

Представление структур данных индуктивными семействами и доказательства их свойств

Андрей Рыбак

НИУ ИТМО

Июнь 2014

Научный руководитель: Малаховски Я. М.

- AVL-деревья
- 2-3-деревья

- двоичное дерево
- заполняется слева
- значение в узле \leq значений в корнях поддеревьев

Пустой тип

`data \perp : Set where` - нет конструкторов

Логическое отрицание

$\neg : \forall \{a\} \rightarrow \text{Set } a \rightarrow \text{Set } a$
 $\neg P = P \rightarrow \perp$

Контрапозиция

`contraposition` : $\forall \{A B : \text{Set}\}$
 $\rightarrow (A \rightarrow B) \rightarrow (\neg B \rightarrow \neg A)$
`contraposition` $f \neg b a = \neg b (f a)$

$\text{Rel}_2 : \text{Set} \rightarrow \text{Set}_1$

$\text{Rel}_2 A = A \rightarrow A \rightarrow \text{Set}$

```

data Tri {A : Set} ( _ <_ _ ==_ _ >_ : Rel2 A) (a b : A)
  : Set where
tri< : (a < b) → ¬ (a == b) → ¬ (a > b)
      → Tri _ <_ _ ==_ _ >_ a b - меньше
tri= : ¬ (a < b) → (a == b) → ¬ (a > b)
      → Tri _ <_ _ ==_ _ >_ a b - равно
tri> : ¬ (a < b) → ¬ (a == b) → (a > b)
      → Tri _ <_ _ ==_ _ >_ a b - больше
  
```

```

Cmp : {A : Set} → Rel2 A → Rel2 A → Set
  
```

```

Cmp {A} _ <_ _ ==_ _ = ∀ (x y : A) →
  Tri ( _ <_ ) ( _ ==_ ) (flip1 _ <_ ) x y
  
```

Транзитивность

$$\begin{aligned} \text{Trans} &: \{A : \text{Set}\} \rightarrow \text{Rel}_2 A \rightarrow \text{Set} \\ \text{Trans } \{A\} \text{ _rel_} &= \{a \ b \ c : A\} \\ &\rightarrow (a \text{ rel } b) \rightarrow (b \text{ rel } c) \rightarrow (a \text{ rel } c) \end{aligned}$$

Симметричность

$$\begin{aligned} \text{Symmetric} &: \forall \{A : \text{Set}\} \rightarrow \text{Rel}_2 A \rightarrow \text{Set} \\ \text{Symmetric } \text{ _rel_} &= \forall \{a \ b\} \rightarrow a \text{ rel } b \rightarrow b \text{ rel } a \end{aligned}$$

$$\begin{aligned} & _Respects_ : \forall \{ \ell \} \{ A : Set \} \\ & \rightarrow (A \rightarrow Set \ell) \rightarrow Rel_2 A \rightarrow Set _ \\ P \ Respects _rel_ &= \forall \{ x \ y \} \rightarrow x \ rel \ y \rightarrow P \ x \rightarrow P \ y \end{aligned}$$

$$\begin{aligned} & _Respects_2_ : \forall \{ A : Set \} \\ & \rightarrow Rel_2 A \rightarrow Rel_2 A \rightarrow Set \\ P \ Respects_2 _rel_ &= \\ & (\forall \{ x \} \rightarrow P \ x \ Respects _rel_) \times \\ & (\forall \{ y \} \rightarrow flip \ P \ y \ Respects _rel_) \end{aligned}$$


```
data _<=_ {A : Set}
  {_<_ : Rel2 A}
  {_==_ : Rel2 A} : Rel2 A where
le :  $\forall \{x\ y\} \rightarrow x < y \rightarrow x \leq y$ 
eq  :  $\forall \{x\ y\} \rightarrow x == y \rightarrow x \leq y$ 
```

$$\begin{aligned}
 \text{resp} \leq & : \{A : \text{Set}\} \{ _ < _ : \text{Rel}_2 A \} \\
 & \{ _ == _ : \text{Rel}_2 A \} \\
 & \rightarrow (\text{resp} : _ < _ \text{Respects}_2 _ == _) \\
 & \rightarrow (\text{trans} == : \text{Trans} _ == _) \\
 & \rightarrow (\text{sym} == : \text{Symmetric} _ == _) \\
 & \rightarrow (_ \leq _ \{A\} \{ _ < _ \} \{ _ == _ \}) \text{Respects}_2 _ == _
 \end{aligned}$$

$$\begin{aligned}
 \text{trans} \leq & : \{A : \text{Set}\} \\
 & \{ _ < _ : \text{Rel}_2 A \} \{ _ == _ : \text{Rel}_2 A \} \\
 & \rightarrow _ < _ \text{Respects}_2 _ == _ \rightarrow \text{Symmetric} _ == _ \\
 & \rightarrow \text{Trans} _ == _ \rightarrow \text{Trans} _ < _ \\
 & \rightarrow \text{Trans} (_ \leq _ \{A\} \{ _ < _ \} \{ _ == _ \})
 \end{aligned}$$

Требования к исходному типу

```
module Heap (A : Set) (_ <_ _ ==_ : Rel2 A)
  (cmp : Cmp _ <_ _ ==_)
  (sym== : Symmetric _ ==_)
  (trans== : Trans _ ==_)
  (trans< : Trans _ <_)
  (resp : _ <_ Respects2 _ ==_)
where
```

```
data expanded (A : Set) : Set where  
  # : A → expanded A - элемент исходного типа  
  top : expanded A - элемент расширения
```

Расширенные отношения

```
data _<E_ : Rel₂ (expanded A) where  
  base : ∀ {x y : A} → x < y → (# x) <E (# y)  
  ext : ∀ {x : A} → (# x) <E top
```

```
data _=E_ : Rel₂ (expanded A) where  
  base : ∀ {x y} → x == y → (# x) =E (# y)  
  ext : top =E top
```

lemma-<E : $\forall \{x\} \{y\} \rightarrow (\# x) <E (\# y) \rightarrow x < y$
 trans<E : Trans __<E__

lemma==E : $\forall \{x\} \{y\} \rightarrow (\# x) ==E (\# y) \rightarrow x == y$
 sym==E : Symmetric __==E__
 trans==E : Trans __==E__

respE : __<E__ Respects2 __==E__

cmpE : Cmp {expanded A} __<E__ __==E__

$_ \leq _ : \text{Rel}_2 \text{ (expanded } A)$
 $_ \leq _ = _ < = _ \{ \text{expanded } A \} \{ _ < E _ \} \{ _ = E _ \}$

$\text{trans} \leq : \text{Trans } _ \leq _$
 $\text{trans} \leq = \text{trans} < = \text{resp} E \text{ sym} = E \text{ trans} = E \text{ trans} < E$

$\text{resp} \leq : _ \leq _ \text{Respects}_2 _ = E _$
 $\text{resp} \leq = \text{resp} < = \text{resp} E \text{ trans} = E \text{ sym} = E$

$$\text{minE} : (x\ y : \text{expanded } A) \rightarrow \text{expanded } A$$

$$\text{minE} = \text{min cmpE}$$

$$\text{min3E} : (\text{expanded } A) \rightarrow (\text{expanded } A)$$

$$\rightarrow (\text{expanded } A) \rightarrow (\text{expanded } A)$$

$$\text{min3E } x\ y\ z = \text{min3 cmpE } x\ y\ z$$

$$\text{lemma-}\leq\text{minE} : \forall \{a\ b\ c\} \rightarrow$$

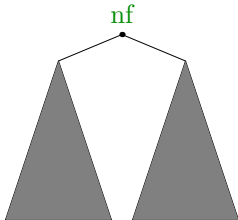
$$a \leq b \rightarrow a \leq c \rightarrow a \leq (\text{minE } b\ c)$$

$$\text{lemma-}\leq\text{min3E} : \forall \{x\ a\ b\ c\}$$

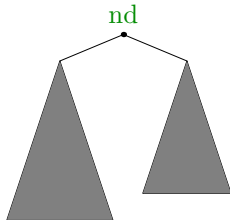
$$\rightarrow x \leq a \rightarrow x \leq b \rightarrow x \leq c \rightarrow x \leq (\text{min3E } a\ b\ c)$$

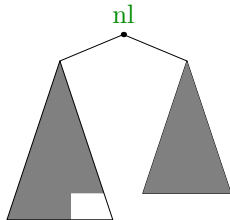
```
data HeapState : Set where  
full almost : HeapState
```

```
data Heap : (expanded A) - минимум  
→ (h :  $\mathbb{N}$ ) - высота  
→ HeapState - заполненность  
→ Set where  
eh : Heap top zero full - Пустая куча
```

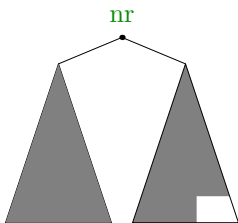

$$\begin{aligned}
 \text{nf} : & \forall \{n\} \{x\ y\} \rightarrow (p : A) \\
 \rightarrow & (i : (\# p) \leq x) \rightarrow (j : (\# p) \leq y) \\
 \rightarrow & (a : \text{Heap } x\ n\ \text{full}) \\
 \rightarrow & (b : \text{Heap } y\ n\ \text{full}) - \text{a b одной высоты} \\
 \rightarrow & \text{Heap } (\# p)\ (\text{succ } n)\ \text{full}
 \end{aligned}$$


$\text{nd} : \forall \{n\} \{x\ y\} \rightarrow (p : A)$
 $\rightarrow (i : (\# p) \leq x) \rightarrow (j : (\# p) \leq y)$
 $\rightarrow (a : \text{Heap } x (\text{succ } n) \text{ full})$
 $\rightarrow (b : \text{Heap } y n \text{ full})$ - а b разной высоты
 $\rightarrow \text{Heap } (\# p) (\text{succ } (\text{succ } n)) \text{ almost}$



$$\begin{aligned}
 &nl : \forall \{n\} \{x \ y\} \rightarrow (p : A) \\
 &\rightarrow (i : (\# p) \leq x) \rightarrow (j : (\# p) \leq y) \\
 &\rightarrow (a : \text{Heap } x \ (\text{succ } n) \ \text{almost}) \\
 &\rightarrow (b : \text{Heap } y \ n \ \text{full}) - \text{b} - \text{полная} \\
 &\rightarrow \text{Heap } (\# p) \ (\text{succ } (\text{succ } n)) \ \text{almost}
 \end{aligned}$$


$nr : \forall \{n\} \{x\ y\} \rightarrow (p : A)$
 $\rightarrow (i : (\# p) \leq x) \rightarrow (j : (\# p) \leq y)$
 $\rightarrow (a : \text{Heap } x (\text{succ } n) \text{ full}) - a - \text{полная}$
 $\rightarrow (b : \text{Heap } y (\text{succ } n) \text{ almost})$
 $\rightarrow \text{Heap } (\# p) (\text{succ } (\text{succ } n)) \text{ almost}$



Вставка в полную кучу

$$\begin{aligned} \text{finsert} &: \forall \{h\ m\} \rightarrow (z : A) \\ &\rightarrow \text{Heap } m\ h\ \text{full} \\ &\rightarrow \Sigma \text{HeapState } (\text{Heap } (\text{minE } m\ (\# z))\ (\text{succ } h)) \end{aligned}$$

Вставка в неполную кучу

$$\begin{aligned} \text{ainsert} &: \forall \{h\ m\} \rightarrow (z : A) \\ &\rightarrow \text{Heap } m\ h\ \text{almost} \\ &\rightarrow \Sigma \text{HeapState } (\text{Heap } (\text{minE } m\ (\# z))\ h) \end{aligned}$$

Слияние двух полных куч одной высоты

$$\begin{aligned}
 & \text{fmerge} : \forall \{x \ y \ h\} \\
 & \rightarrow \text{Heap } x \ h \text{ full} \rightarrow \text{Heap } y \ h \text{ full} \\
 & \rightarrow \text{OR } (\text{Heap } x \ \text{zero full} \times (x \equiv y) \times (h \equiv \text{zero})) \\
 & \quad (\text{Heap } (\text{minE } x \ y) (\text{succ } h) \text{ almost})
 \end{aligned}$$

Составление полной кучи высотой $h + 1$ из двух куч высотой h и одного элемента

$$\begin{aligned} \text{makeH} &: \forall \{x \ y \ h\} \rightarrow (p : A) \\ &\rightarrow \text{Heap } x \ h \text{ full} \rightarrow \text{Heap } y \ h \text{ full} \\ &\rightarrow \text{Heap } (\text{min3E } x \ y \ (\# \ p)) \ (\text{succ } h) \text{ full} \end{aligned}$$

Слияние поддеревьев **nd**

$\text{ndmerge} : \forall \{x \ y \ h\}$
 $\rightarrow \text{Heap } x \ (\text{succ} \ (\text{succ } h)) \ \text{full}$
 $\rightarrow \text{Heap } y \ (\text{succ } h) \ \text{full}$
 $\rightarrow \text{Heap } (\text{minE } x \ y) \ (\text{succ} \ (\text{succ} \ (\text{succ } h))) \ \text{almost}$

Слияние неполной кучи высотой $h + 2$ и полной кучи высотой $h + 1$ или $h + 2$

```
afmerge :  $\forall \{h \times y\}$ 
→ Heap  $x$  (succ (succ  $h$ )) almost
→ OR (Heap  $y$  (succ  $h$ ) full)
    (Heap  $y$  (succ (succ  $h$ )) full)
→ OR (Heap (minE  $x \ y$ ) (succ (succ  $h$ )) full)
    (Heap (minE  $x \ y$ ) (succ (succ (succ  $h$ ))) almost)
```

Извлечение минимума из полной кучи

$$\begin{aligned} \text{fpop} : \forall \{m\ h\} \rightarrow & \text{Heap } m \ (\text{succ } h) \ \text{full} \\ \rightarrow & \text{OR} \\ & (\Sigma \ (\text{expanded } A) \\ & \quad (\lambda x \rightarrow (\text{Heap } x \ (\text{succ } h) \ \text{almost}) \times (m \leq x))) \\ & (\text{Heap } \text{top } h \ \text{full}) \end{aligned}$$

Извлечение минимума из неполной кучи

$$\begin{aligned} \text{apop} : \forall \{m\ h\} \rightarrow & \text{Heap } m \ (\text{succ } h) \ \text{almost} \\ \rightarrow & \text{OR} \ (\Sigma \ (\text{expanded } A) \\ & \quad (\lambda x \rightarrow (\text{Heap } x \ (\text{succ } h) \ \text{almost}) \times (m \leq x))) \\ & \quad (\Sigma \ (\text{expanded } A) \\ & \quad \quad (\lambda x \rightarrow (\text{Heap } x \ h \ \text{full}) \times (m \leq x)))) \end{aligned}$$

Спасибо за внимание!