University of Sussex
Informatics

Spring 2020

Limits of Computation
Feedback for Exercises 6
Dr Bernhard Reus

WHILE-(Un)Decidability, Rice's Theorem and Reduction
(Lectures 8–10)

1. Below you find a list (a–c) of decision problems about (encodings of) WHILE-programs. Which of these problems are WHILE-decidable and which are undecidable? In other words, for which problems A can we write a WHILE-program that takes $d \in \mathbb{D}$ as input and decides $d \in A$? (returning `true` or `false` accordingly)? Explain your answer.

   Answer:

   (a) A is the set of WHILE-programs p, encoded i.e. in form $\ulcorner p \urcorner$, such that p terminates for *all* inputs.
   We notice immediately that the property of programs in A (i.e. $[\![p]\!]^{\texttt{WHILE}}(\texttt{d}) \downarrow$ for all d) is *non-trivial*, that is, we know that there is at least one program with this property, for instance program `read X { } write X`. Moreover, we know that not all programs in $\mathbb{D}$ terminate for all input. For instance, program `read X { while X { } } write X` does not terminate for any other input than nil.
   Furthermore, we notice that the property we are considering here is *extensional*, which means that the property of p only depends on the input-output behaviour of the program p rather than its appearance. This can be seen by writing down this problem (in this case property of WHILE-programs) more carefully: Program p terminates for *all* inputs is the following property (or set) A:

   $$p \in A \text{ if, and only if, } [\![p]\!]^{\texttt{WHILE}}(d) \downarrow \text{ for all } d \in \mathbb{D}$$

   So if $[\![p]\!]^{\texttt{WHILE}} = [\![q]\!]^{\texttt{WHILE}}$ we get that $[\![p]\!]^{\texttt{WHILE}}(d) \downarrow$ for all $d \in \mathbb{D}$ iff $[\![q]\!]^{\texttt{WHILE}}(d) \downarrow$ for all $d \in \mathbb{D}$ and thus $p \in A$ iff $q \in A$.
   As property A is about programs, non-trivial and extensional if satisfies all conditions of Rice's theorem are therefore we can reason by Rice's theorem that A is *undecidable*.

(b) A is the set of `WHILE`-programs p where p is one of two programs, in other words:

$$A = \left\{ \ulcorner p \urcorner \mid \begin{array}{l} p = \texttt{read X \{ while 1 \{\} \} write X} \quad \vee \\ p = \texttt{read Z \{ Z := cons X nil \} write Z} \end{array} \right\}$$

In this case A is the two-element set containing the (encodings of) the two programs above. We can write a program which inspects each node of an input program-as-data and returns `false` the moment it fails to find a match for the current node in one of the two elements of A. If each node is matched against each node of one or the other of the programs in A then our deciding program will return `true`. We notice that both programs in A are finite in length therefore our program always terminates. Since we are satisfying both conditions for the decidability of A we can state that this problem is *decidable*. Note that the encoded program use numbers top identify variables not names. *Note:* We assume that the numbering is uniquely determined, for instance by starting with 0 and numbering variable sin the order of appearance.

(c) A is the set of `WHILE`-programs p that decide whether its argument d is nil.
This property A of programs is clearly non-trivial, as we can think immediately of a program that can tell whether its input is nil, e.g.

```
read X{
   if X {
      Y:= false }
   else {
      Y:= true
    }
  }
write Y
```

and not all programs satisfy this property (pick your favourite different program :-)).
Secondly, this property A is *extensional* since it depends solely on the input-output behaviour of the program. This can be seen by writing up the definition of A formally again and checking (as done for (a)).

$$p \in A \text{ if, and only if,}$$

$$[\![p]\!]^{\texttt{WHILE}}(d) = \mathit{true} \text{ iff } d = \text{nil for all } d \in \mathbb{D}$$

So if $[\![p]\!]^{\texttt{WHILE}} = [\![q]\!]^{\texttt{WHILE}}$ we get that $[\![p]\!]^{\texttt{WHILE}}(d) = \mathit{true}$ iff $d = \text{nil}$ for all $d \in \mathbb{D}$ iff $[\![q]\!]^{\texttt{WHILE}}(d) = \mathit{true}$ iff $d = \text{nil}$ for all $d \in \mathbb{D}$ iff $q \in A$.

Since all conditions of Rice's theorem are met, it can be applied and Rice's theorem tells us again that this problem is *undecidable*.

2. Give each one example of

   (a) an undecidable problem that is semi-decidable:

   Answer: Halting Problem

   (b) a decidable problem that is semi-decidable:

   Answer: any decidable problem is semi-decidable, e.g. is a number even?

   (c) an undecidable problem that is not semi-decidable:

   Answer: complement of the Halting problem.

3. Prove that for two sets (or problems) A and B, if $A \leq_{\text{rec}} B$ and B is decidable, then A is also decidable (as stated in Lecture 9), using "WHILE-decidable" as notion of "decidable", i.e. WHILE-programs as effective procedures.

   Proceed as follows:

   (a) explain what $A \leq_{\text{rec}} B$ means.

   Answer:
   $A \leq_{\text{rec}} B$ means that there is a total and (WHILE-)computable function $f : \mathbb{D} \to \mathbb{D}$ such that $x \in A$ if, and only if, $f(x) \in B$.

   Since f is (WHILE-)computable we therefore know there must be a (WHILE-)program r that always terminates such that $[\![r]\!]^{\texttt{WHILE}}(d) = f(d)$ for all $d \in \mathbb{D}$.
   Note that we can replace WHILE here with any other reasonable notion of effective procedures.

   (b) using the obtained assumption and the one obtained from the WHILE-decidability of B, write the program p that proves that A is WHILE-decidable.

   Answer:

   Our further assumption now is that B is decidable, which means that there is a WHILE-program q that aways terminates and for which we know that $[\![q]\!]^{\texttt{WHILE}}(d) = \mathit{true}$ if, and only if, $d \in B$ for all $\texttt{d} \in \mathbb{D}$.
   We need to prove that A is WHILE-decidable, that means

we need to find a WHILE-program p that always terminates and for which we know that $[\![p]\!]^{\texttt{WHILE}}(d) = \mathit{true}$ if, and only if, $d \in A$ for all $d \in \mathbb{D}$.

How do we know how to program this? Well, the reduction gives us the recipe: $x \in A$ if, and only if, $f(x) \in B$. So to decide membership in A we "simply" need to program a procedure that checks whether $f(x)$ is in B. The latter can be done with the help of the decision procedure for B and the implementation of f:

The required decision procedure p can thus be defined as follows:

```
p read d {
 y := <r> d;
 out := <q> y
}
write out
```

By definition of p and using the semantics of WHILE-programs, we get (omitting the detailed steps of the semantics as done once on Exercise Sheet 2) that

$$[\![p]\!]^{\texttt{WHILE}}(d) = [\![q]\!]^{\texttt{WHILE}}([\![r]\!]^{\texttt{WHILE}}(d)) = [\![p]\!]^{\texttt{WHILE}}(f(d))$$

which by our assumption about q is *true* if, and only if, $f(d) \in B$. But because of our assumption that $x \in A$ if, and only if, $f(x) \in B$, we get that

$$[\![p]\!]^{\texttt{WHILE}}(d) = [\![q]\!]^{\texttt{WHILE}}(f(d)) = \mathit{true} \,\text{iff}\, f(d) \in B \,\text{iff}\, d \in A$$

and so we have shown that $[\![p]\!]^{\texttt{WHILE}}(d) = \mathit{true} \,\text{iff}\, d \in A$.

It remains to show that p will always terminate, but this is trivial, since we know that r and q always terminate by assumption.

We thus have proved that A is WHILE-decidable.