Rank Annotated Trees

Todor Peev, Vincent Trélat

École Nationale Supérieure des Mines de Nancy
Département Informatique

May 23, 2022

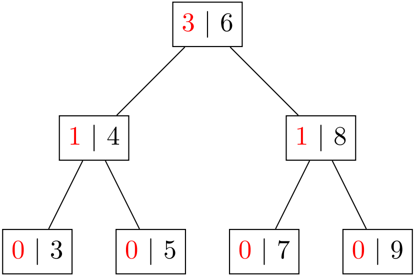
1 Introduction

2 Proofs

Useful functions

First lemmas

Definition



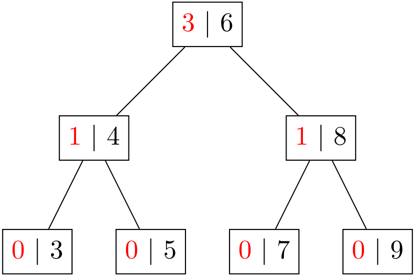
(Type definition)

```
datatype 'a rtree = Leaf | Node "'a rtree" nat 'a "'a rtree"
```

Example 1

$\langle\langle\langle\langle\rangle, 0, 3, \langle\rangle\rangle, 1, 4, \langle\rangle\rangle, 3, 6 :: \mathtt{nat}, \langle\langle\langle\rangle, 0, 7, \langle\rangle\rangle, 1, 8, \langle\langle\rangle, 0, 9, \langle\rangle\rangle\rangle\rangle$

Definition



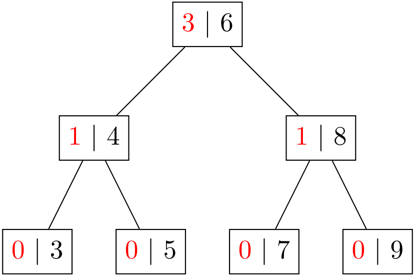
(Type definition)

```
datatype 'a rtree = Leaf | Node "'a rtree" nat 'a "'a rtree"
```

Example 1

$\langle\langle\langle\langle\rangle, 0, 3, \langle\rangle\rangle, 1, 4, \langle\rangle\rangle, 3, 6 :: \mathtt{nat}, \langle\langle\langle\rangle, 0, 7, \langle\rangle\rangle, 1, 8, \langle\langle\rangle, 0, 9, \langle\rangle\rangle\rangle\rangle$

Definition



(Type definition)

```
datatype 'a rtree = Leaf | Node "'a rtree" nat 'a "'a rtree"
```

Example 1

$\langle\langle\langle\langle\rangle, 0, 3, \langle\rangle\rangle, 1, 4, \langle\rangle\rangle, 3, 6 :: \mathtt{nat}, \langle\langle\langle\rangle, 0, 7, \langle\rangle\rangle, 1, 8, \langle\langle\rangle, 0, 9, \langle\rangle\rangle\rangle\rangle$

Useful functions

```
fun num_nodes ::  "'a rtree ⇒ nat" where
   "num_nodes ⟨⟩ = 0" |
   "num_nodes ⟨ l, _, _, r⟩ = 1 + num_nodes l + num_nodes r"
```

```
fun set_rtree ::  "'a rtree ⇒ 'a set" where
   "set_rtree ⟨⟩ = {}" |
   "set_rtree ⟨ l, n, x, r⟩ = set_rtree l ∪ set_rtree r ∪ {x}"
```

```
fun rbst ::  "('a::linorder) rtree ⇒ bool" where
   "rbst ⟨⟩ = True" |
   "rbst ⟨ l, n, x, x⟩ = ((∀a ∈ set_rtree l.  a < x) ∧
      (∀a ∈ set_rtree r.  x < a) ∧
      rbst l ∧
      rbst r ∧
      n = num_nodes l)"
```

Useful functions

```
fun num_nodes ::  "'a rtree ⇒ nat" where
   "num_nodes ⟨⟩ = 0" |
   "num_nodes ⟨ l, _, _, r⟩ = 1 + num_nodes l + num_nodes r"
```

```
fun set_rtree ::  "'a rtree ⇒ 'a set" where
   "set_rtree ⟨⟩ = {}" |
   "set_rtree ⟨ l, n, x, r⟩ = set_rtree l ∪ set_rtree r ∪ {x}"
```

```
fun rbst ::  "('a::linorder) rtree ⇒ bool" where
   "rbst ⟨⟩ = True" |
   "rbst ⟨ l, n, x, x⟩ = ((∀a ∈ set_rtree l.  a < x) ∧
      (∀a ∈ set_rtree r.  x < a) ∧
      rbst l ∧
      rbst r ∧
      n = num_nodes l)"
```

Useful functions

```
fun num_nodes ::  "'a rtree ⇒ nat" where
   "num_nodes ⟨⟩ = 0" |
   "num_nodes ⟨ l, _, _, r⟩ = 1 + num_nodes l + num_nodes r"
```

```
fun set_rtree ::  "'a rtree ⇒ 'a set" where
   "set_rtree ⟨⟩ = {}" |
   "set_rtree ⟨ l, n, x, r⟩ = set_rtree l ∪ set_rtree r ∪ {x}"
```

```
fun rbst ::  "('a::linorder) rtree ⇒ bool" where
   "rbst ⟨⟩ = True" |
   "rbst ⟨ l, n, x, x⟩ = ((∀a ∈ set_rtree l.  a < x) ∧
      (∀a ∈ set_rtree r.  x < a) ∧
      rbst l ∧
      rbst r ∧
      n = num_nodes l)"
```