# Model Checking (IN2050): Exercise 1
# Proving Correctness

Andrey Rybalchenko

April 22, 2010

**Abstract**

We present a basic algorithm that computes the set of nodes reachable in a (finite) graph. For this algorithm we formulate the corresponding correctness properties and present their proofs.

## 1 Reachability algorithm

Figure 1 presents an algorithm BRA that computes graph nodes that are reachable from the start node by traversing the graph edges.

## 2 Properties and proofs

### 2.1 Termination

**Theorem 1.** *BRA terminates on finite graphs.*

*Proof.* **Ordering**

We define an ordering on pairs of integer numbers as follows.

$$(a, b) > (a', b') : \quad a > a' \lor a = a' \land b > b'$$

Then $(a, b) \geq (a', b')$ if either $(a, b) = (a', b')$ or $(a, b) > (a', b')$.

**Ranking function**

Given a finite set $X$, let $|X|$ be its size. Next, we define a ranking function $r$ from the values of `C` and `done` to pairs of integer numbers as follows, which is possible since both `N` and `C` are finite.

$$r(\texttt{C}, \texttt{done}) : \quad (|\texttt{N}| - |\texttt{C}|, \textbf{if } \texttt{done} \textbf{ then } 0 \textbf{ else } 1)$$

```
algorithm BRA
input
  N : set of nodes
  n0 : start node, where n0 \in N
  E : set of edges, where E \subseteq N \times N
var
  C : nodes reached so far
  done : Boolean flag
  D : auxiliary set of nodes
begin
  C := {n0}
  done := false
  while \neg done do
    D := { d \in N | \exists c \in C: (c, d) \in E }
    if \neg (D \subseteq C) then
      C := C \cup D
    else
      done := true
  od
  return C
end.
```

Figure 1: A basic algorithm BRA for computing reachable nodes in a (finite) graph.

For example, for $\mathtt{N} = \{1, 2, 3\}$ we have

$$\begin{array}{ll} r(\{1\}, \mathit{false}) & = (2, 1)\ , \\ r(\{1, 2\}, \mathit{false}) & = (1, 1)\ , \\ r(\{1, 2, 3\}, \mathit{true}) & = (0, 0)\ . \end{array}$$

### Ranking decrease

Now we show that the value of the ranking function decreases during each loop iteration.

First, we consider the path through the loop that traverses the if branch of the conditional statement. The corresponding proof obligation is

$$r(\mathtt{C}, \mathit{false}) > r(\mathtt{C} \cup \mathtt{D}, \mathit{false})\ ,$$

under the assumption that $\neg(\mathtt{D} \subseteq \mathtt{C})$. This assumption implies that there exists a node $\mathtt{d} \in \mathtt{D}$ such that $\mathtt{d} \notin \mathtt{C}$. Hence, $|\mathtt{C}| < |\mathtt{C} \cup \mathtt{D}|$. which proves the obligation.

Second, we consider the path that traverses the else branch. Since the set C does not change, we immediately obtain

$$r(\mathtt{C}, \mathit{false}) > r(\mathtt{C}, \mathit{true})\ .$$

### Ranking bound

We prove that an iteration of the loop can only take place if the algorithm state satisfies the following condition.

$$r(\mathtt{C}, \mathtt{done}) \geq (0, 0)$$

This statement follows from the fact that the algorithm maintains the relation $\mathtt{C} \subseteq \mathtt{N}$ and the loop condition.

### Putting everything together

First, we observe that there is no infinite chain of pairs of integers $(a_0, b_0) > (a_1, b_1) > \dots$ such that for each $i \geq 0$ we have $(a_i, b_i) \geq (0, 0)$. Together with the ranking decrease and ranking bound statements this observation implies termination of the algorithm. □

## 2.2 Reachability

**Theorem 2.** *Each node* `c` *in the set* `C` *computed by BRA is reachable from* `n0` *by following edges from* `E`. *Formally,*

$$\forall \mathtt{c} \in \mathtt{C} : (\mathtt{n0}, \mathtt{c}) \in \mathtt{E}^* \ .$$

*Proof.* We prove the theorem by induction on the number of the loop iterations $k$. Our induction hypothesis $\mathit{Hyp}(k)$ is:

> Each node `c` that was added to `C` at the iteration $k'$ such that $k' \leq k$ is reachable, i.e., $(\mathtt{n0}, \mathtt{c}) \in \mathtt{E}^*$.

### Base case

For $k = 0$, we have $\mathtt{C} = \{\mathtt{n0}\}$. Since $(\mathtt{n0}, \mathtt{n0}) \in \mathtt{E}^*$, $\mathit{Hyp}(0)$ holds.

### Step

We assume that for $k$ the induction hypothesis $\mathit{Hyp}(k)$ holds, i.e., each node `c` added to `C` at the iteration $k'$ such that $k' \leq k$ is reachable, i.e., $(\mathtt{n0}, \mathtt{c}) \in \mathtt{E}^*$. We prove $\mathit{Hyp}(k+1)$, which amounts to proving that `D` computed during the $k+1$th interaction by following the if branch is reachable. The case when the $k+1$th iteration goes through the else branch does not modify `C` and hence $\mathit{Hyp}(k+1)$ holds.

By the induction hypothesis, for each $\mathtt{d} \in \mathtt{D}$ there exists $\mathtt{c} \in \mathtt{C}$ at step $k$ such that $(\mathtt{c}, \mathtt{d}) \in \mathtt{E}$ and `c` is reachable, i.e., $(\mathtt{n0}, \mathtt{c}) \in \mathtt{E}^*$. The induction step follows immediately. □

# A Notation

| English | math | ASCII | example |
|---|---|---|---|
| element of | $\in$ | `\in` | $1 \in \{1, 2, 3\}$ |
| subset of | $\subseteq$ | `\subseteq` | $\{1, 2\} \subseteq \{1, 2, 3\}$ |
| union of | $\cup$ | `\cup` | $\{1, 2\} \cup \{2, 3\} = \{1, 2, 3\}$ |
| intersection of | $\cap$ | `\cap` | $\{1, 2\} \cap \{2, 3\} = \{2\}$ |
| subtraction of | $\setminus$ | `\setminus` | $\{1, 2\} \setminus \{2, 3\} = \{1\}$ |
| Cartesian product | $\times$ | `\times` | $\{1, 2\} \times \{2, 3\} = \{(1, 2), (1, 3), (2, 2), (2, 3)\}$ |
| exists | $\exists$ | `\exists` | |
| forall | $\forall$ | `\forall` | |
| negation | $\neg$ | `\lneg` | |
| conjunction | $\land$ | `\land` | |
| disjunction | $\lor$ | `\lor` | |

Table 1: Mathematical symbols in ASCII.