# Представление структур данных индуктивными семействами и доказательства их свойств

Андрей Рыбак

НИУ ИТМО

июнь 2014

Научный руководитель: Малаховски Я. М.

*Индуктивное семейство* — индуктивный (рекурсивный) тип данных, который может зависеть от других типов и значений.

*Индуктивное семейство* — индуктивный (рекурсивный) тип данных, который может зависеть от других типов и значений.

- Data.AVL
- 2-3-деревья

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

# Отношения

$$\mathsf{Rel}_2 : \mathsf{Set} \to \mathsf{Set}_1$$
$$\mathsf{Rel}_2\ A = A \to A \to \mathsf{Set}$$

# Трихотомия

```
data Tri {A : Set} (_<_ _==_ _>_ : Rel_2 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
```

# Трихотомия

```
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

$$\mathrm{Trans} : \{A : \mathrm{Set}\} \to \mathrm{Rel}_2\ A \to \mathrm{Set}$$
$$\mathrm{Trans}\ \{A\}\ \_rel\_ = \{a\ b\ c : A\}$$
$$\to (a\ rel\ b) \to (b\ rel\ c) \to (a\ rel\ c)$$

Симметричность

$$\mathrm{Symmetric} : \forall\ \{A : \mathrm{Set}\} \to \mathrm{Rel}_2\ A \to \mathrm{Set}$$
$$\mathrm{Symmetric}\ \_rel\_ = \forall\ \{a\ b\} \to a\ rel\ b \to b\ rel\ a$$

```
_Respects_ : ∀ {ℓ} {A : Set}
  → (A → Set ℓ) → Rel₂ A → Set _
P Respects _rel_ = ∀ {x y} → x rel y → P x → P y

_Respects₂_ : ∀ {A : Set}
  → Rel₂ A → Rel₂ A → Set
P Respects₂ _rel_ =
  (∀ {x} → P x  Respects _rel_) ×
  (∀ {y} → flip P y Respects _rel_)
```

# Обобщенное <=

```
data _<=_ {A : Set}
  {_<_ : Rel₂ A}
  {_==_ : Rel₂ A} : Rel₂ A where
  le : ∀ {x y} → x < y → x <= y
  eq : ∀ {x y} → x == y → x <= y
```

```
resp<= : {A : Set} {_<_ : Rel₂ A}
  {_==_ : Rel₂ A}
  → (resp : _<_ Respects₂ _==_)
  → (trans== : Trans _==_)
  → (sym== : Symmetric _==_)
  → (_<=_ {A}{_<_}{_==_}) Respects₂ _==_
```

```
trans<= : {A : Set}
  {_<_ : Rel₂ A} {_==_ : Rel₂ A}
  → _<_ Respects₂ _==_ → Symmetric _==_
  → Trans _==_ → Trans _<_
  → Trans (_<=_ {A}{_<_}{_==_})
```

Требования к исходному типу

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

# Расширение

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

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

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
```

# Свойства расширенных отношений

lemma-<E : $\forall \{x\} \{y\} \to (\# \ x)$ <E $(\# \ y) \to x < y$
trans<E : Trans _<E_

lemma-=E : $\forall \{x\} \{y\} \to (\# \ x)$ =E $(\# \ y) \to x == y$
sym=E : Symmetric _=E_
trans=E : Trans _=E_

respE : _<E_ $\text{Respects}_2$ _=E_

cmpE : Cmp {expanded $A$} _<E_ _=E_

$\_\leq\_$ : $\mathrm{Rel}_2$ (expanded $A$)
$\_\leq\_$ = _<=_ {expanded $A$} {_<E_} {_=E_}

trans≤ : Trans _≤_
trans≤ = trans<= respE sym=E trans=E trans<E

resp≤ : _≤_ $\mathrm{Respects}_2$ _=E_
resp≤ = resp<= respE trans=E sym=E

$$\text{minE} : (x\ y : \text{expanded}\ A) \to \text{expanded}\ A$$
$$\text{minE} = \text{min cmpE}$$
$$\text{min3E} : (\text{expanded}\ A) \to (\text{expanded}\ A) \to (\text{expanded}\ A) \to (\text{expanded}\ A)$$
$$\text{min3E}\ x\ y\ z = \text{min3 cmpE}\ x\ y\ z$$

$$\text{lemma-<=minE} : \forall\ \{a\ b\ c\} \to a \leq b \to a \leq c \to a \leq (\text{minE}\ b\ c)$$

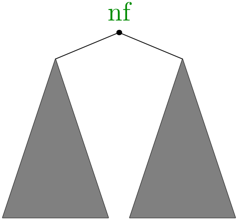
$$\text{lemma-<=min3E} : \forall\ \{x\ a\ b\ c\} \to x \leq a \to x \leq b \to x \leq c \to x \leq (\text{min3E}\ a\ b\ c)$$

# Heap

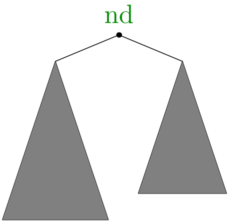
```
data HeapState : Set where
full almost : HeapState

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

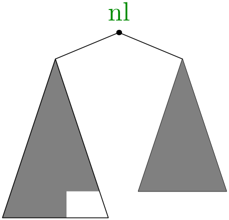
$$
\begin{aligned}
&\mathsf{nf} : \forall\ \{n\}\ \{x\ y\} \to (p : A) \\
&\quad \to (i : (\#\ p) \leq x) \to (j : (\#\ p) \leq y) \\
&\quad \to (a : \mathsf{Heap}\ x\ n\ \mathsf{full}) \\
&\quad \to (b : \mathsf{Heap}\ y\ n\ \mathsf{full}) \texttt{ - a b одной высоты} \\
&\quad \to \mathsf{Heap}\ (\#\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



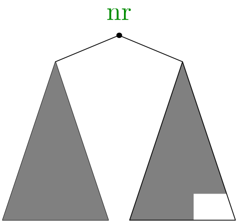
$$\mathrm{nd} : \forall\ \{n\}\ \{x\ y\} \to (p : A)$$
$$\to (i : (\#\ p) \leq x) \to (j : (\#\ p) \leq y)$$
$$\to (a : \mathrm{Heap}\ x\ (\mathrm{succ}\ n)\ \mathrm{full})$$
$$\to (b : \mathrm{Heap}\ y\ n\ \mathrm{full})$$ `- a b разной высоты`
$$\to \mathrm{Heap}\ (\#\ p)\ (\mathrm{succ}\ (\mathrm{succ}\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



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



$$
\begin{aligned}
&\mathrm{nr} : \forall\ \{n\}\ \{x\ y\} \to (p : A)\\
&\to (i : (\#\ p) \le x) \to (j : (\#\ p) \le y)\\
&\to (a : \mathrm{Heap}\ x\ (\mathrm{succ}\ n)\ \mathrm{full})\ \texttt{- a - полная}\\
&\to (b : \mathrm{Heap}\ y\ (\mathrm{succ}\ n)\ \mathrm{almost})\\
&\to \mathrm{Heap}\ (\#\ p)\ (\mathrm{succ}\ (\mathrm{succ}\ n))\ \mathrm{almost}
\end{aligned}
$$



## Вставка

Вставка в полную кучу

$$\text{finsert} : \forall \{h\ m\} \to (z : A) \to \text{Heap}\ m\ h\ \text{full} \to \Sigma\ \text{HeapState}\ (\text{Heap}\ (\text{minE}\ m\ (\#\ z))\ (\text{succ}\ h))$$

Вставка в неполную кучу

$$\text{ainsert} : \forall \{h\ m\} \to (z : A) \to \text{Heap}\ m\ h\ \text{almost} \to \Sigma\ \text{HeapState}\ (\text{Heap}\ (\text{minE}\ m\ (\#\ z))\ h)$$

Слияние двух полных куч одной высоты

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

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

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

Слияние поддеревьев nd

$$\begin{aligned}&\text{ndmerge} : \forall \{x\ y\ h\}\\&\to \text{Heap}\ x\ (\text{succ}\ (\text{succ}\ h))\ \text{full}\\&\to \text{Heap}\ y\ (\text{succ}\ h)\ \text{full}\\&\to \text{Heap}\ (\text{minE}\ x\ y)\ (\text{succ}\ (\text{succ}\ (\text{succ}\ h)))\ \text{almost}\end{aligned}$$

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

```
afmerge : ∀ {h x 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

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

Извлечение минимума из неполной кучи

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

# Заключение

- Разработаны типы данных для структуры данных и инвариантов
- Реализованы функции обработки
- Доказано сохранение инвариантов

Спасибо за внимание!