Estudo Comparativo de Métodos de Verificação numa Urgência Hospitalar

Miguel Ângelo Brázia Santos

Extended Abstract

November de 2007
Index

1. Introduction ......................................................................................................................... 3
2. Formal Verification .................................................................................................................. 3
3. Model Checking ...................................................................................................................... 3
4. Theorem Proving ..................................................................................................................... 4
5. Case Study ............................................................................................................................ 4
6. Verification Property ............................................................................................................... 5
   6.1. Model Checking Property ............................................................................................... 5
   6.2. Theorem Proving Property ............................................................................................. 5
7. Verification ............................................................................................................................ 5
   7.1. Model Checking ............................................................................................................. 5
   7.2. Theorem Proving ............................................................................................................ 6
8. Comparison ............................................................................................................................ 6
   8.1. Modelling Language ....................................................................................................... 6
   8.2. Property Definition ......................................................................................................... 7
   8.3. Time Dispended ............................................................................................................. 7
9. Conclusions .......................................................................................................................... 7
1. Introduction

The objective of this study is to compare two different software formal verification methods by applying them to the same case study. Its motivation arises from a study made before which used one of the methods we speak of to prove correct the model of a given system. Software formal verification is very important because of the impact software has in today's world. There are several formal methods for verifying software but there are few comparisons between those methods.

2. Formal Verification

Formal verification is a way of testing the model of a system against some requirements, being a special case those gathered before its design. It’s a very reliable method because it makes a mathematical analysis of the system in study. The existing methods differ on the way they effectively do this analysis. In this paper we address two formal verification methods, model checking and theorem proving, and analyse their differences, advantages and disadvantages from the point of view of a computer engineer.

3. Model Checking

The several methods for formal verifications differ on many aspects, being the most important one the main aspects analysed in the model. Model checking is a well known formal verification method which emphasises de communication between the several elements of the system. It is widely used to prove correctness of hardware. The software systems analysed by this method are generally concurrent and its components communicate by sending and receiving (synchronous or asynchronous) messages.

The mathematic analysis made by model checking consists of creating a finite automaton which represents the original model. The model is written in a modelling language (PROMELA – PRocess MEta LAnguage) which is interpreted by the tool used to perform the verification (SPIN) and translated to an automaton. The requirements against which we want to prove the correctness of the model are written in a specific language (in this case, LTL – Linear Time Logic) and are also interpreted by the tool, generating another automaton.

The verification process consists of computing the first automaton with the negation of the second one. This process results in yet another automaton. If this resulting automaton recognizes only the empty language (\( \{ \} \)), it means that the model is correct. Otherwise, the model is not correct and the states of the
resulting automaton point directly to the error states of the model. Using these states, the user can modify correct the model.

The major advantage of model checking is the automation of the verification process. Once you have introduced the model and the requirements, the verification is straightforward. You push the button and wait for the result. On the other hand, the major disadvantage is that it can only verify systems with finite states. The internal representation used for the model and requirements is problematic because the number and size of the states of the automata rise exponentially, rapidly consuming all the physical memory of the machine in which the verifications is being performed.

4. Theorem Proving

Theorem proving is a formal verification method which emphasises the information created and maintained by the system. With theorem proving it’s possible to create complex data types. The modelling language used in this study is AMN (Abstract Machine Notation) which is totally based in sets theory.

The process verification consists of, based on the model (which includes the requirements we want to verify in the form of a machine invariant), construct a number of theorems which have to be mathematically proved so that the machine can be considered correct.

The main advantage of theorem proving is the fact that it can prove infinite systems because of the way it proves the model. Its main disadvantage is the verification process itself. The tool (in this study we used Click 'n Prove B4Free, a free version of Atelier-B) has an automatic prover that most of the times can't prove all the theorems generated. When this happens, expert help is needed.

5. Case Study

The chosen example for this comparison was taken from a PhD thesis by Carlos Santos. It's about an emergency room. Sick people go to the ER to be treated. An individual arriving the ER may be dependent or independent. We consider that one is independent when he can move by himself and can register without help other than the employee of the ER. On the other hand, one is considered to be dependent if he cannot move properly by himself and needs help. In this case, he should be accompanied by other person who should register him in the ER service. Also, the ER has some wheelchairs or stretches for moving these people inside the ER.

The first step a person has to do in the ER is registration. After this, he is able to be attended. So, the next step is being treated by medical personnel. After being treated, sick people are discharged and leave the ER.
6. Verification Property

As stated before, formal software verification is based on proving a model correct against some properties. After knowing the model which is being used, we can define those properties. In this case, we want to prove that every person who enters de Emergency Room, also leaves it. There is a multitude of ways this could happen. A person can be discharge, die, be admitted to some specific unit in the hospital, etc. We don’t care which way it is, since the patient effectively leaves the ER. Because we need to represent one of the enumerated cases in our models, we chose to use the “normal way”, discharge.

6.1. Model Checking Property

The model chosen for this study is quite simple and it was not difficult to write it in PROMELA. In the same way, writing the property which is going to be proved is also quite simple and doesn’t require the use of LTL. Spin can verify some properties other than those specified in LTL. One of those properties is assertions introduced in the model itself. This was the way we proved this model correct without using LTL.

6.2. Theorem Proving Property

In my point of view, writing a model for theorem proving is somewhat more difficult than for model checking. This model resulted more complex than the other one, in the same way that it was more difficult to define the verification property.

Every patient has a clock associated to him. When he enters the ER the clock is initialized and it is incremented when some act is done on the patient. In this case, the verification property states than no patient can be in the ER more than a pre-specified time interval.

7. Verification

The verification process is different for each of the methods we chose to use. This chapter will present the steps taken when proving the model either in model checking or theorem proving.

7.1. Model Checking

By the time we reach this process, we already wrote the model in PROMELA and specified the property we want to prove. The next step is precisely doing the prove. With the tool Spin, this is straight forward. All
We have to do is push a button and the checker will do the rest. This process can end in three different ways. The first is success: the model checker terminates with a message to the user saying it reached the end of the check finding no errors in the model. The second is error: the model checker finds some error in the model and reports it giving an error message and an example for the user to understand the error found. Finally, the checker can state that the physical memory of the machine in which it is running is not enough to check the model totally. It states that as far as it has gone, it found no error but there could be errors in the part of the model that wasn’t checked.

Because of this problem with memory, our tool has some options the user can chose to decrease the amount of memory used. One of them (that had to be used in this study in order to check the model) avoids the creation of the automaton, representing the model in some mathematical logic. In this way, the representation of the model takes much less memory (orders of magnitude less memory) than the necessary to represent the automaton.

The model used in this study was proved correct using the option referred in the last paragraph successfully.

**7.2. Theorem Proving**

The process of verifying the model in Click ’n Prove consists of inserting the model in the tool and taking a series of steps. In the first step, the tool does a type check, searching for problems with type definitions. The second step consists of generating the theorems which are going to be proved. In the third step, the prover tries to prove the theorems. All these three steps are totally automatic and performed by Click ’n Prove with no external help. The forth step consists of a human trying to prove those theorems the prover didn’t succeed proving.

In this specific case, there were some theorems that I could not prove by myself. So I searched help of someone who knew the method and together we managed to succeed in the proof as we proved the model correct.

**8. Comparison**

We can use several items to compare these two formal verification methods. The items we’ll refer to are modelling language, property definition and time dispended in the process.

**8.1. Modelling Language**

Both modelling languages used by these two methods are a bit difficult at first but, due to some resemblance with known technology (e.g. programming languages), the user easily learns how to use
them. Despite this, AMN is more difficult than PROMELA because it's entirely based on mathematic definitions.

8.2. Property Definition

The definition of properties in PROMELA represents additional work once it is made in a different language which is based in mathematical logic, though presenting the same problem as model definition in theorem proving. On the other hand, with AMN, once the user is familiarized with the language and sets theory, writing the properties is straightforward.

8.3. Time Dispended

The amount of time spent in either of the processes doesn't differ much. The main difference is what we were doing on the bigger part of that time.

In model checking, the most of the time was spent trying to simplify the model in order to the verification process to terminate (once again, the memory problem), while the verification itself is quite quick. In theorem proving, the most of the time was spent manually proving the theorems generated by the theorem prover.

9. Conclusions

After all this work, the method that seems easier is the model checking. In a real case I would chose model checking (although I had to have some powerful machine, regarding its memory). All its simplicity gains the benefits of using theorem proving. But, if the model we want to prove can't be proved by model checking (because we cannot simplify it enough or because the model is effectively infinite) we definitely have to forget the simplicity of model checking and emphasise the power of theorem proving. Although we might need some specialized help, it's better to have to get that help than not managing to prove the model.