Available online at www.sciencedirect.com

SciVerse ScienceDirect

Electronic Notes in Theoretical Computer Science

Electronic Notes in Theoretical Computer Science 280 (2011) 47–56

www.elsevier.com/locate/entcs

Proving Reachability in B using Substitution Refinement

Marc Frappier[a,1,2] Fama Diagne[a,b,3] Amel Mammar[b,4]

[a] *GRIL, Dép. d'informatique, Université de Sherbrooke, Sherbrooke (Québec), Canada*

[b] *Institut Telecom SudParis, CNRS/SAMOVAR, Paris, France*

Abstract

This paper proposes an approach to prove reachability properties of the form AG ($\psi \Rightarrow$ EF ϕ) using substitution refinement in classical B. Such properties denote that there exists an execution path for each state satisfying ψ to a state satisfying ϕ. These properties frequently occur in security policies and information systems. We show how to use Morgan's specification statement to represent a property and refinement laws to prove it. The idea is to construct by stepwise refinement a program whose elementary statements are operation calls. Thus, the execution of such a program provides an execution satisfying AG ($\psi \Rightarrow$ EF ϕ). Proof obligations are represented using assertions (ASSERTIONS clause of B) and can be discharged using Atelier B.

Keywords: B Notation, proof, reachability, CTL, refinement calculus

1 Introduction

Reachability properties frequently occurs in information systems and security policies. For example, in a library system, a typical property is that a member should always be able to borrow a book. If the book is available and the member hasn't reached his loan limit, he can proceed immediately and borrow the book; if he has reached his loan limit, he can return one of his borrowed book and then borrow the book. If the book is already borrowed by another member, then he can make a reservation and wait for his turn to borrow the book. In this description, we see that the specifier must take into account several cases when proving such a property. They are documented in use cases and scenarios during requirements analysis. In a

[1] This research is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC)

[2] Email: Marc.Frappier@USherbrooke.ca

[3] Email: Fama.Diagne@USherbrooke.ca

[4] Email: Amel.Mammar@it-sudparis.eu

doi:10.1016/j.entcs.2011.11.017

clinical information system, similar properties arise, especially when access control rules and patient consent rules are used. One wants to ensure that in case of emergencies, doctors can still reach the desired information, or appropriately process patient transfers between departments, etc.

Such reachability properties can be expressed in CTL [8], because one only needs to show the existence of a path. LTL [18] is inappropriate, because LTL properties must be satisfied by all execution paths. Our reachability properties need not be satisfied by all execution path: for instance, a member is never forced to borrow a book. In fact, such properties triggered the definition of CTL in the early 80's.

This form of reachability is expressed in CTL as $\mathsf{AG}\ (\psi \Rightarrow \mathsf{EF}\ \ \phi)$. This formula denotes that there exists an execution path from each state satisfying ψ to a state satisfying ϕ. In the context of proving CTL properties for classical B abstract machines, an execution path is a sequence of operation calls. One way to prove a reachability property is to provide a program p, whose elementary statements are operation calls, which are combined with some operators, and to show that $\psi \Rightarrow [p]\phi$. Operator "[ ]" is the traditional substitution semantic operator of the B theory, which is the same as Dijkstra's weakest precondition operator, denoted by $wp(p, \phi)$. This statement essentially states that p, when started in ψ, is guaranteed to terminate in a state satisfying ϕ. By proving this statement, one proves the existence of a path from ψ to ϕ. If one uses B substitutions to construct p, then the B theory can be used to prove this statement.

Existing tools like Atelier B cannot directly handle an expression like $\psi \Rightarrow [p]\phi$, but such an expression can easily be translated into an assertion, using the laws of "[ ]". One can then use traditional tools like Atelier B to prove it. However, the proof obligations generated from $\psi \Rightarrow [p]\phi$ can be large and complex, so hard to prove. This is why the idea of refinement calculus was introduced [2,3,10,13,15,17]. In this paper, we show how to prove these statements using the well-documented refinement calculus of Carroll Morgan [14] in a B context.

2 Proving Reachability using Substitution Refinement

Morgan has proposed a number of refinement rules to develop sequential programs in a stepwise fashion. He introduced the notion of *specification statement*, denoted by $w : [pre, post]$, that specifies a computation which, when started in a state satisfying pre, must terminate in a state satisfying $post$, by modifying variables w. To avoid any confusion with "[...]" of the B notation, let us write Morgan's specification statement as $\mathsf{Spec}(pre\ ,\ post)$ and consider it as a new B substitution. We eliminate w from the notation, because it suffices for our purpose to implicitly let w denote all variables of a B machine. Note that this statement can be written in B as:

$$w : [pre, post] = \text{PRE } pre \text{ THEN ANY } w' \text{ WHERE } post' \text{ THEN } w := w' \qquad (1)$$

The wp-semantics of $\mathsf{Spec}(pre\ ,\ post)$ is defined as follows:

$$[\mathsf{Spec}(pre\ ,\ post)]Q \quad \Leftrightarrow \quad pre \wedge (\forall\, w \cdot post \Rightarrow Q)$$