UFR-IMAG
Université Joseph Fourier

Programming Language and Compiler Design, 2010/2011

Marion Daubignard
Yassine Lakhnech
Laurent Mounier

Series 1

Exercise 8

We build a set B of boolean expressions using the following elements:

- constants $true$ and $false$,
- a set of boolean variables denoted $Bool$
- $\neg$ rule: if $b \in B$ then $(\neg b) \in B$
- $\wedge$ rule: if $b_1, b_2 \in B$ then $(b_1 \wedge b_2) \in B$

Write the formal sentence corresponding to the following english sentence and prove it: two states that coincide on every boolean variable yield equal values for any expression in B.
Optional question : how can we adapt this statement for **Bexp** ?

Semantics of expressions in B:
We can copy what we did for **BExp**. States associate boolean values to boolean variables. Let $\mathcal{B}$ denote the semantics function for expressions in B.

- $\mathcal{B}[true]\sigma = tt$, $\mathcal{B}[false]\sigma = ff$;
- $\forall z \in Bool$, $\mathcal{B}[z]\sigma = \sigma(z)$;
- $\mathcal{B}[\neg b]\sigma = \begin{cases} ff & \text{if } \mathcal{B}[b]\sigma = tt \\ tt & \text{otherwise} \end{cases}$
- $\mathcal{B}[b_1 \wedge b_2]\sigma = \begin{cases} tt & \text{if } \mathcal{B}[b_1]\sigma = tt \text{ and } \mathcal{B}[b_2]\sigma = tt \\ ff & \text{otherwise} \end{cases}$

What we want to prove the following property:
For any expression $b \in B$, if X is the set of boolean variables appearing in b,

$$\forall x \in X, \sigma(x) = \sigma'(x) \Rightarrow \mathcal{B}[b]\sigma = \mathcal{B}[b]\sigma'$$

By induction on b:

- if $b = true$, $\mathcal{B}[b]\sigma = \mathcal{B}[true]\sigma = tt = \mathcal{B}[true]\sigma' = \mathcal{B}[b]\sigma'$,
- if $b = false$, $\mathcal{B}[b]\sigma = ff = \mathcal{B}[b]\sigma'$ (same as for $true$)
- if b is a variable z, $\mathcal{B}[z]\sigma = \sigma(z) \overset{H}{=} \sigma'(z) = \mathcal{B}[z]\sigma'$
- We assume b satisfies the induction hypothesis. It means that if X denotes the set of boolean variables appearing in b, then $\forall x \in X, \sigma(x) = \sigma'(x) \Rightarrow \mathcal{B}[b]\sigma = \mathcal{B}[b]\sigma'$.
  Let us show that the property is preserved when we use the $\neg$ rule to build other expressions. By defintion of function $\mathcal{B}$,
  $\mathcal{B}[\neg b]\sigma = \begin{cases} ff & \text{if } \mathcal{B}[b]\sigma = tt \\ tt & \text{otherwise} \end{cases}$
  and
  $\mathcal{B}[\neg b]\sigma' = \begin{cases} ff & \text{if } \mathcal{B}[b]\sigma' = tt \\ tt & \text{otherwise} \end{cases}$
  Our induction hypothesis yields $\mathcal{B}[b]\sigma = \mathcal{B}[b]\sigma'$, thus if $\mathcal{B}[b]\sigma = tt$, then $\mathcal{B}[b]\sigma'$ too, and $\mathcal{B}[\neg b]\sigma = ff = \mathcal{B}[\neg b]\sigma'$. Moreover, if $\mathcal{B}[b]\sigma = ff$, then $\mathcal{B}[b]\sigma'$ too, and $\mathcal{B}[\neg b]\sigma = tt = \mathcal{B}[\neg b]\sigma'$.

- We assume b_1 and b_2 satisfy the induction hypothesis. It means that if X is the set of boolean variables appearing in b_1 and b_2, then $\forall x \in X, \sigma(x) = \sigma'(x) \Rightarrow \mathcal{B}[b_1]\sigma = \mathcal{B}[b_1]\sigma'$ and $\forall x \in X, \sigma(x) = \sigma'(x) \Rightarrow \mathcal{B}[b_2]\sigma = \mathcal{B}[b_2]\sigma'$.

  Now we want to show that $b_1 \wedge b_2$ verifies the property. We take σ and σ' such that $\forall x \in X, \sigma(x) = \sigma'(x)$. By definition of the semantics function,

  $\mathcal{B}[b_1 \wedge b_2]\sigma = \begin{cases} tt & \text{if } \mathcal{B}[b_1]\sigma = tt \text{ and } \mathcal{B}[b_2]\sigma = tt \\ ff & \text{otherwise} \end{cases}$

  and

  $\mathcal{B}[b_1 \wedge b_2]\sigma' = \begin{cases} tt & \text{if } \mathcal{B}[b_1]\sigma' = tt \text{ and } \mathcal{B}[b_2]\sigma' = tt \\ ff & \text{otherwise} \end{cases}$

  Two cases can arise : first, if $\mathcal{B}[b_1]\sigma = tt$ and $\mathcal{B}[b_2]\sigma = tt$ too. This provides $\mathcal{B}[b_1 \wedge b_2]\sigma = tt$ by definition of the semantics function. Moreover, applying our induction hypothesis, $\mathcal{B}[b_1]\sigma = \mathcal{B}[b_1]\sigma'(= tt)$ and $\mathcal{B}[b_2]\sigma = \mathcal{B}[b_2]\sigma'(= tt)$. Then $\mathcal{B}[b_1 \wedge b_2]\sigma' = tt$, by definition of the semantics function. We conclude to $\mathcal{B}[b_1 \wedge b_2]\sigma = \mathcal{B}[b_1 \wedge b_2]\sigma'$ in this case.

  Secondly, if $\mathcal{B}[b_1]\sigma = ff$ or $\mathcal{B}[b_2]\sigma = ff$, by definition of the semantics function, $\mathcal{B}[b_1 \wedge b_2]\sigma = ff$. Moreover, as $\mathcal{B}[b_1]\sigma = \mathcal{B}[b_1]\sigma'$ and $\mathcal{B}[b_2]\sigma = \mathcal{B}[b_2]\sigma'$ (according to our induction hypothesis), we have that $\mathcal{B}[b_1]\sigma' = ff$ or $\mathcal{B}[b_2]\sigma' = ff$. Then, by definition of the semantics function, $\mathcal{B}[b_1 \wedge b_2]\sigma' = ff$. We conclude to $\mathcal{B}[b_1 \wedge b_2]\sigma = \mathcal{B}[b_1 \wedge b_2]\sigma'$ in this case too.

  As the conclusion holds in both cases, it holds in general: $\mathcal{B}[b_1 \wedge b_2]\sigma = \mathcal{B}[b_1 \wedge b_2]\sigma'$.

We adapt the statement for BExp:
In the definition of boolean expressions in the lecture, expressions $a_1 = a_2$ and $a_1 \leq a_2$ play the role of boolean variables of this exercise: they are atomic expressions whose 'meaning' (i.e. value associated by the semantics function $\mathcal{B}$) can change. Hence, if we want to characterize the states for which meanings of boolean expressions of **BExp** coincide, we have to impose that states give the same meaning to arithmetic expressions. We proved in exercise 2 that if we impose on states to coincide on variables in Var, then states give the same meaning to arithmetic expressions.