Grafos

Árvores: Diâmetro

Prof. Edson Alves

Faculdade UnB Gama

Definição de diâmetro

Definição de diâmetro

Seja $G(V, E)$ um grafo. O diâmetro de G é igual ao maior dentre todos os tamanhos dos caminhos entre os pares de vértices $u, v \in V$.

Características do diâmetro

Características do diâmetro

- ⋆ O caminho cujo tamanho determina o diâmetro não é, necessariamente, único

Características do diâmetro

- $\star$ O caminho cujo tamanho determina o diâmetro não é, necessariamente, único

- $\star$ Computar todos os tamanho com Floyd-Warshall em $O(V^3)$ e determinar o maior dentre eles em $O(V^2)$ determina o diâmetro corretamente

Características do diâmetro

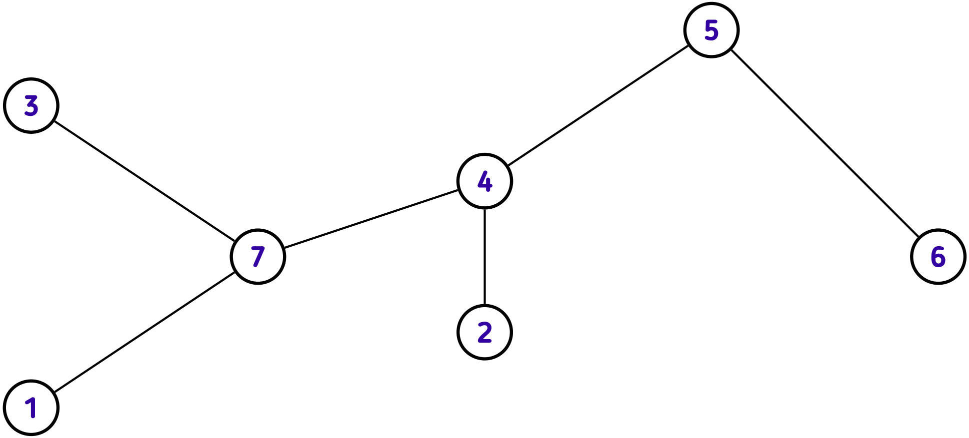
- $\star$ O caminho cujo tamanho determina o diâmetro não é, necessariamente, único

- $\star$ Computar todos os tamanho com Floyd-Warshall em $O(V^3)$ e determinar o maior dentre eles em $O(V^2)$ determina o diâmetro corretamente

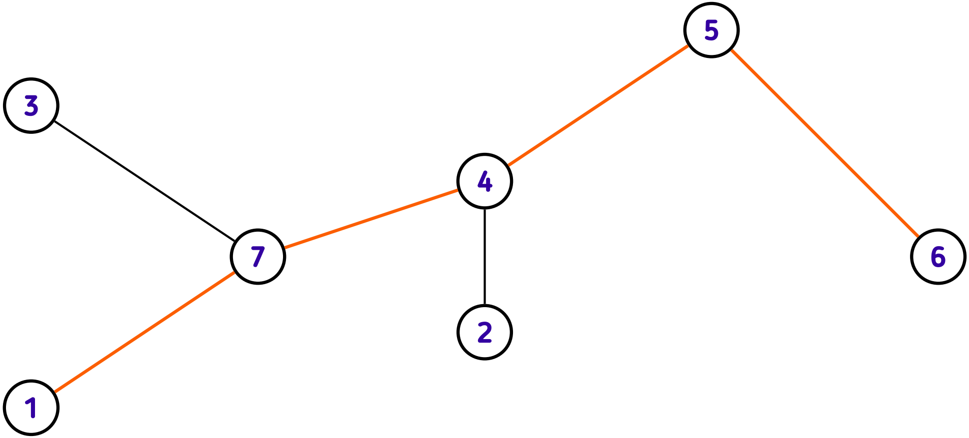
- $\star$ Porém, é possível determinar o diâmetro usando programação dinâmica ou duas BFS

Características do diâmetro

- $\star$ O caminho cujo tamanho determina o diâmetro não é, necessariamente, único
- $\star$ Computar todos os tamanho com Floyd-Warshall em $O(V^3)$ e determinar o maior dentre eles em $O(V^2)$ determina o diâmetro corretamente
- $\star$ Porém, é possível determinar o diâmetro usando programação dinâmica ou duas BFS
- $\star$ Em ambos casos, a complexidade é $O(V)$


5
3
4
7
6
2
1

Diâmetro: 4



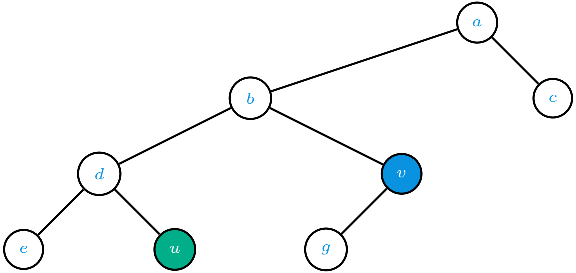
Definição de pico de um caminho

Definição de pico de um caminho

Seja T uma árvore enraizada e considere dois vértices u e v de T. O pico do caminho de u a v é o nó que ocupa o nível mais baixo em T.

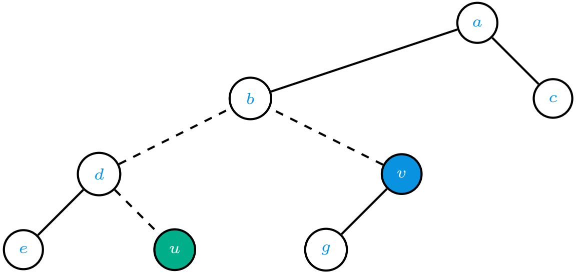
Definição de pico de um caminho

Seja T uma árvore enraizada e considere dois vértices u e v de T. O pico do caminho de u a v é o nó que ocupa o nível mais baixo em T.



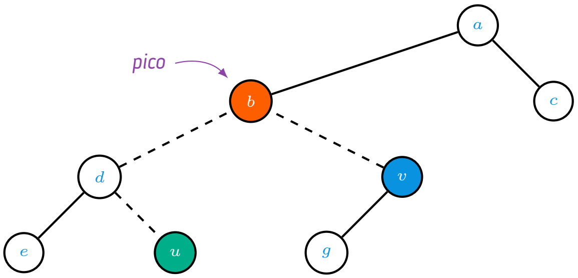
Definição de pico de um caminho

Seja T uma árvore enraizada e considere dois vértices u e v de T. O pico do caminho de u a v é o nó que ocupa o nível mais baixo em T.



Definição de pico de um caminho

Seja T uma árvore enraizada e considere dois vértices u e v de T. O pico do caminho de u a v é o nó que ocupa o nível mais baixo em T.



Diâmetro de uma árvore com programação dinâmica

Diâmetro de uma árvore com programação dinâmica

$\text{maxLength}[u]$

Diâmetro de uma árvore com programação dinâmica

Tamanho do maior caminho que tem pico igual a u

$\text{maxLength}[u]$

Diâmetro de uma árvore com programação dinâmica

$$\text{maxLength}[u] = \begin{cases} 0, & \text{se } u \text{ não tem filhos} \\ & \end{cases}$$

Diâmetro de uma árvore com programação dinâmica

$$\text{maxLength}[u] = \begin{cases} 0, & \text{se } u \text{ não tem filhos} \\ 1 + \text{toLeaf}[v], & \text{se } u \text{ tem apenas um filho } v, \\ & \end{cases}$$

Diâmetro de uma árvore com programação dinâmica

$$\text{maxLength}[u] = \begin{cases} 0, & \text{se } u \text{ não tem filhos} \\ 1 + \text{toLeaf}[v], & \text{se } u \text{ tem apenas um filho } v, \\ 2 + \text{toLeaf}[v] + \text{toLeaf}[w], & \text{caso contrário,} \end{cases}$$

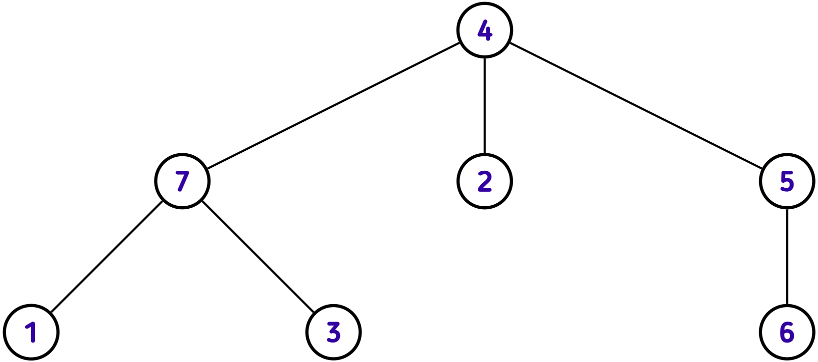
onde v **e** w **são dois filhos distintos de** u **com os dois maiores valores de** toLeaf

Diâmetro de uma árvore com programação dinâmica

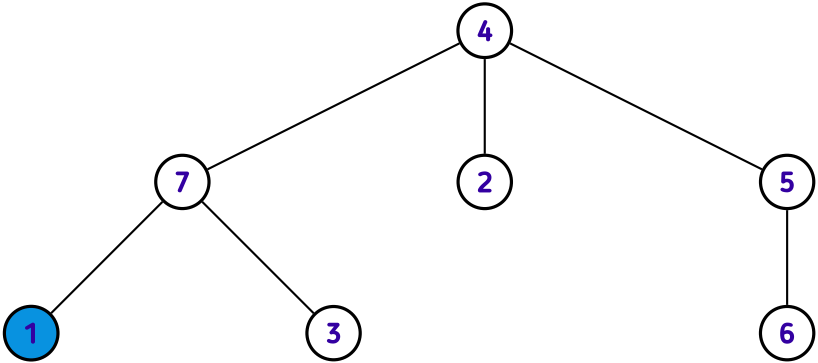
$$\mathsf{diameter}(G) = \max_{u \in V} \{ \ \mathsf{maxLength}[u] \ \}$$

$$\mathsf{maxLength}[u] = \begin{cases} 0, & \text{se } u \text{ não tem filhos} \\ 1 + \mathsf{toLeaf}[v], & \text{se } u \text{ tem apenas um filho } v, \\ 2 + \mathsf{toLeaf}[v] + \mathsf{toLeaf}[w], & \text{caso contrário,} \end{cases}$$

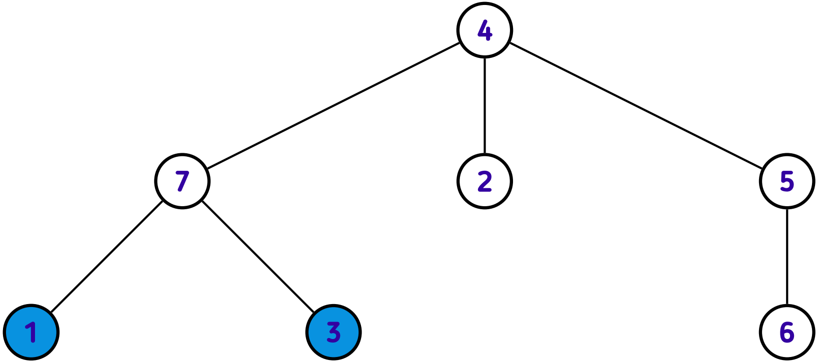
onde v **e** w **são dois filhos distintos de** u **com os dois maiores valores de** toLeaf



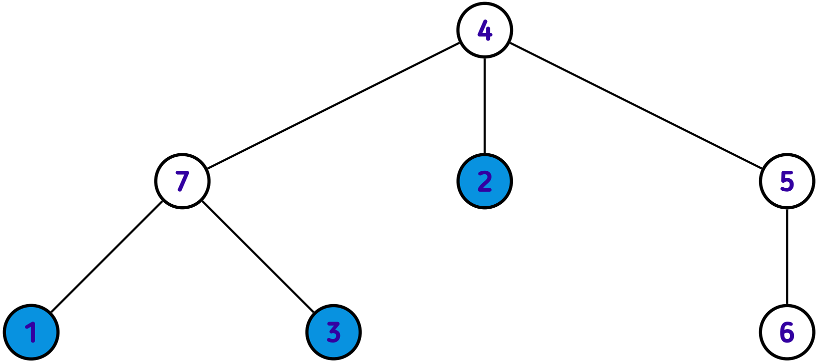
	1	2	3	4	5	6	7
toLeaf$[u]$ =	-	-	-	-	-	-	-
maxLength$[u]$ =	-	-	-	-	-	-	-



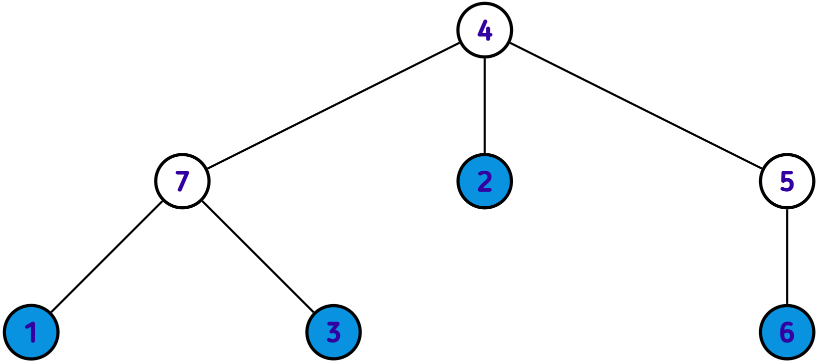
	1	2	3	4	5	6	7
toLeaf$[u]$ =	0	-	-	-	-	-	-
maxLength$[u]$ =	0	-	-	-	-	-	-



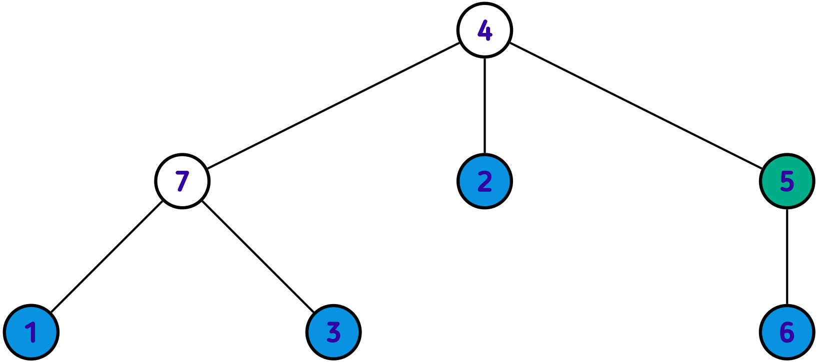
	1	2	3	4	5	6	7
toLeaf$[u]$ =	0	-	**0**	-	-	-	-
maxLength$[u]$ =	0	-	**0**	-	-	-	-



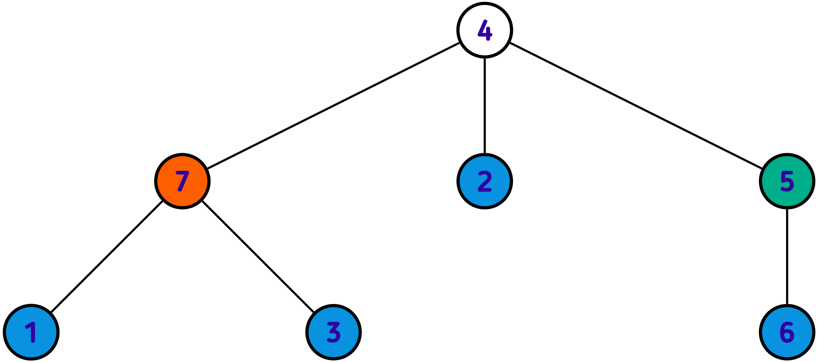
	1	2	3	4	5	6	7
toLeaf$[u]$ =	0	**0**	0	-	-	-	-
maxLength$[u]$ =	0	**0**	0	-	-	-	-



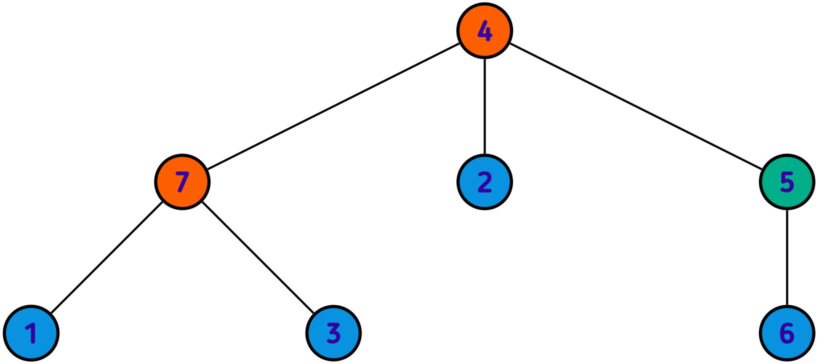
	1	2	3	4	5	6	7
$\text{toLeaf}[u] =$	0	0	0	-	-	**0**	-
$\text{maxLength}[u] =$	0	0	0	-	-	**0**	-



	1	2	3	4	5	6	7
toLeaf$[u]$ =	0	0	0	-	1	0	-
maxLength$[u]$ =	0	0	0	-	1	0	-



	1	2	3	4	5	6	7
$\text{toLeaf}[u] =$	0	0	0	-	1	0	**1**
$\text{maxLength}[u] =$	0	0	0	-	1	0	**2**



	1	2	3	4	5	6	7
$\text{toLeaf}[u] =$	0	0	0	**2**	1	0	1
$\text{maxLength}[u] =$	0	0	0	**4**	1	0	2

```cpp
int diameter(int root, int N)
{
    dfs(root, 0);

    int d = 0;

    for (int u = 1; u <= N; ++u)
        d = max(d, max_length[u]);

    return d;
}
```

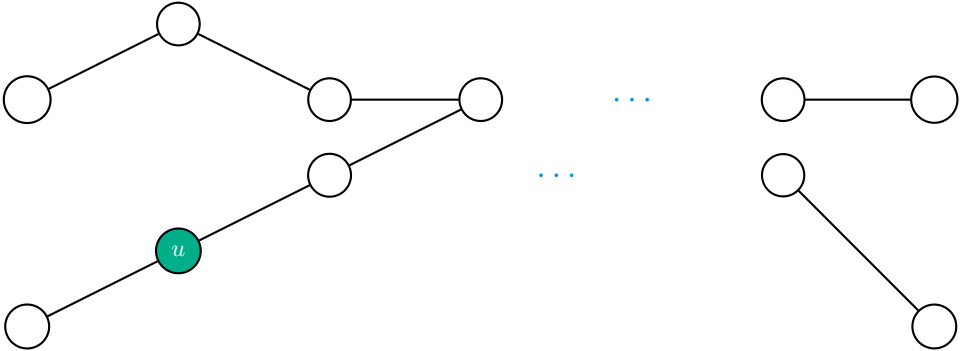
```
void dfs(int u, int p)
{
    vector<int> ds;

    for (auto v : adj[u])
    {
        if (v == p)
            continue;

        dfs(v, u);
        ds.push_back(to_leaf[v]);
    }

    sort(ds.begin(), ds.end());

    to_leaf[u] = ds.empty() ? 0 : ds.back() + 1;
```

```
    auto N = ds.size();

    switch (N) {
    case 0:
        max_length[u] = 0;
        break;

    case 1:
        max_length[u] = ds.back() + 1;
        break;

    default:
        max_length[u] = ds[N - 1] + ds[N - 2] + 2;
    }
}
```

Uso de BFS para o cálculo do diâmetro

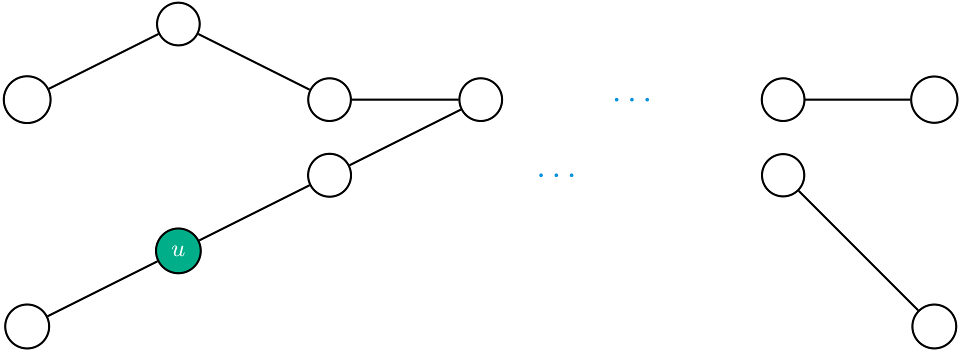
Uso de BFS para o cálculo do diâmetro

A BFS permite computar as distância, em arestas, de qualquer vértice a u



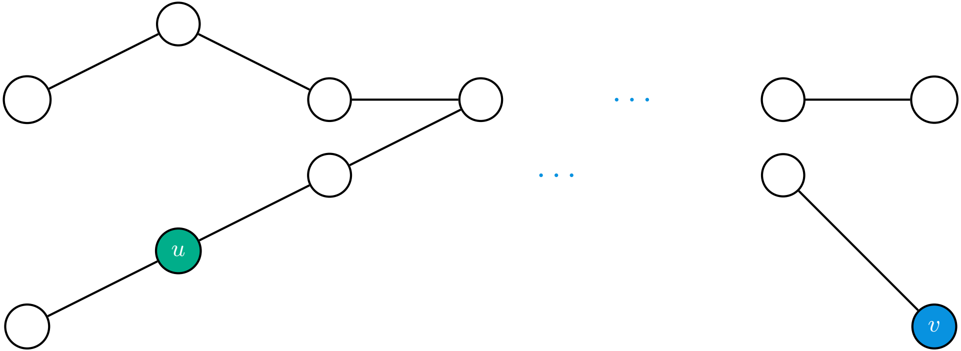
Uso de BFS para o cálculo do diâmetro

Seja v o vértice mais distante de u e D o diâmetro da árvore T



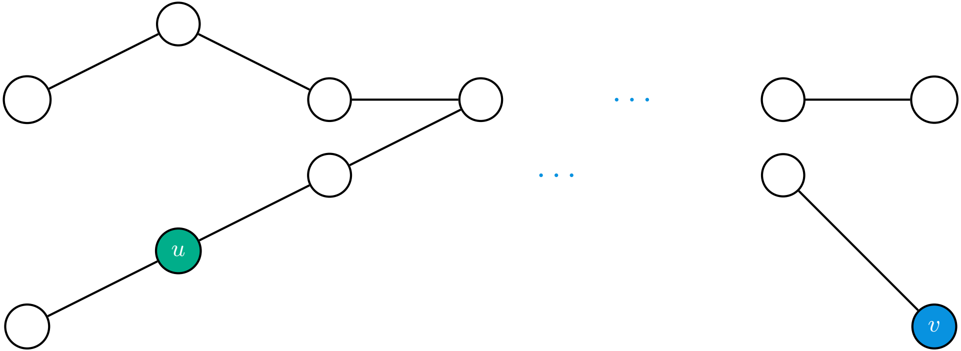
Uso de BFS para o cálculo do diâmetro

Seja v o vértice mais distante de u e D o diâmetro da árvore T



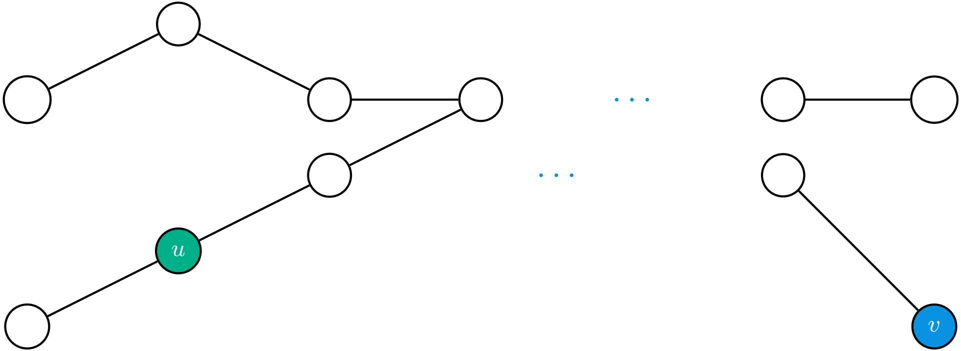
Uso de BFS para o cálculo do diâmetro

Fato #1: Ao menos um nó do caminho de u a v faz parte de um caminho cujo comprimento é D



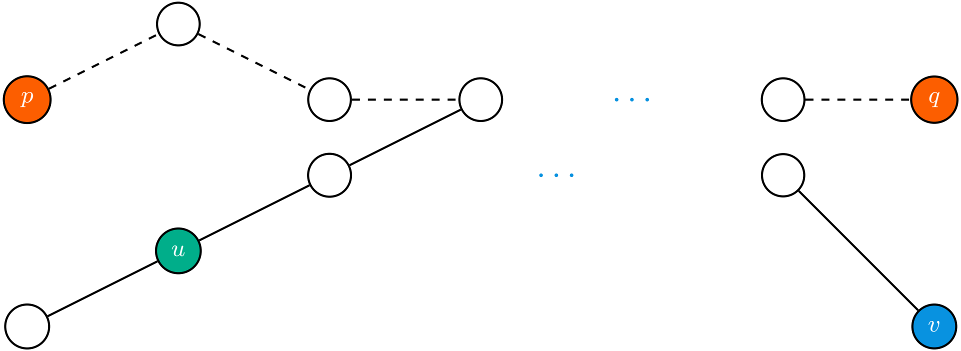
Uso de BFS para o cálculo do diâmetro

Prova: Suponha que nenhum vértice do caminho de u a v faça parte de um caminho cujo comprimento é D



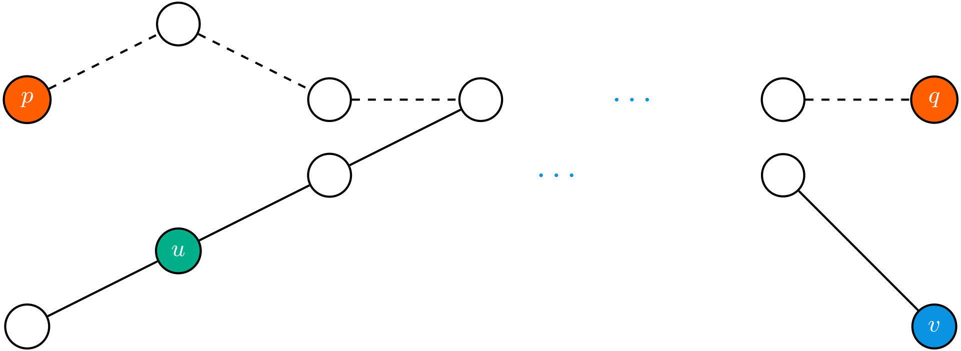
Uso de BFS para o cálculo do diâmetro

Assuma que $\text{dist}(p, q) = D$



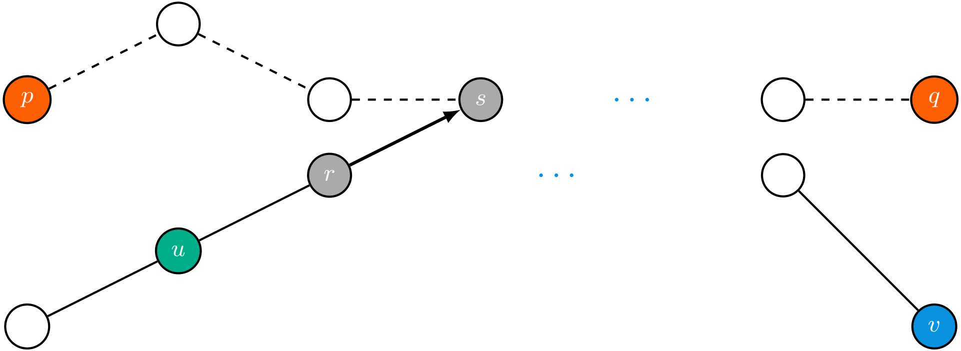
Uso de BFS para o cálculo do diâmetro

Sendo T conectada, existe ao menos um caminho de um vértice r no caminho de u a v para um vértice s no caminho de p a q



Uso de BFS para o cálculo do diâmetro

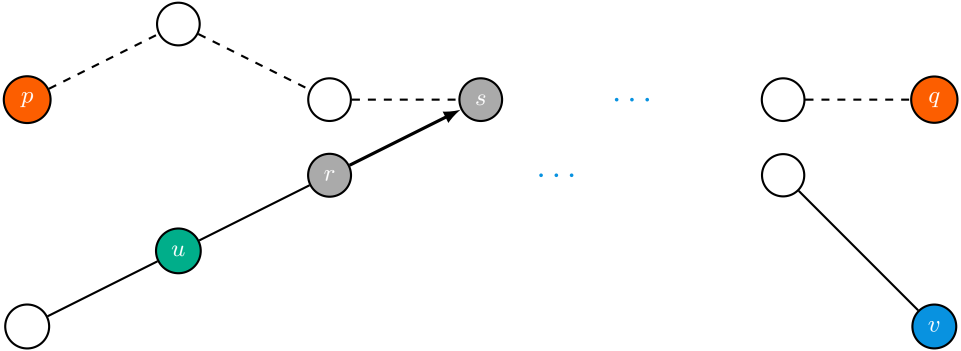
Sendo T conectada, existe ao menos um caminho de um vértice r no caminho de u a v para um vértice s no caminho de p a q



Uso de BFS para o cálculo do diâmetro

Como v é vértice mais distante de u, vale que

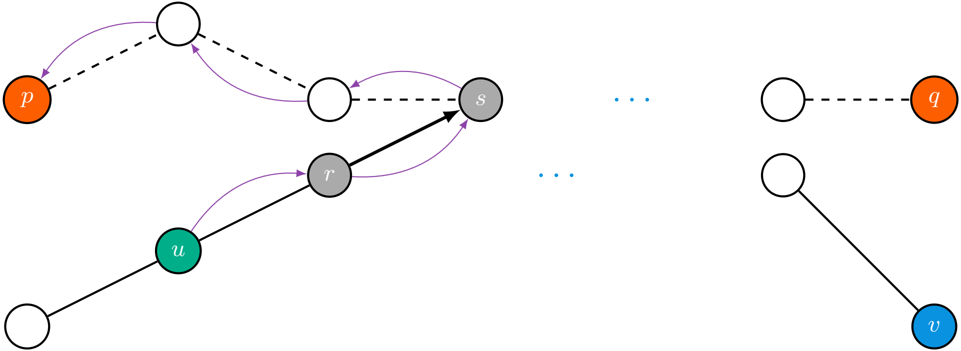
$$\text{dist}(u, r) + \text{dist}(r, s) + \text{dist}(s, p) \leq \text{dist}(u, v)$$



Uso de BFS para o cálculo do diâmetro

Como v é vértice mais distante de u, vale que

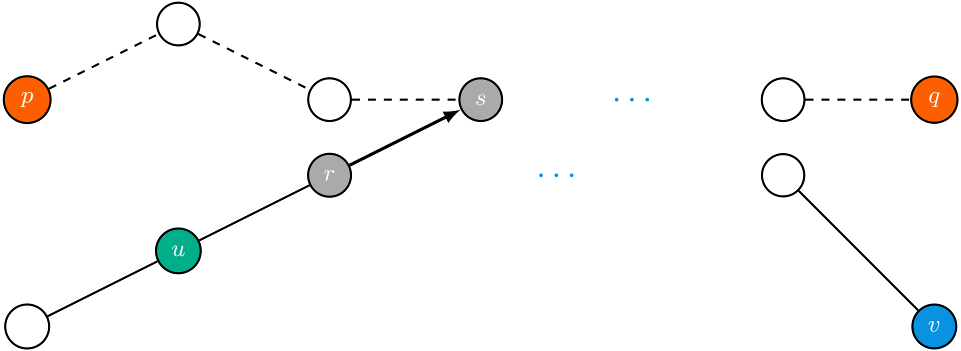
$$\text{dist}(u, r) + \text{dist}(r, s) + \text{dist}(s, p) \leq \text{dist}(u, v)$$



Uso de BFS para o cálculo do diâmetro

e também que

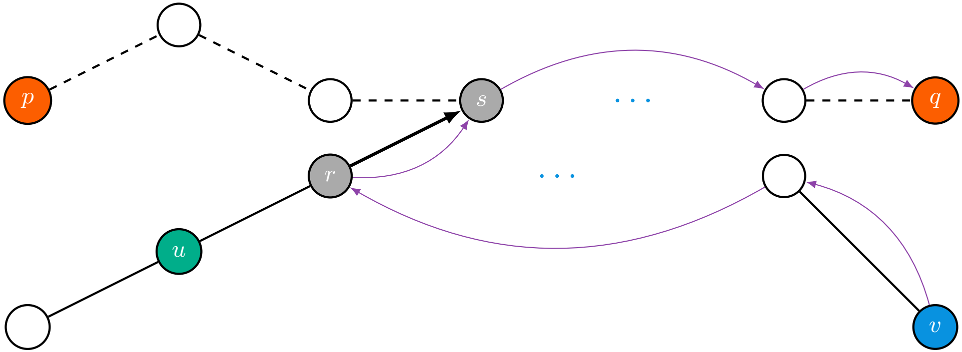
$$\text{dist}(v, r) + \text{dist}(r, s) + \text{dist}(s, q) \leq \text{dist}(u, v)$$



Uso de BFS para o cálculo do diâmetro

e também que

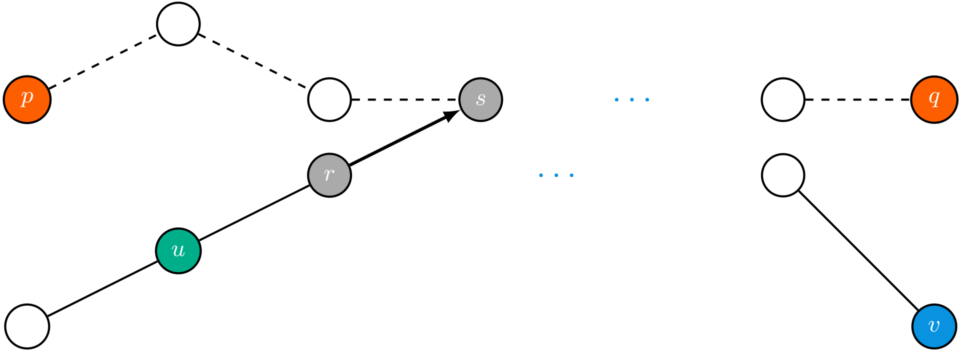
$$\text{dist}(v, r) + \text{dist}(r, s) + \text{dist}(s, q) \leq \text{dist}(u, v)$$



Uso de BFS para o cálculo do diâmetro

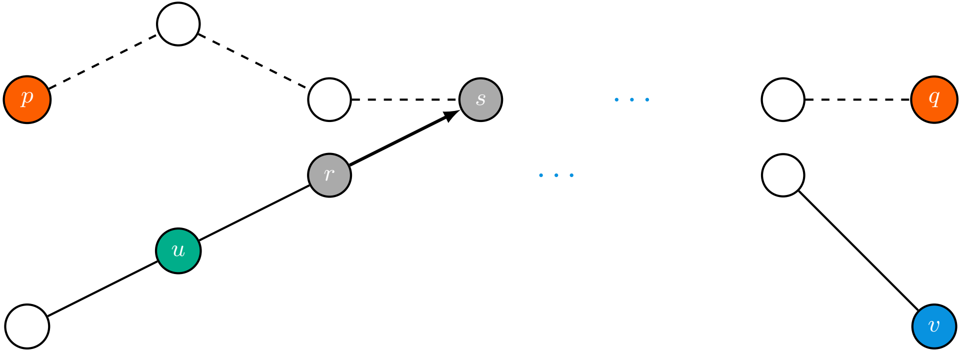
Somando ambas desigualdades, temos que

$$D + 2 \times \text{dist}(r, s) \leq \text{dist}(u, v)$$



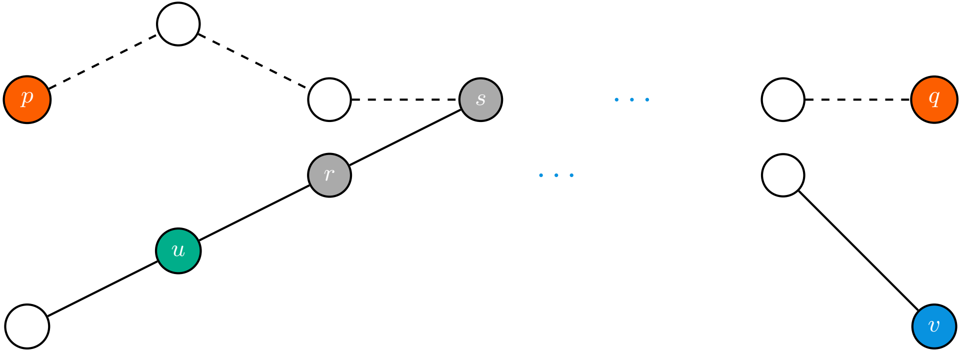
Uso de BFS para o cálculo do diâmetro

Como $\text{dist}(r, s) > 0$**, teríamos** $\text{dist}(u, v) > D$**, uma contradição!**



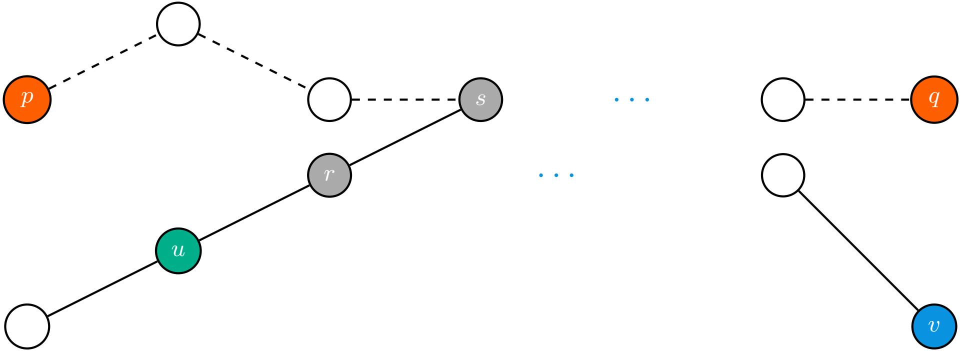
Uso de BFS para o cálculo do diâmetro

Fato #2: v **é um dos extremos de um caminho cujo tamanho é** D



Uso de BFS para o cálculo do diâmetro

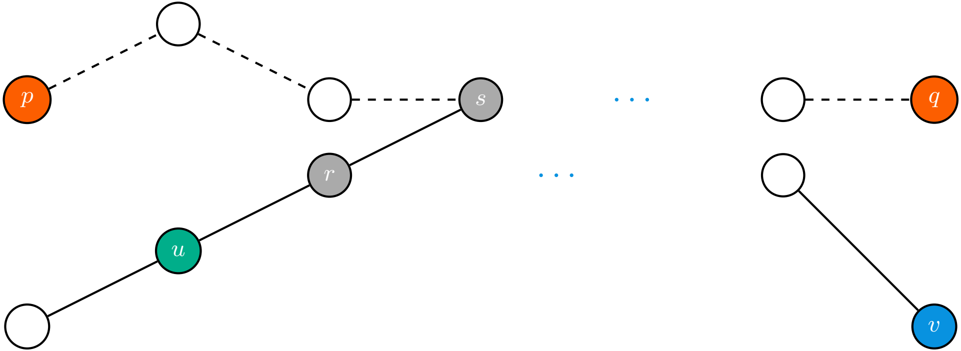
Prova: Seja s um nó do caminho de u a v pelo qual passa o caminho de p a q cujo tamanho é D



Uso de BFS para o cálculo do diâmetro

Se v não é extremo de um caminho cujo tamanho é D, então

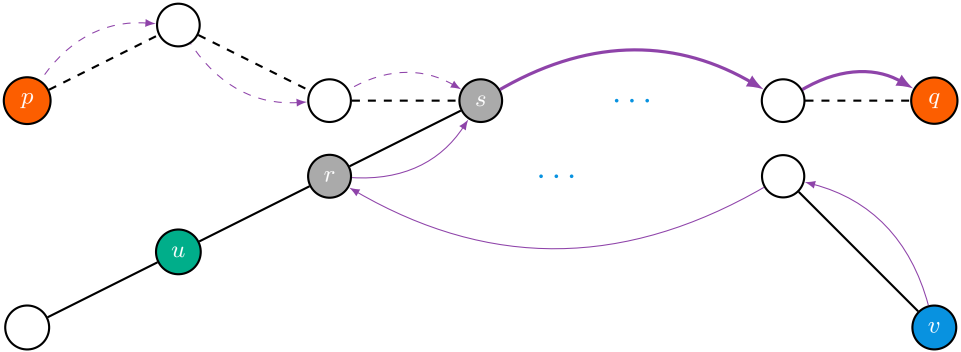
$$\text{dist}(v, n) < D,\ \forall n \in V$$



Uso de BFS para o cálculo do diâmetro

Em particular,

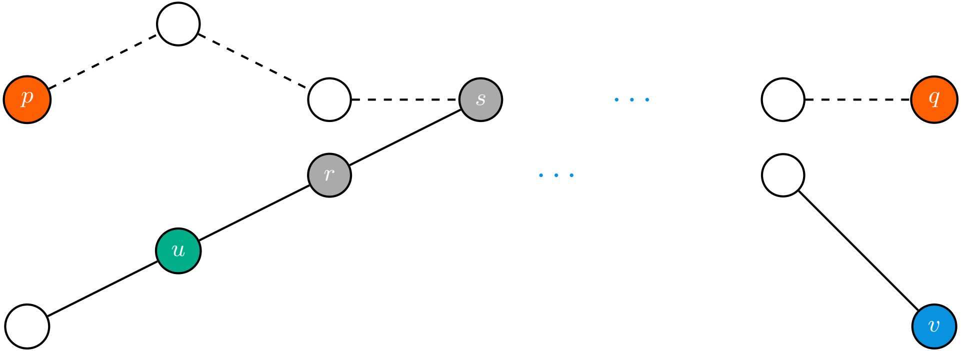
$$\text{dist}(v, s) + \text{dist}(s, q) = \text{dist}(v, q) < \text{dist}(p, q) = \text{dist}(p, s) + \text{dist}(s, q)$$



Uso de BFS para o cálculo do diâmetro

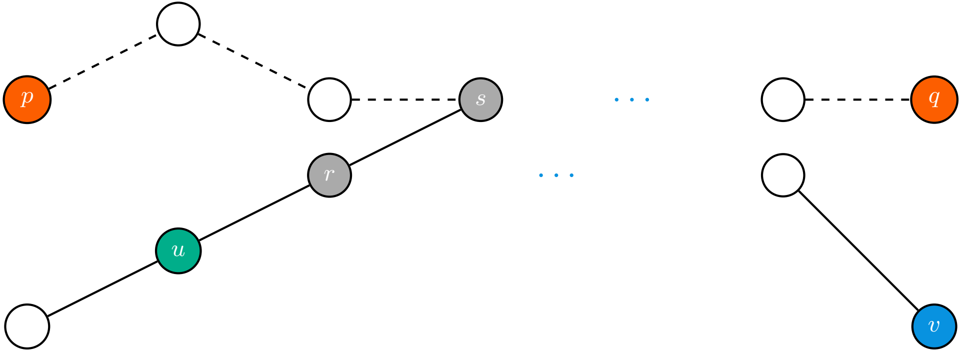
Deste modo, $\text{dist}(s, p) > \text{dist}(s, v)$**, o que leva a**

$$\text{dist}(u, v) = \text{dist}(u, s) + \text{dist}(s, v) < \text{dist}(u, s) + \text{dist}(s, p) = \text{dist}(u, p)$$



Uso de BFS para o cálculo do diâmetro

Logo p estaria mais distante de u do que v, que é o vértice mais distante de u, outra contradição!



Pseudocódigo

Pseudocódigo

Entrada: uma árvore $T(V, E)$

Saída: o diâmetro D da árvore

Pseudocódigo

Entrada: uma árvore $T(V, E)$

Saída: o diâmetro D da árvore

1. Escolha um vértice $u \in V$ qualquer

Pseudocódigo

Entrada: uma árvore $T(V, E)$

Saída: o diâmetro D da árvore

1. Escolha um vértice $u \in V$ qualquer

2. Seja v o vértice mais distante de u, identificado por meio de uma BFS

Pseudocódigo

Entrada: uma árvore $T(V, E)$

Saída: o diâmetro D da árvore

1. Escolha um vértice $u \in V$ qualquer
2. Seja v o vértice mais distante de u, identificado por meio de uma BFS
3. Seja w o vértice mais distante de v, identificado por meio de uma BFS

Pseudocódigo

Entrada: uma árvore $T(V, E)$

Saída: o diâmetro D da árvore

1. Escolha um vértice $u \in V$ qualquer

2. Seja v o vértice mais distante de u, identificado por meio de uma BFS

3. Seja w o vértice mais distante de v, identificado por meio de uma BFS

4. Retorne $D = \text{dist}(v, w)$

```
pair<int, int> bfs(int s, int N)
{
    vector<int> dist(N + 1, oo); dist[s] = 0;
    queue<int> q; q.push(s);
    int last = s;

    while (not q.empty()) {
        auto u = q.front(); q.pop();
        last = u;

        for (auto v : adj[u]) {
            if (dist[v] == oo) {
                dist[v] = dist[u] + 1;
                q.push(v);
            }
        }
   }

    return { last, dist[last] };
}
```

```cpp
int diameter(int N)
{
    auto [v, _] = bfs(1, N);
    auto [w, D] = bfs(v, N);

    return D;
}
```

Problemas sugeridos

1. AIZU Online Judge GRL 5A – Diameter of a Tree
2. Codechef DTREE – Diameter of Tree
3. DM::OJ – Tree Tasks
4. OJ 10308 – Roads in the North

Referências

1. **DROZDEK, Adam.** *Algoritmos e Estruturas de Dados em C++,* **2002.**
2. **HALIM, Felix; HALIM, Steve.** *Competitive Programming 3,* **2010.**
3. **LAAKSONEN, Antti.** *Competitive Programmer's Handbook,* **2018.**
4. **SKIENA, Steven; REVILLA, Miguel.** *Programming Challenges,* **2003.**
5. **Wikipédia.** *Tree (graph theory),* **acesso em 06/08/2021.**