Discussion of SuperMarket Model Ken Robinson April 22, 2009

1 Purpose of this tutorial

This tutorial will discuss the Event-B/RODIN solution previously distributed, which will be referred to as SuperMarketSolnV0. It is recommended that the archive for this model be installed and referenced through RODIN. The new supermarket model is SuperMarketSolnV1 and an archive of that is also available.

This discussion will:

1. Explain some aspects of the model.
2. Correct any errors in the model that are noted.
3. Discuss undischarged proof obligations. Some of that discussion will uncover errors or omissions in the model.
4. Discuss the use of the RODIN provers.

1.1 SuperMarket_ctx: The Context

PRODUCT the set of all products of interest to the supermarket. In the original solution this was given as an enumerated set. This might make the model seem more “real”, but it is not necessary. In version 1 of the solution it will simply be a finite set. This requires a new axiom finite(PRODUCT), which was implicit in the case it was an enumerated set.

SHELF models a shelf of products, that is a bag of products. Originally modelled as a partial function, PRODUCT \rightarrow \mathbb{N}, but as a partial function it doesn’t make a lot of sense to use \mathbb{N} for the range (quantity) so will change to \mathbb{N}_1.

PRICE modelled as \mathbb{N}.

BASKET a bag of products associated with a customer’s trolley, modelled as PRODUCT \rightarrow \mathbb{N}_1.

ProdInTrolleys a function that models a bag of products composed of the bags of products that are the baskets of the set of trolleys.
1.2 ProdIntTrolleys

The Prodinttrolleys function in the earlier model was incomplete and also faulty. The type of Prodinttrolleys is

$$ProdIntTrolleys \in PRODUCT \times (TROLLEY \rightarrow BASKET) \rightarrow \mathbb{N}$$

where

$$BASKET = PRODUCT \rightarrow \mathbb{N}$$

models a set of bags of products. This will be used to model the basket associated with trolley. The intended meaning is that ProdIntTrolleys\((product \mapsto customers)\) where

$$customers \in trolleys \rightarrow (products \rightarrow \mathbb{N})$$

denotes the sum of the count of product across all the trolleys in use by customers. Thus ProdIntTrolleys models a distributed bag.

**Defining the behaviour of ProdIntTrolleys**

The likely first mistake in defining the behaviour of ProdIntTrolleys is to attempt to define a function that will compute the sum for some particular state of the supermarket. This is doomed to failure and is not necessary. What we need to define is how the sum of some product changes in events. The process resembles and inductive proof:

- **Initial state:** in which there are no trolleys and the sum of any product will be 0.

- **an event:** the event may change the state of the trolleys —usually only one trolley— and we are interested in the incremental change in the sum of the products.

We will present a set of axioms that describe the various ways in which the content of the trolleys will be modified.

No customers, hence no products in trolleys

$$axm8: \forall p, ts \cdot p \in PRODUCT \land ts \in TROLLEY \rightarrow BASKET \land ts = \emptyset \Rightarrow ProdIntTrolleys(p \mapsto ts) = 0$$

The trolley for one basket is modified, and the modified basket contains the product:

$$axm9: \forall p, t, b \cdot p \in PRODUCT \land t \in TROLLEY \land b \in BASKET \land p \in \text{dom}(b) \Rightarrow ProdIntTrolleys(p \mapsto \{t \mapsto b\}) = ProdIntTrolleys(p \mapsto \{t\} \cup ts) + b(p)$$
The trolley for one basket is modified, and the modified basket does not contain the product:

axm10: \( \forall p, ts, t, b \cdot p \in PRODUCT \land ts \in TROLLEY \rightarrow BASKET \land t \in TROLLEY \land b \in BASKET \land p \notin \text{dom}(b) \rightarrow \prod_{\text{Trolleys}}(p \mapsto (ts \leftarrow \{t \mapsto b\})) = \prod_{\text{Trolleys}}(p \mapsto (\{t\} \leftarrow ts)) \)

A trolley containing the product is removed:

axm11: \( \forall p, ts, t \cdot p \in PRODUCT \land ts \in TROLLEY \rightarrow BASKET \land t \in \text{dom}(ts) \land p \in \text{dom}(ts(t)) \rightarrow \prod_{\text{Trolleys}}(p \mapsto (\{t\} \leftarrow ts)) = \prod_{\text{Trolleys}}(p \mapsto ts) - ts(t)(p) \)

A trolley that does not contain the product is removed:

axm12: \( \forall p, ts, t \cdot p \in PRODUCT \land ts \in TROLLEY \rightarrow BASKET \land t \in \text{dom}(ts) \land p \notin \text{dom}(ts(t)) \rightarrow \prod_{\text{Trolleys}}(p \mapsto (\{t\} \leftarrow ts)) = \prod_{\text{Trolleys}}(p \mapsto ts) \)

1.3 Proof

In this section we will discuss all undischarged proof obligations and show how they are discharged.

1.3.1 INIT/inv14/INV

Goal: \( \prod_{\text{Trolleys}}(p \mapsto \emptyset) = 0 \)

Find universal quantification for case with \( ts = \emptyset \)

\[ \forall p, ts, t \cdot \prod_{\text{Trolleys}}(p \mapsto ts) = 0 \]

and instantiate \( \prod_{\text{Trolleys}} \) with \( p = p \) and \( ts = \emptyset \).

New hypothesis

\[ \prod_{\text{Trolleys}}(p \mapsto \emptyset) = 0 \]
Click on $\rightarrow$ and select apply modus ponens.

goal: $\emptyset \in TROLLEY \rightarrow BASKET$

run auto prover PO discharged.

### 1.3.2 SetPrice/inv2/INV

goal: $shelf' \in products \cup product \rightarrow \mathbb{N}$

Examine the hypotheses and you will notice that it is not known whether $product \in products$. The proof can go in two directions.

Enter $product \in products$ in the proof control scratchpad and select dc to proceed by cases. The proof will proceed first with $product \in products$ and when the goal is discharged the proof will proceed with $product \notin products$.

Select autoprover and PO will discharge and the hypotheses will display the second case.

Select autoprover again and PO will completely discharge.

### 1.3.3 SetPrice/inv3/INV

goal: $stock'(p) = shelf'(p) + ProdInTrolleys(p \rightarrow customers)$

Again use case analysis on $product$ starting with $product \in products$.

Two runs of the autoprover should discharge the PO.