

User can set known values of model variables in dedicated editor fields. There are also special field
on form for specification of requirements in terms of control logic, buttons for solving (trying to
evaluation of feasibility – sat or unsat, and finding parameters which are make the set of requirements
feasible).

![Figure 10. Graphical editor of the cyclogram (schedule).](image)

3. Conclusion and Future Work
We have shown how the algebraic and logical based models of real-time control logic can be applied
for feasibility checking using Satisfiability Modulo Theories solvers. The Real Time Control Logic
presented in the paper, is a product of evolution of ideas formulated by A.A. Kalentyev in his early formal calculus of control algorithms, and algebra of real-time control algorithms. The proposed approach uses transformation of formal specification represented in terms of RTCL formulas, into equations and inequalities with integers. Then we convert them in SMT solver compatible smt-lib language. The paper presents prototype of software tool supporting the approach and based on calling Z3 SMT solver through its application programming interface.

One more prospective direction of future researches and development is introduction of graphical notation which is intuitive understandable for users who are not a professional programmers. This notation should be designed for the representation of RTCA with taking in account its features. The prospective visual tool can show specification of RTCA as well as a set of cyclogrammes of various variants of control algorithm associated with different values of logical conditions in logical vector, and then call the SMT solver for checking the satisfiability of the specification as it shown in Figure 10.

4. References
[1] Ashby W R 1956 Introduction to Cybernetics (New York: Chapman & Hall)
[2] Sollogub A V Akhmetov R N and Makarov V P 2012 Principles of the Earth Observation Satellites Control in Contingencies Information and Control Systems 1 16-22
[3] Tyugashev A A 2006 Integrated environment for designing real-time control algorithms Journal of Computer and Systems Sciences international 2(45) 287-300
[4] Kalentyev A A and Tyugashev A A 2006 Application of CALS technologies in Lifecycle of Complex Control Software (Samara: Samara Centre of RAS Publishing, in Russian).
[5] Tyugashev A Ermakov I and Ilyin I 2012 Ways to improve quality and reliability of software in aerospace industry Large-Scale Systems Control 39 288-299 (in Russian)
[6] Tyugashev A 2017 Use of graph-based and algebraic models in lifecycle of real-time flight control software CEUR Workshop Proceeding 1904
[7] Tyugashev A 2017 Build and evaluation of real-time control algorithms in case of incomplete information about functional processes' parameters Proc. XX IEEE Int. Conf. on Soft Computing and Measurements 179-185

Acknowledgments
Author wants to gratefully acknowledge Professor Anatoly Kalentyev who introduced him into world of science and particular area of space technologies and who found basis of Real Time Control Algorithm’s Logic.