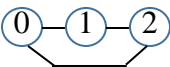


Como o union-find funciona?

Suponha um grafo: 

Antes de chamar essas funções, criamos um vetor chamado subconjunto, com o tamanho igual a quantidade de vértices do grafo, aonde cada posição do vetor representa um vértice. Inicialmente preenchemos esse vetor com -1:

Subconjunto[3]: posições: 0 1 2

Valor: -1 -1 -1

Função find (subconjunto (endereço do vetor), vértice a ser analisado)

Se (subconjunto[v] == -1)

Retorne v;

Retorne find (subconjunto, subconjunto[v]); // veja que essa função é recursiva

Vamos verificar cada uma das arestas, chamando essa função para cada vértice:

Aresta 0---1: vértice x= 0

vértice y= 1

vx= find (subconjunto, 0)

Veja que subconjunto[0] tem valor -1, portanto, a função retorna v, que é 0.

Vx= 0.

vy= find (subconjunto, 1)

Veja que subconjunto[1] tem valor -1, portanto, a função retorna v, que é 1.

Vy= 1.

Agora chamamos a função union:

Função union (subconjunto (endereço do vetor), vx, vy)

Vx_set= find (subconjunto, vx);

Vy_set= find (subconjunto, vy);

Se vx != vy // se vx = vy, então essa aresta forma um ciclo

Subconjunto[vx_set]= vy_set;

Sabemos que, ao chamar essa função para os vértices 0 e 1, $vx_set= 0$ e $vy_set= 1$, logo,

Subconjunto[0]= 1. E o vetor subconjunto ficará:

Subconjunto[3]: posições: 0 1 2

Valor: 1 -1 -1

Aresta 1---2: vértice $x= 1$

vértice $y= 2$

$vx= \text{find}(\text{subconjunto}, 1)$

Veja que subconjunto[1] tem valor -1, portanto, a função retorna v, que é 1.

$Vx= 1$.

$vy= \text{find}(\text{subconjunto}, 2)$

Veja que subconjunto[1] tem valor -1, portanto, a função retorna v, que é 2.

$Vy= 2$.

Chamamos a função union:

Sabemos que, ao chamar essa função para os vértices 1 e 2, $vx_set= 1$ e $vy_set= 2$, logo:

Subconjunto[1]= 2. E o vetor subconjunto ficará:

Subconjunto[3]: posições: 0 1 2

Valor: 1 2 -1

Aresta 0---2: vértice $x= 0$

vértice $y= 2$

$vx= \text{find}(\text{subconjunto}, 0)$

subconjunto[0]= 1 e diferente de -1, portanto:

$\text{find}(\text{subconjunto}, 1)$,

subconjunto[1]= 2 e diferente de -1, portanto:

$\text{find}(\text{subconjunto}, 2)$

subconjunto[2]= -1, então retorna v, que é 2, portanto, $vx= 2$.

$vy= \text{find}(\text{subconjunto}, 2)$

Veja que subconjunto[2] tem valor -1, portanto, a função retorna v, que é 2, $vy= 2$.

Nesse caso, v_x é igual a v_y ($2=2$), essa aresta forma ciclo e não faz parte da árvore geradora mínima.

Perceba que a função `union` serve para atualizar os dados do vetor subconjunto, mas somente quando a aresta analisada não forma ciclo.

Union e find possuem complexidade linear no pior caso, pois as árvores criadas para representar os subconjuntos podem ser inclinadas e podem se tornar como uma lista encadeada. Essas operações podem ser otimizadas para a complexidade $O(\log n)$, a ideia é anexar uma árvore de profundidade menor sob a raiz da árvore mais profunda, a essa técnica damos o nome de união por classificação.

A segunda otimização é no método `find`, através da compressão de caminho. Quando `find` é chamada para um vértice x , essa função percorre x até encontrar a raiz, e ela será dada como pai de x . Se x for raiz de uma subárvore, o caminho de todos os nós até a raiz, em x , também serão compactados.