Tópicos de Análise de Algoritmos

Análise amortizada

Análise do Union-Find

CLRS cap 21

Coleção de conjuntos disjuntos

Queremos uma ED boa para representar uma
partição de um conjunto, e as seguintes operações sobre a partição:

Coleção de conjuntos disjuntos

Queremos uma ED boa para representar uma partição de um conjunto, e as seguintes operações sobre a partição:

- MakeSet(x): cria um conjunto unitário com o elemento x;
- FindSet(x): devolve o identificador do conjunto da partição que contém x;
- Union(x, y): substitui os conjuntos da partição que contêm x e y pela união deles.

Coleção de conjuntos disjuntos

Queremos uma ED boa para representar uma
partição de um conjunto, e as seguintes operações sobre a partição:

- MakeSet(x): cria um conjunto unitário com o elemento x;
- FindSet(x): devolve o identificador do conjunto da partição que contém x;
- Union(x, y): substitui os conjuntos da partição que contêm x e y pela união deles.

O identificador de um conjunto é um elemento do conjunto:
o seu representante.

Coleção de conjuntos disjuntos

Queremos uma ED boa para representar uma
partição de um conjunto, e as seguintes operações sobre a partição:

- MakeSet(x): cria um conjunto unitário com o elemento x;
- FindSet(x): devolve o identificador do conjunto da partição que contém x;
- Union(x, y): substitui os conjuntos da partição que contêm x e y pela união deles.

O identificador de um conjunto é um elemento do conjunto:
o seu representante.

Como podemos armazenar cada conjunto da partição?

Union-Find: florestas enraizadas

Make-Set (x)

$\text{pai}[x] \leftarrow x$

$\text{rank}[x] \leftarrow 0$ $\quad \triangleright$ altura da árvore

Union-Find: florestas enraizadas

Make-Set (x)

$\quad pai[x] \leftarrow x$

$\quad rank[x] \leftarrow 0 \qquad \triangleright$ altura da árvore

Heurística dos tamanhos:
no union, pendura a árvore mais baixa na mais alta

Union $(x, y) \qquad \triangleright$ x e y representantes distintos

$\quad$ **se** $rank[x] \geq rank[y]$

$\qquad$ **então** $pai[y] \leftarrow x$

$\qquad\qquad$ **se** $rank[x] = rank[y]$

$\qquad\qquad\quad$ **então** $rank[x] \leftarrow rank[x] + 1$

$\qquad$ **senão** $pai[x] \leftarrow y$

Union-Find: florestas enraizadas

Make-Set (x)

$\quad \text{pai}[x] \leftarrow x$

$\quad \text{rank}[x] \leftarrow 0$ $\qquad \triangleright$ altura da árvore

Heurística dos tamanhos:
no union, pendura a árvore mais baixa na mais alta

Union (x, y) $\qquad \triangleright$ x e y representantes distintos

$\quad$ **se** $\text{rank}[x] \geq \text{rank}[y]$

$\qquad$ **então** $\text{pai}[y] \leftarrow x$

$\qquad\qquad$ **se** $\text{rank}[x] = \text{rank}[y]$

$\qquad\qquad\quad$ **então** $\text{rank}[x] \leftarrow \text{rank}[x] + 1$

$\qquad$ **senão** $\text{pai}[x] \leftarrow y$

Note que o $\text{rank}[x]$ só é alterado enquanto x é raiz de uma árvore.

Union-Find: florestas enraizadas

Heurística da compressão dos caminhos:
quando busca um representante, aproveita para comprimir a árvore.

```
Find (x)
    if pai[x] ≠ x
        então pai[x] ← Find (pai[x])
    devolva pai[x]
```

Union-Find: florestas enraizadas

Heurística da compressão dos caminhos:
quando busca um representante, aproveita para comprimir a árvore.

```
Find (x)
    if pai[x] ≠ x
        então pai[x] ← Find (pai[x])
    devolva pai[x]
```

Consumo *amortizado* de tempo de cada operação:

$$\mathrm{O}(\lg^* n),$$

onde $\lg^* n$ é o número de vezes que temos que aplicar
o lg até atingir um número menor ou igual a 1.

Union-Find: florestas enraizadas

Heurística da compressão dos caminhos:
quando busca um representante, aproveita para comprimir a árvore.

```
Find (x)
    if pai[x] ≠ x
        então pai[x] ← Find (pai[x])
    devolva pai[x]
```

Consumo *amortizado* de tempo de cada operação:

$$\mathrm{O}(\lg^* n),$$

onde $\lg^* n$ é o número de vezes que temos que aplicar
o lg até atingir um número menor ou igual a 1.

Na verdade, é melhor do que isso,
e há uma análise justa, conforme discutido em aula.

Função log-estrela

$\lg^* n$ é o menor k tal que

$$\underbrace{\lg \lg \ldots \lg}_{k} n \leq 1$$

Função log-estrela

$\lg^* n$ é o menor k tal que

$$\underbrace{\lg \lg \ldots \lg}_{k} n \leq 1$$

n	$\lg^* n$
1	0
2	1
3	2
4	2
5	3
$\vdots$	$\vdots$
15	3
16	3
17	4
$\vdots$	$\vdots$
65535	4
65536	4
65537	5
$\vdots$	$\vdots$
$\underbrace{1000000000000000 \cdots 000000000000}_{80}$	5
$\vdots$	$\vdots$

Union-Find

```
Make-Set (x)
    pai[x] ← x
    rank[x] ← 0

Find (x)
    if pai[x] ≠ x
        então pai[x] ← Find (pai[x])
    devolva pai[x]

Union (x, y)        ▷ x e y representantes distintos
    se rank[x] ≥ rank[y]
        então  pai[y] ← x
            se rank[x] = rank[y]
                então rank[x] ← rank[x] + 1
        senão pai[x] ← y
```

Union-Find

```
Union (x, y)
    x' ← Find (x)
    y' ← Find (y)
    se x' ≠ y'
        então Link (x', y')
```

```
Link (x, y)        ▷ x e y representantes distintos
    se rank[x] ≥ rank[y]
        então pai[y] ← x
            se rank[x] = rank[y]
                então rank[x] ← rank[x] + 1
        senão pai[x] ← y
```

Análise

Dada sequência de makeset, findset e union,
converta-a em uma sequência de makeset, findset e link.

Análise

Dada sequência de makeset, findset e union,
converta-a em uma sequência de makeset, findset e link.

Sequência de m operações makeset, findset e link
das quais n são makeset.

Análise

Dada sequência de makeset, findset e union,
converta-a em uma sequência de makeset, findset e link.

Sequência de m operações makeset, findset e link
das quais n são makeset.

Custo amortizado de cada operação: $\mathrm{O}(\lg^* n)$.

Análise

Dada sequência de makeset, findset e union,
converta-a em uma sequência de makeset, findset e link.

Sequência de m operações makeset, findset e link
das quais n são makeset.

Custo amortizado de cada operação: $\mathrm{O}(\lg^* n)$.

Seja $\lg^{(1)} x = \lg x$.

Para $i \geq 2$, seja $\lg^{(i)} x = \lg(\lg^{(i-1)} x)$.

Análise

Dada sequência de makeset, findset e union,
converta-a em uma sequência de makeset, findset e link.

Sequência de m operações makeset, findset e link
das quais n são makeset.

Custo amortizado de cada operação: $\mathrm{O}(\lg^* n)$.

Seja $\lg^{(1)} x = \lg x$.

Para $i \geq 2$, seja $\lg^{(i)} x = \lg(\lg^{(i-1)} x)$.

A função $\lg^* n$ é definida da seguinte maneira:

$$\lg^* n \;=\; \min\{i : \lg^{(i)} n \leq 1\}.$$

Análise: a função potencial

Para cada nó x da floresta, o grupo de x é o conjunto

$$G(x) \;=\; \{y \;:\; \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}.$$

Análise: a função potencial

Para cada nó x da floresta, o grupo de x é o conjunto

$$G(x) \;=\; \{y \;:\; \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}.$$

Quantos grupos diferentes podemos ter?

Análise: a função potencial

Para cada nó x da floresta, o grupo de x é o conjunto

$$G(x) \;=\; \{y \;:\; \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}.$$

Quantos grupos diferentes podemos ter? Resposta: $1 + \lg^* n$

Análise: a função potencial

Para cada nó x da floresta, o grupo de x é o conjunto

$$G(x) \;=\; \{y \;:\; \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}.$$

Quantos grupos diferentes podemos ter? Resposta: $1 + \lg^* n$

Função potencial:

$$\Phi \;=\; |\{x \;:\; G(x) = G(\mathrm{pai}[x])\}|$$

(número de nós no mesmo grupo que seu pai)

Raízes contribuem com 1, logo Φ é pelo menos o número de árvores.

Análise: a função potencial

Para cada nó x da floresta, o grupo de x é o conjunto

$$G(x) \;=\; \{y \;:\; \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}.$$

Quantos grupos diferentes podemos ter? Resposta: $1 + \lg^* n$

Função potencial:

$$\Phi \;=\; |\{x \;:\; G(x) = G(\mathrm{pai}[x])\}|$$

(número de nós no mesmo grupo que seu pai)

Raízes contribuem com 1, logo Φ é pelo menos o número de árvores.

Ideia: Se o potencial é o número de árvores, então a altura das árvores é no máximo $1 + \lg^* n$ e o custo do findset é $\mathrm{O}(\lg^* n)$.

Análise: a função potencial

Para cada nó x da floresta, o grupo de x é o conjunto

$$G(x) \;=\; \{y \;:\; \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}.$$

Quantos grupos diferentes podemos ter? Resposta: $1 + \lg^* n$

Função potencial:

$$\Phi \;=\; |\{x \;:\; G(x) = G(\mathrm{pai}[x])\}|$$

(número de nós no mesmo grupo que seu pai)

Raízes contribuem com 1, logo Φ é pelo menos o número de árvores.

Ideia: Se o potencial é o número de árvores, então a altura das árvores é no máximo $1 + \lg^* n$ e o custo do findset é $\mathrm{O}(\lg^* n)$. Se altura é maior, o potencial será maior também.

Análise: cálculo dos custos amortizados

Para cada nó x da floresta, o grupo de x é o conjunto

$$G(x) \; = \; \{y \; : \; \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}.$$

Função potencial:

$$\Phi \; = \; |\{x \; : \; G(x) = G(\mathrm{pai}[x])\}|$$

(número de nós no mesmo grupo que seu pai)

$\Phi \geq 0$ e o valor inicial $\Phi_0 = 0$.
(não há nenhum conjunto na coleção inicialmente)

Análise: cálculo dos custos amortizados

Para cada nó x da floresta, o grupo de x é o conjunto

$$G(x) \;=\; \{y \;:\; \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}.$$

Função potencial:

$$\Phi \;=\; |\{x \;:\; G(x) = G(\mathrm{pai}[x])\}|$$

(número de nós no mesmo grupo que seu pai)

$\Phi \geq 0$ e o valor inicial $\Phi_0 = 0$.
(não há nenhum conjunto na coleção inicialmente)

Φ_a: função potencial antes da operação
Φ_d: função potencial depois da operação

Análise: cálculo dos custos amortizados

$$G(x) \;=\; \{y \;:\; \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}$$

Função potencial: $\Phi \;=\; |\{x \;:\; G(x) = G(\mathrm{pai}[x])\}|$

Φ_a: função potencial antes da operação

Φ_d: função potencial depois da operação

Análise: cálculo dos custos amortizados

$G(x) \;=\; \{y \;:\; \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}$

Função potencial: $\Phi \;=\; |\{x \;:\; G(x) = G(\mathrm{pai}[x])\}|$

Φ_a: função potencial antes da operação
Φ_d: função potencial depois da operação

O custo amortizado de cada operação é:

- makeset(x): $\hat{c} \;=\; c + \Phi_d - \Phi_a \;=\; 1 + 1 \;=\; 2.$

Análise: cálculo dos custos amortizados

$G(x) = \{y : \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}$

Função potencial: $\Phi = |\{x : G(x) = G(\mathrm{pai}[x])\}|$

Φ_a: função potencial antes da operação
Φ_d: função potencial depois da operação

O custo amortizado de cada operação é:

- makeset(x): $\hat{c} = c + \Phi_d - \Phi_a = 1 + 1 = 2.$
- link(x, y): $\hat{c} = c + \Phi_d - \Phi_a = 1 + \Phi_d - \Phi_a \leq 1$ pois

Análise: cálculo dos custos amortizados

$G(x) \;=\; \{y \;:\; \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}$

Função potencial: $\Phi \;=\; |\{x \;:\; G(x) = G(\mathrm{pai}[x])\}|$

Φ_a: função potencial antes da operação
Φ_d: função potencial depois da operação

O custo amortizado de cada operação é:

- makeset(x): $\hat{c} \;=\; c + \Phi_d - \Phi_a \;=\; 1 + 1 \;=\; 2.$
- link(x, y): $\hat{c} \;=\; c + \Phi_d - \Phi_a \;=\; 1 + \Phi_d - \Phi_a \;\leq\; 1$ pois

se $\mathrm{pai}[x]$ é alterado para nó x
e novo $\mathrm{pai}[x]$ é tal que $G(x) \neq G(\mathrm{pai}[x])$, então $\Phi_d - \Phi_a = -1$;

Análise: cálculo dos custos amortizados

$$G(x) \;=\; \{y \;:\; \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}$$

Função potencial: $\Phi \;=\; |\{x \;:\; G(x) = G(\mathrm{pai}[x])\}|$

Φ_a: função potencial antes da operação
Φ_d: função potencial depois da operação

O custo amortizado de cada operação é:

- makeset(x): $\hat{c} \;=\; c + \Phi_d - \Phi_a \;=\; 1 + 1 \;=\; 2.$
- link(x, y): $\hat{c} \;=\; c + \Phi_d - \Phi_a \;=\; 1 + \Phi_d - \Phi_a \;\leq\; 1$ pois

se $\mathrm{pai}[x]$ é alterado para nó x
e novo $\mathrm{pai}[x]$ é tal que $G(x) \neq G(\mathrm{pai}[x])$, então $\Phi_d - \Phi_a = -1$;
se novo $\mathrm{pai}[x]$ é tal que $G(x) = G(\mathrm{pai}[x])$, então $\Phi_d - \Phi_a = 0$.

Análise: cálculo dos custos amortizados

$G(x) \;=\; \{y \;:\; \lg^*(\mathrm{rank}[y]) = \lg^*(\mathrm{rank}[x])\}$

Função potencial: $\Phi \;=\; |\{x \;:\; G(x) = G(\mathrm{pai}[x])\}|$

Φ_a: função potencial antes da operação
Φ_d: função potencial depois da operação

O custo amortizado de cada operação é:

- makeset(x): $\hat{c} \;=\; c + \Phi_d - \Phi_a \;=\; 1 + 1 \;=\; 2.$
- link(x, y): $\hat{c} \;=\; c + \Phi_d - \Phi_a \;=\; 1 + \Phi_d - \Phi_a \;\leq\; 1$ pois

se $\mathrm{pai}[x]$ é alterado para nó x
e novo $\mathrm{pai}[x]$ é tal que $G(x) \neq G(\mathrm{pai}[x])$, então $\Phi_d - \Phi_a = -1$;
se novo $\mathrm{pai}[x]$ é tal que $G(x) = G(\mathrm{pai}[x])$, então $\Phi_d - \Phi_a = 0$.

Ademais, incremento de $\mathrm{rank}[y]$ pode fazer o potencial diminuir.

Custo amortizado do findset(x)

$z = \text{findset}(x)$

$k = $ número de nós do caminho P de x a z

Custo amortizado do findset(x)

$z = \text{findset}(x)$
k = número de nós do caminho P de x a z

Então $c = k$ e $\hat{c} = k + \Phi_d - \Phi_a$.

Custo amortizado do findset(x)

$z = \text{findset}(x)$
k = número de nós do caminho P de x a z

Então $c = k$ e $\hat{c} \;=\; k + \Phi_d - \Phi_a$.

Quanto vale $\Phi_d - \Phi_a$?

Custo amortizado do findset(x)

$z = \text{findset}(x)$
$k =$ número de nós do caminho P de x a z

Então $c = k$ e $\hat{c} = k + \Phi_d - \Phi_a$.

Quanto vale $\Phi_d - \Phi_a$?

Partição dos arcos da floresta:
para cada vértice u, o arco entre u e $pai[u]$ é

Custo amortizado do findset(x)

z = findset(x)
k = número de nós do caminho P de x a z

Então $c = k$ e $\hat{c} = k + \Phi_d - \Phi_a$.

Quanto vale $\Phi_d - \Phi_a$?

Partição dos arcos da floresta:
para cada vértice u, o arco entre u e $pai[u]$ é

- vermelho se $G(\mathrm{pai}[u]) = G(u) = G(r)$, onde r é a raiz da árvore em que u se encontra.
- azul se $G(\mathrm{pai}[u]) \neq G(u)$.
- amarelo se $G(\mathrm{pai}[u]) = G(u) \neq G(r)$.

Custo amortizado do findset(x)

z = findset(x)
k = número de nós do caminho P de x a z

Então $c = k$ e $\hat{c} = k + \Phi_d - \Phi_a$.

Quanto vale $\Phi_d - \Phi_a$?

Partição dos arcos da floresta:
para cada vértice u, o arco entre u e $pai[u]$ é

- vermelho se $G(\mathrm{pai}[u]) = G(u) = G(r)$,
  onde r é a raiz da árvore em que u se encontra.
- azul se $G(\mathrm{pai}[u]) \neq G(u)$.
- amarelo se $G(\mathrm{pai}[u]) = G(u) \neq G(r)$.

Φ: número de arcos vermelhos mais amarelos.

Custo amortizado do findset(x)

$z = \text{findset}(x) \qquad k = \text{comprimento do caminho } P \text{ de } x \text{ a } z$

$\hat{c} \;=\; k + \Phi_d - \Phi_a.$

Partição dos arcos da floresta:

para cada vértice u, o arco de u para $\mathrm{pai}[u]$ é

- vermelho se $G(\mathrm{pai}[u]) = G(u) = G(r)$, onde r é a raiz da árvore em que u se encontra.
- azul se $G(\mathrm{pai}[u]) \neq G(u)$.
- amarelo se $G(\mathrm{pai}[u]) = G(u) \neq G(r)$.

Custo amortizado do findset(x)

z = findset(x) k = comprimento do caminho P de x a z

$\hat{c} \ = \ k + \Phi_d - \Phi_a.$

Partição dos arcos da floresta:
para cada vértice u, o arco de u para $\mathrm{pai}[u]$ é

- vermelho se $G(\mathrm{pai}[u]) = G(u) = G(r)$,
  onde r é a raiz da árvore em que u se encontra.
- azul se $G(\mathrm{pai}[u]) \neq G(u)$.
- amarelo se $G(\mathrm{pai}[u]) = G(u) \neq G(r)$.

Ao final de findset(x), nós de P têm z como pai.
(Arcos amarelos viram azuis.)

Custo amortizado do findset(x)

$z = \text{findset}(x)$ $\quad$ $k =$ comprimento do caminho P de x a z

$\hat{c} \;=\; k + \Phi_d - \Phi_a.$

Partição dos arcos da floresta:
para cada vértice u, o arco de u para $\mathrm{pai}[u]$ é

- vermelho se $G(\mathrm{pai}[u]) = G(u) = G(r)$,
  onde r é a raiz da árvore em que u se encontra.
- azul se $G(\mathrm{pai}[u]) \neq G(u)$.
- amarelo se $G(\mathrm{pai}[u]) = G(u) \neq G(r)$.

Ao final de findset(x), nós de P têm z como pai.
(Arcos amarelos viram azuis.)

Φ decresce do número de arcos amarelos em P.

Custo amortizado do findset(x)

$z = \text{findset}(x)$ $\quad k = $ comprimento do caminho P de x a z

$\hat{c} \;=\; k + \Phi_d - \Phi_a.$

Partição dos arcos da floresta:
para cada vértice u, o arco de u para $\text{pai}[u]$ é

- vermelho se $G(\text{pai}[u]) = G(u) = G(r)$,
  onde r é a raiz da árvore em que u se encontra.
- azul se $G(\text{pai}[u]) \neq G(u)$.
- amarelo se $G(\text{pai}[u]) = G(u) \neq G(r)$.

Ao final de findset(x), nós de P têm z como pai.
(Arcos amarelos viram azuis.)

Φ decresce do número de arcos amarelos em P.

Então $\hat{c} = v + a$, onde v é o número de arcos vermelhos e
a é o número de arcos azuis em P.

Custo amortizado do findset(x)

rank cresce estritamente ao subirmos nas árvores.

Custo amortizado do findset(x)

rank cresce estritamente ao subirmos nas árvores.

Arco azul: passamos de um grupo para outro.

Custo amortizado do findset(x)

rank cresce estritamente ao subirmos nas árvores.

Arco azul: passamos de um grupo para outro.

Número de arcos azuis no caminho de qualquer nó a uma raiz é no máximo o número de grupos, que é $\leq 1 + \lg^* n$.

Custo amortizado do findset(x)

rank cresce estritamente ao subirmos nas árvores.

Arco azul: passamos de um grupo para outro.

Número de arcos azuis no caminho de qualquer nó a uma raiz é no máximo o número de grupos, que é $\leq 1 + \lg^* n$.

rank assume valores entre 0 e $\lg n$,
assim os grupos vão de 0 a $\lg^*(\lg n) \leq \lg^* n$.

Custo amortizado do findset(x)

rank cresce estritamente ao subirmos nas árvores.

Arco azul: passamos de um grupo para outro.

Número de arcos azuis no caminho de qualquer nó a uma raiz é no máximo o número de grupos, que é $\leq 1 + \lg^* n$.

rank assume valores entre 0 e $\lg n$,
assim os grupos vão de 0 a $\lg^*(\lg n) \leq \lg^* n$.

Ou seja, $a \leq 1 + \lg^* n$
e o custo amortizado do findset fica

$$\hat{c} \;=\; v + a \;\leq\; v + 1 + \lg^* n.$$

O que falta?

$\hat{c}_i$: custo amortizado da i-ésima operação
c_i: custo real da i-ésima operação

O que falta?

$\hat{c}_i$: custo amortizado da i-ésima operação
c_i: custo real da i-ésima operação

Queremos mostrar que

$$\sum_{i=1}^{m} c_i = \mathrm{O}(m \lg^* n).$$

O que falta?

$\hat{c}_i$: custo amortizado da i-ésima operação
c_i: custo real da i-ésima operação

Queremos mostrar que

$$\sum_{i=1}^{m} c_i = \mathrm{O}(m \lg^* n).$$

Assim, o custo amortizado por operação é $\mathrm{O}(\lg^* n)$.

O que falta?

$\hat{c}_i$: custo amortizado da i-ésima operação
c_i: custo real da i-ésima operação

Queremos mostrar que

$$\sum_{i=1}^{m} c_i \; = \; \mathrm{O}(m \lg^* n).$$

Assim, o custo amortizado por operação é $\mathrm{O}(\lg^* n)$.

Lembre-se que $\sum_{i=1}^{m} \hat{c}_i \; = \; \sum_{i=1}^{m} c_i + \Phi_f - \Phi_0$,
onde Φ_f é o potencial final da floresta.

O que falta?

$\hat{c}_i$: custo amortizado da i-ésima operação
c_i: custo real da i-ésima operação

Queremos mostrar que

$$\sum_{i=1}^{m} c_i \;=\; \mathrm{O}(m \lg^* n).$$

Assim, o custo amortizado por operação é $\mathrm{O}(\lg^* n)$.

Lembre-se que $\sum_{i=1}^{m} \hat{c}_i \;=\; \sum_{i=1}^{m} c_i + \Phi_f - \Phi_0$,
onde Φ_f é o potencial final da floresta.

Como $\Phi_0 = 0$ e $\Phi_f \geq 0$, temos que $\sum_{i=1}^{m} c_i \;\leq\; \sum_{i=1}^{m} \hat{c}_i$.

Delimitando a soma dos custos amortizados

Por operação:

$$
\begin{array}{lcl}
\text{makeset}(x): & \hat{c} = c + \Phi_d - \Phi_a = 2. \\
\text{link}(x, y): & \hat{c} = c + \Phi_d - \Phi_a \leq 1. \\
\text{findset}(x): & \hat{c} = v + a \leq v + 1 + \lg^* n.
\end{array}
$$

Delimitando a soma dos custos amortizados

Por operação:

$$
\begin{array}{llllll}
\text{makeset}(x): & \hat{c} & = & c + \Phi_d - \Phi_a & = & 2. \\
\text{link}(x, y): & \hat{c} & = & c + \Phi_d - \Phi_a & \leq & 1. \\
\text{findset}(x): & \hat{c} & = & v + a & \leq & v + 1 + \lg^* n.
\end{array}
$$

Portanto, para as m operações,

$$\sum_{i=1}^{m} \hat{c}_i \leq 2(m - f) + f(1 + \lg^* n) + \sum_{i=1}^{m} v_i,$$

onde f é o número de operações findset na sequência,
v_i é 0 se a i-ésima operação não é findset e
é o número de arcos vermelhos em P se é findset(x).

Delimitando a soma dos custos amortizados

$$\sum_{i=1}^{m} \hat{c}_i \;\leq\; 2(m - f) + f(1 + \lg^* n) + \sum_{i=1}^{m} v_i,$$

Delimitando a soma dos custos amortizados

$$\sum_{i=1}^{m} \hat{c}_i \;\leq\; 2(m-f) + f(1+\lg^* n) + \sum_{i=1}^{m} v_i,$$

Para delimitar $\sum_{i=1}^{m} v_i$, lembre-se que
um arco é vermelho se $G(\mathrm{pai}[u]) = G(u) = G(r)$.

Delimitando a soma dos custos amortizados

$$\sum_{i=1}^{m} \hat{c}_i \;\leq\; 2(m - f) + f(1 + \lg^* n) + \sum_{i=1}^{m} v_i,$$

Para delimitar $\sum_{i=1}^{m} v_i$, lembre-se que
um arco é vermelho se $G(\mathrm{pai}[u]) = G(u) = G(r)$.

Se um u deixa de ser ponta inferior de arco vermelho,
não volta a ser ponta inferior de arco vermelho.

Delimitando a soma dos custos amortizados

$$\sum_{i=1}^{m} \hat{c}_i \;\leq\; 2(m-f) + f(1+\lg^* n) + \sum_{i=1}^{m} v_i,$$

Para delimitar $\sum_{i=1}^{m} v_i$, lembre-se que
um arco é vermelho se $G(\mathrm{pai}[u]) = G(u) = G(r)$.

Se um u deixa de ser ponta inferior de arco vermelho,
não volta a ser ponta inferior de arco vermelho.

Todos os nós de P,
exceto os dois últimos (a raiz e o filho dela no caminho),
têm o campo pai alterado no findset.

Delimitando a soma dos custos amortizados

$$\sum_{i=1}^{m} \hat{c}_i \;\leq\; 2(m - f) + f(1 + \lg^* n) + \sum_{i=1}^{m} v_i,$$

Para delimitar $\sum_{i=1}^{m} v_i$, lembre-se que
um arco é vermelho se $G(\mathrm{pai}[u]) = G(u) = G(r)$.

Se um u deixa de ser ponta inferior de arco vermelho,
não volta a ser ponta inferior de arco vermelho.

Todos os nós de P,
exceto os dois últimos (a raiz e o filho dela no caminho),
têm o campo pai alterado no findset.

Se o pai de um nó é alterado, rank do seu pai aumenta

Delimitando a soma dos custos amortizados

$$\sum_{i=1}^{m} \hat{c}_i \;\leq\; 2(m-f) + f(1+\lg^* n) + \sum_{i=1}^{m} v_i,$$

Para delimitar $\sum_{i=1}^{m} v_i$, lembre-se que
um arco é vermelho se $G(\mathrm{pai}[u]) = G(u) = G(r)$.

Se um u deixa de ser ponta inferior de arco vermelho,
não volta a ser ponta inferior de arco vermelho.

Todos os nós de P,
exceto os dois últimos (a raiz e o filho dela no caminho),
têm o campo pai alterado no findset.

Se o pai de um nó é alterado, rank do seu pai aumenta
pois rank cresce estritamente ao subirmos nas árvores.

Delimitando a soma dos custos amortizados

y: ponta inferior de um arco vermelho

Enquanto isso, o campo $\mathrm{pai}[y]$ sofre alteração quantas vezes?

Delimitando a soma dos custos amortizados

y: ponta inferior de um arco vermelho

Enquanto isso, o campo $\mathrm{pai}[y]$ sofre alteração quantas vezes?

Seja $t(s) = \min\{k \in Z : \lg^* k \geq s\}$ (tipo de inversa do $\lg^* n$).

Em palavras, $t(s)$ é o menor rank de alguém do grupo s.

Exemplos: $t(3) = 5$, $t(4) = 17$ e $t(5) = 65537$.

k	$\lg^* k$
1	0
2	1
3	2
4	2
5	3
$\vdots$	$\vdots$
15	3
16	3
17	4
$\vdots$	$\vdots$
65536	4
65537	5
$\vdots$	$\vdots$

Delimitando a soma dos custos amortizados

y: ponta inferior de um arco vermelho

Enquanto isso, o campo $\mathrm{pai}[y]$ sofre alteração quantas vezes?

Seja $t(s) = \min\{k \in Z : \lg^* k \geq s\}$ (tipo de inversa do $\lg^* n$).

Em palavras, $t(s)$ é o menor rank de alguém do grupo s.

Exemplos: $t(3) = 5$, $t(4) = 17$ e $t(5) = 65537$.

Se $g = \lg^*(\mathrm{rank}[y])$, então $\mathrm{pai}[y]$ se altera e
y permanece ponta inferior de arco vermelho
no máximo $t(g+1) - t(g) \leq t(g+1)$ vezes.

Delimitando a soma dos custos amortizados

y: ponta inferior de um arco vermelho

Enquanto isso, o campo $\mathrm{pai}[y]$ sofre alteração quantas vezes?

Seja $t(s) = \min\{k \in Z : \lg^* k \geq s\}$ (tipo de inversa do $\lg^* n$).

Em palavras, $t(s)$ é o menor rank de alguém do grupo s.

Exemplos: $t(3) = 5$, $t(4) = 17$ e $t(5) = 65537$.

Se $g = \lg^*(\mathrm{rank}[y])$, então $\mathrm{pai}[y]$ se altera e
y permanece ponta inferior de arco vermelho
no máximo $t(g+1) - t(g) \leq t(g+1)$ vezes.

Logo

$$\sum_{i=1}^{m} v_i \leq 2f + \sum_{g=1}^{\lg^* n} |\{y : \lg^*(\mathrm{rank}[y]) = g\}|\, t(g+1).$$

Delimitando a soma dos custos amortizados

y: ponta inferior de um arco vermelho

Enquanto isso, o campo $\mathrm{pai}[y]$ sofre alteração quantas vezes?

Seja $t(s) = \min\{k \in Z : \lg^* k \geq s\}$ (tipo de inversa do $\lg^* n$).

Em palavras, $t(s)$ é o menor rank de alguém do grupo s.

Exemplos: $t(3) = 5$, $t(4) = 17$ e $t(5) = 65537$.

Se $g = \lg^*(\mathrm{rank}[y])$, então $\mathrm{pai}[y]$ se altera e
y permanece ponta inferior de arco vermelho
no máximo $t(g+1) - t(g) \leq t(g+1)$ vezes.

Logo

$$\sum_{i=1}^{m} v_i \leq 2f + \sum_{g=1}^{\lg^* n} |\{y : \lg^*(\mathrm{rank}[y]) = g\}| \, t(g+1).$$

$2f$: por 2 possíveis arcos vermelhos da raiz e de seu filho no caminho.

Delimitando a soma dos custos amortizados

$$\sum_{i=1}^{m} v_i \;\leq\; 2f + \sum_{g=1}^{\lg^* n} |\{y : \lg^*(\mathrm{rank}[y]) = g\}|\, t(g+1).$$

Agora queremos delimitar $|\{y : \lg^*(\mathrm{rank}[y]) = g\}|$.

Delimitando a soma dos custos amortizados

$$\sum_{i=1}^{m} v_i \;\le\; 2f + \sum_{g=1}^{\lg^* n} |\{y : \lg^*(\mathrm{rank}[y]) = g\}|\, t(g+1).$$

Agora queremos delimitar $|\{y : \lg^*(\mathrm{rank}[y]) = g\}|$.

O número de nós y tais que $\mathrm{rank}[y] = r$ é no máximo $n/2^r$.
(Subárvore com raiz de rank r tem pelo menos 2^r nós.)

Delimitando a soma dos custos amortizados

$$\sum_{i=1}^{m} v_i \;\leq\; 2f + \sum_{g=1}^{\lg^* n} |\{y : \lg^*(\mathrm{rank}[y]) = g\}|\, t(g+1).$$

Agora queremos delimitar $|\{y : \lg^*(\mathrm{rank}[y]) = g\}|$.

O número de nós y tais que $\mathrm{rank}[y] = r$ é no máximo $n/2^r$.
(Subárvore com raiz de rank r tem pelo menos 2^r nós.)

Além disso, $t(g) = \min\{k \in Z : \lg^* k \geq g\} \leq \mathrm{rank}[y]$.

Delimitando a soma dos custos amortizados

$$\sum_{i=1}^{m} v_i \;\leq\; 2f + \sum_{g=1}^{\lg^* n} |\{y : \lg^*(\mathrm{rank}[y]) = g\}| t(g+1).$$

Agora queremos delimitar $|\{y : \lg^*(\mathrm{rank}[y]) = g\}|$.

O número de nós y tais que $\mathrm{rank}[y] = r$ é no máximo $n/2^r$.
(Subárvore com raiz de rank r tem pelo menos 2^r nós.)

Além disso, $t(g) = \min\{k \in Z : \lg^* k \geq g\} \leq \mathrm{rank}[y]$. Logo

$$|\{y : \lg^*(\mathrm{rank}[y]) = g\}| \;=\; \sum_{r=t(g)}^{t(g+1)-1} |\{y : \mathrm{rank}[y] = r\}|$$

Delimitando a soma dos custos amortizados

$$\sum_{i=1}^{m} v_i \;\leq\; 2f + \sum_{g=1}^{\lg^* n} |\{y : \lg^*(\mathrm{rank}[y]) = g\}|\, t(g+1).$$

Agora queremos delimitar $|\{y : \lg^*(\mathrm{rank}[y]) = g\}|$.

O número de nós y tais que $\mathrm{rank}[y] = r$ é no máximo $n/2^r$.
(Subárvore com raiz de rank r tem pelo menos 2^r nós.)

Além disso, $t(g) = \min\{k \in Z : \lg^* k \geq g\} \leq \mathrm{rank}[y]$. Logo

$$\begin{aligned} |\{y : \lg^*(\mathrm{rank}[y]) = g\}| \;&=\; \sum_{r=t(g)}^{t(g+1)-1} |\{y : \mathrm{rank}[y] = r\}| \\ &<\; \sum_{r=t(g)}^{\infty} \frac{n}{2^r} \;=\; \frac{n}{2^{t(g)}} \sum_{r=0}^{\infty} \frac{1}{2^r} \end{aligned}$$

Delimitando a soma dos custos amortizados

$$\sum_{i=1}^{m} v_i \ \leq \ 2f + \sum_{g=1}^{\lg^* n} |\{y : \lg^*(\mathrm{rank}[y]) = g\}| t(g+1).$$

Agora queremos delimitar $|\{y : \lg^*(\mathrm{rank}[y]) = g\}|$.

O número de nós y tais que $\mathrm{rank}[y] = r$ é no máximo $n/2^r$.
(Subárvore com raiz de rank r tem pelo menos 2^r nós.)

Além disso, $t(g) = \min\{k \in Z : \lg^* k \geq g\} \leq \mathrm{rank}[y]$. Logo

$$\begin{aligned} |\{y : \lg^*(\mathrm{rank}[y]) = g\}| &= \sum_{r=t(g)}^{t(g+1)-1} |\{y : \mathrm{rank}[y] = r\}| \\ &< \sum_{r=t(g)}^{\infty} \frac{n}{2^r} = \frac{n}{2^{t(g)}} \sum_{r=0}^{\infty} \frac{1}{2^r} = \frac{2n}{2^{t(g)}}. \end{aligned}$$

Finalizando

Portanto concluímos que

$$
\begin{aligned}
\sum_{i=1}^{m} v_i &\leq 2f + \sum_{g=0}^{\lg^* n} |\{y : \lg^*(\mathrm{rank}[y]) = g\}| t(g+1) \\
&\leq 2f + 2n \sum_{g=0}^{\lg^* n} \frac{t(g+1)}{2^{t(g)}}.
\end{aligned}
$$

Finalizando

Portanto concluímos que

$$\begin{aligned}\sum_{i=1}^{m} v_i &\leq 2f + \sum_{g=0}^{\lg^* n} |\{y : \lg^*(\mathrm{rank}[y]) = g\}| t(g+1) \\ &\leq 2f + 2n \sum_{g=0}^{\lg^* n} \frac{t(g+1)}{2^{t(g)}}.\end{aligned}$$

Mas, $t(g+1) - 1 = 2^{t(g)-1}$, logo $t(g+1) = 2^{t(g)-1} + 1 \leq 2^{t(g)}$.

Finalizando

Portanto concluímos que

$$\begin{aligned}\sum_{i=1}^{m} v_i &\leq 2f + \sum_{g=0}^{\lg^* n} |\{y : \lg^*(\mathrm{rank}[y]) = g\}| t(g+1) \\ &\leq 2f + 2n \sum_{g=0}^{\lg^* n} \frac{t(g+1)}{2^{t(g)}}.\end{aligned}$$

Mas, $t(g+1) - 1 = 2^{t(g)-1}$, logo $t(g+1) = 2^{t(g)-1} + 1 \leq 2^{t(g)}$.

Então $\sum_{i=1}^{m} v_i \leq 2f + 2n(1 + \lg^* n)$ e

Finalizando

Portanto concluímos que

$$\begin{aligned}\sum_{i=1}^{m} v_i &\leq 2f + \sum_{g=0}^{\lg^* n} |\{y : \lg^*(\mathrm{rank}[y]) = g\}|\, t(g+1) \\ &\leq 2f + 2n \sum_{g=0}^{\lg^* n} \frac{t(g+1)}{2^{t(g)}}.\end{aligned}$$

Mas, $t(g+1) - 1 = 2^{t(g)-1}$, logo $t(g+1) = 2^{t(g)-1} + 1 \leq 2^{t(g)}$.

Então $\sum_{i=1}^{m} v_i \leq 2f + 2n(1 + \lg^* n)$ e

$\sum_{i=1}^{m} c_i \leq \sum_{i=1}^{m} \hat{c}_i \leq 2m + 2n + 2f + 2m \lg^* n = \mathrm{O}(m \lg^* n)$.

Finalizando

Portanto concluímos que

$$\begin{aligned}\sum_{i=1}^{m} v_i &\le 2f + \sum_{g=0}^{\lg^* n} |\{y : \lg^*(\mathrm{rank}[y]) = g\}| t(g+1) \\ &\le 2f + 2n \sum_{g=0}^{\lg^* n} \frac{t(g+1)}{2^{t(g)}}.\end{aligned}$$

Mas, $t(g+1) - 1 = 2^{t(g)-1}$, logo $t(g+1) = 2^{t(g)-1} + 1 \le 2^{t(g)}$.

Então $\sum_{i=1}^{m} v_i \ \le\ 2f + 2n(1 + \lg^* n)$ e

$\sum_{i=1}^{m} c_i \ \le\ \sum_{i=1}^{m} \hat{c}_i \ \le\ 2m + 2n + 2f + 2m \lg^* n \ =\ \mathrm{O}(m \lg^* n)$.

Logo o custo amortizado por operação é $\mathrm{O}(\lg^* n)$.