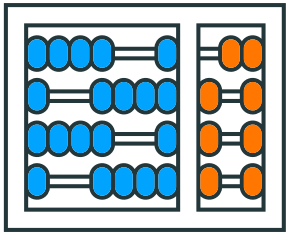


Instituto de Computação

UNIVERSIDADE ESTADUAL DE CAMPINAS

UNICAMP

Árvores Filogenéticas

MO640 - Biologia Computacional / MC668 - Bioinformática

Zanoni Dias

2021

Instituto de Computação

Roteiro

Árvores Filogenéticas

Árvores Aditivas

Árvores Aditivas Compactas

Árvores Ultramétricas

UPGMA

Neighbor-Joining

Sanduíche Ultramétrico

Características com Estados Discretos

Árvores Filogenéticas

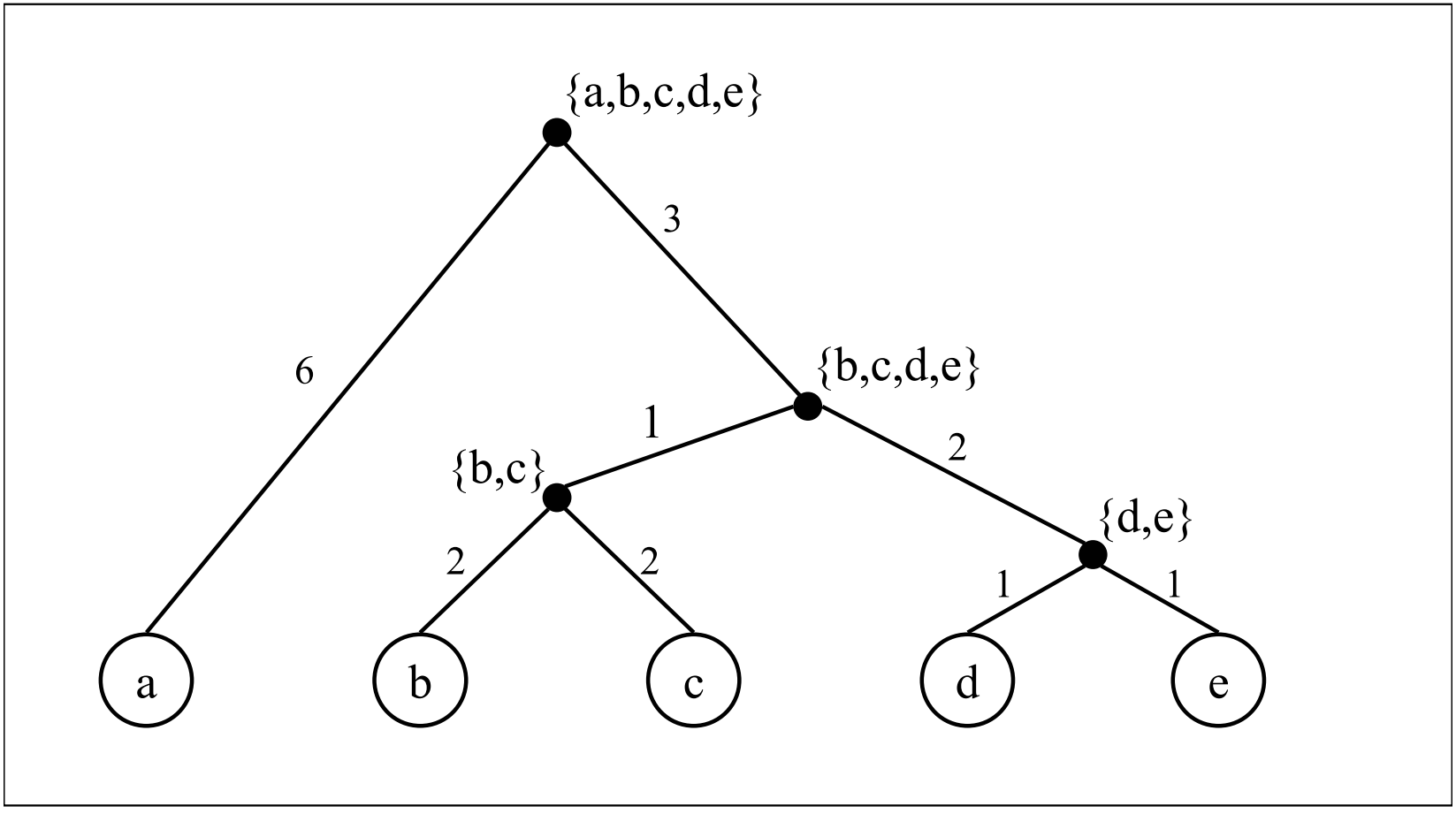
Árvores Filogenéticas

- Árvores filogenéticas representam a abordagem mais comumente usada para reconstruir a relação entre objetos biológicos.
- Cada folha da árvore denota um dos objetos biológicos, enquanto os nós internos representam ancestrais hipotéticos.
- A distância entre os objetos na árvore pode servir com uma medida do grau de relação entre os objetos.

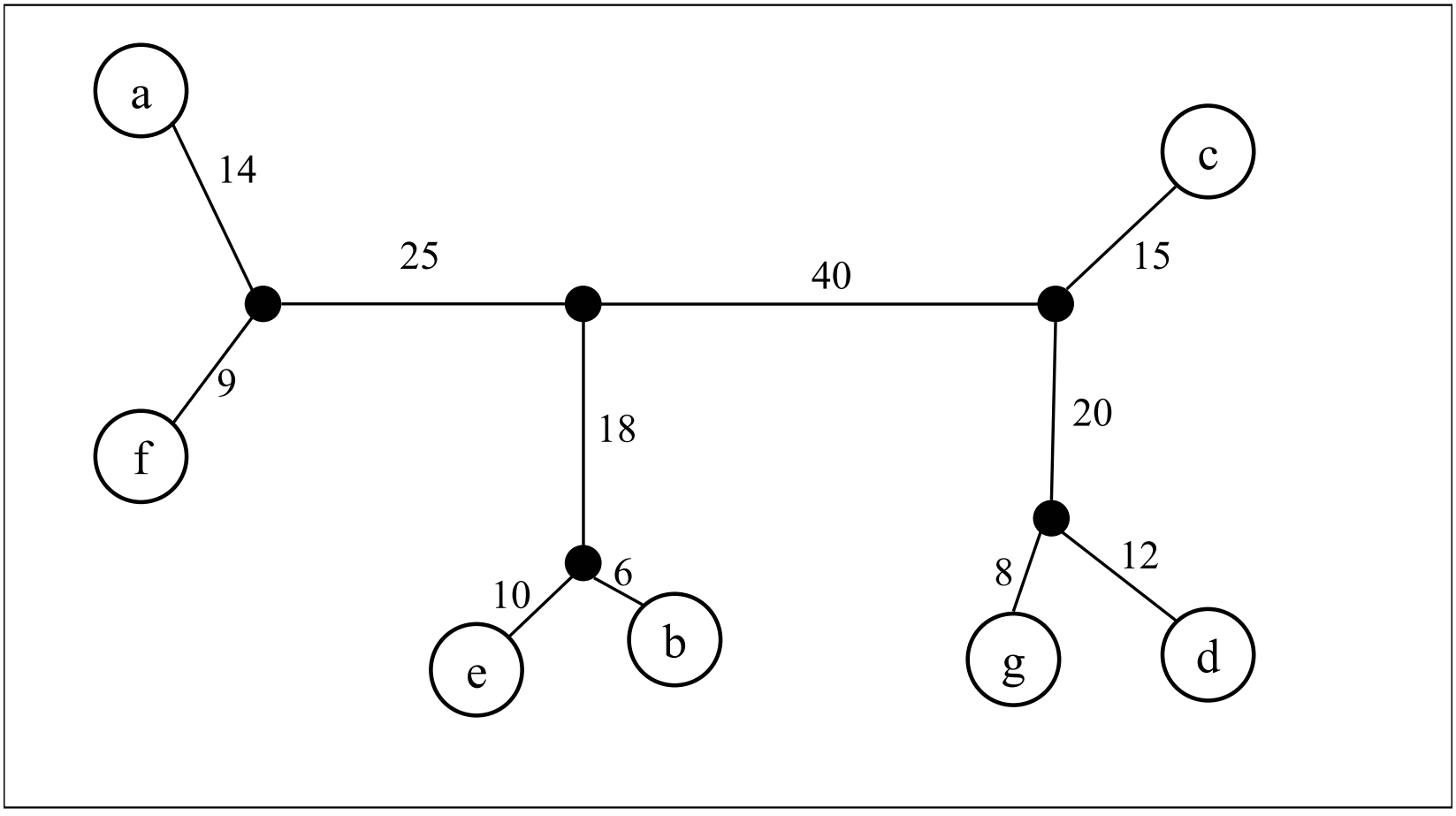
Árvores Filogenéticas

- Em relação às árvores filogenéticas, há dois interesses principais:
  - Obter a topologia da árvore, ou seja, a forma como os nós internos se conectam uns com os outros e com as folhas.
  - Obter as distâncias entre todos os nós da árvore.
- Em relação a raiz de uma árvore filogenética, temos dois casos:
  - Nas árvores com raiz (ou enraizadas), a raiz representa o ancestral comum a todos os nós da árvore.
  - Nem sempre temos informações suficientes para determinar o ancestral comum a todos os nós. Neste caso, constroi-se uma árvore sem raiz.

Árvore Filogenética com Raiz



Árvore Filogenética sem Raiz



Dados para Construção de Árvores Filogenéticas

- Os tipos de informações utilizadas para reconstrução filogenética são, normalmente, divididos em três categorias:
  - Informação comparativa numérica, chamada Matriz de Distância entre os pares de objetos.
  - Características discretas, tais como cor da pele, número de dedos, presença de asas, presença de um sítio de restrição, presença de um SNP, etc. Cada característica possui um número finito de estados (valores distintos que a característica pode assumir). Neste caso a informação é organizada numa matriz chamada Matriz de Estados das Características.
  - Características contínuas, tais como altura na fase adulta, peso no nascimento, tamanho do genoma, etc. Cada característica pode possuir um número infinito de estados. Neste caso a informação também pode ser organizada numa Matriz de Estados das Características.

Árvores Aditivas

Árvores Aditivas

Definição

Seja $\mathcal{A}$ um conjunto de objetos e $\delta : \mathcal{A} \times \mathcal{A} \to \mathbb{R}^+$ uma função. Então δ é uma métrica para $\mathcal{A}$ se satisfaz as seguintes propriedades:

- *Para todo par $a, b \in \mathcal{A}$, $\delta(a, b) = 0$ se e somente se $a = b$.*
- *Para todo par $a, b \in \mathcal{A}$, $\delta(a, b) = \delta(b, a)$ (simetria).*
- *Para toda trinca $a, b, c \in \mathcal{A}$, $\delta(a, b) \leq \delta(a, c) + \delta(c, b)$ (desigualdade triangular).*

Árvores Aditivas

Definição

Seja $\mathcal{A}$ um conjunto de objetos e $\delta : \mathcal{A} \times \mathcal{A} \to \mathbb{R}^+$ uma medida de distância métrica para $\mathcal{A}$. Seja $T = (V, E, d)$ uma árvore ponderada tal que $\mathcal{A} \subseteq V$. Seja $dist(x, y)$ a distância entre dois vértice quaisquer x e y em T, calculada como a soma dos pesos das arestas do caminho entre x e y. A árvore T é chamada aditiva para $\mathcal{A}$ e δ se e somente se, para todo $a, b \in \mathcal{A}$, $dist(a, b) = \delta(a, b)$.

Como Verificar se uma Matriz de Distância é Aditiva

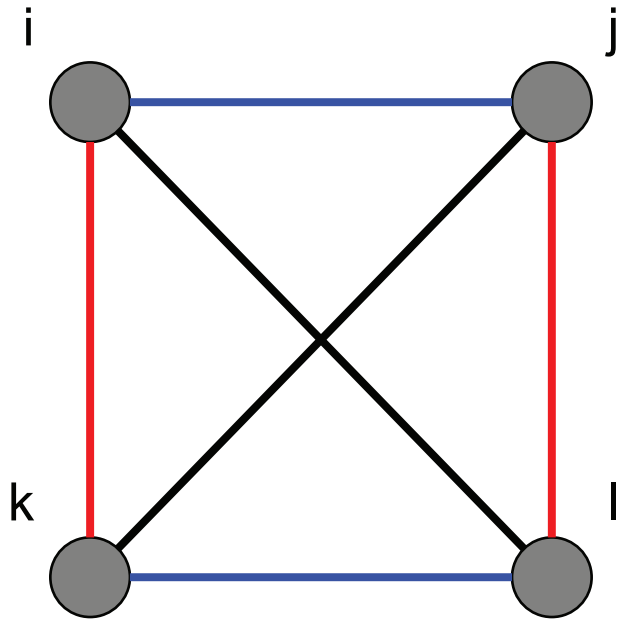
Teorema

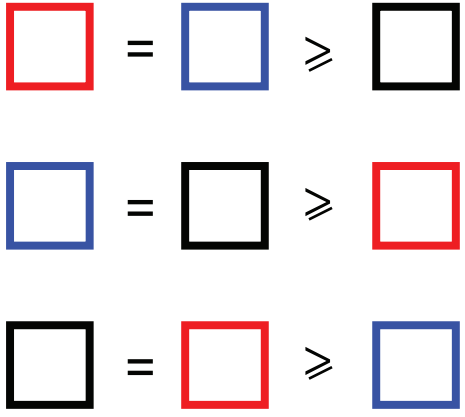
Seja $\mathcal{A}$ um conjunto de objetos e $\delta : \mathcal{A} \times \mathcal{A} \to \mathbb{R}^+$ uma medida de distância métrica para $\mathcal{A}$. A métrica δ é dita aditiva para $\mathcal{A}$, ou seja admite uma árvore aditiva, se e somente se para todo conjunto de 4 elementos $i, j, k, l \in \mathcal{A}$, temos que:

- *ou* $\delta(i,j) + \delta(k,l) = \delta(i,k) + \delta(j,l) \geq \delta(i,l) + \delta(j,k)$,
- *ou* $\delta(i,l) + \delta(k,j) = \delta(i,k) + \delta(j,l) \geq \delta(i,j) + \delta(k,l)$,
- *ou* $\delta(i,j) + \delta(k,l) = \delta(i,l) + \delta(k,j) \geq \delta(i,k) + \delta(j,l)$.

- Teorema provado, independentemente, por Peter Buneman (1971) e Annete Dobson (1974).

Árvore Aditiva





Como Construir uma Árvore Aditiva

Lema

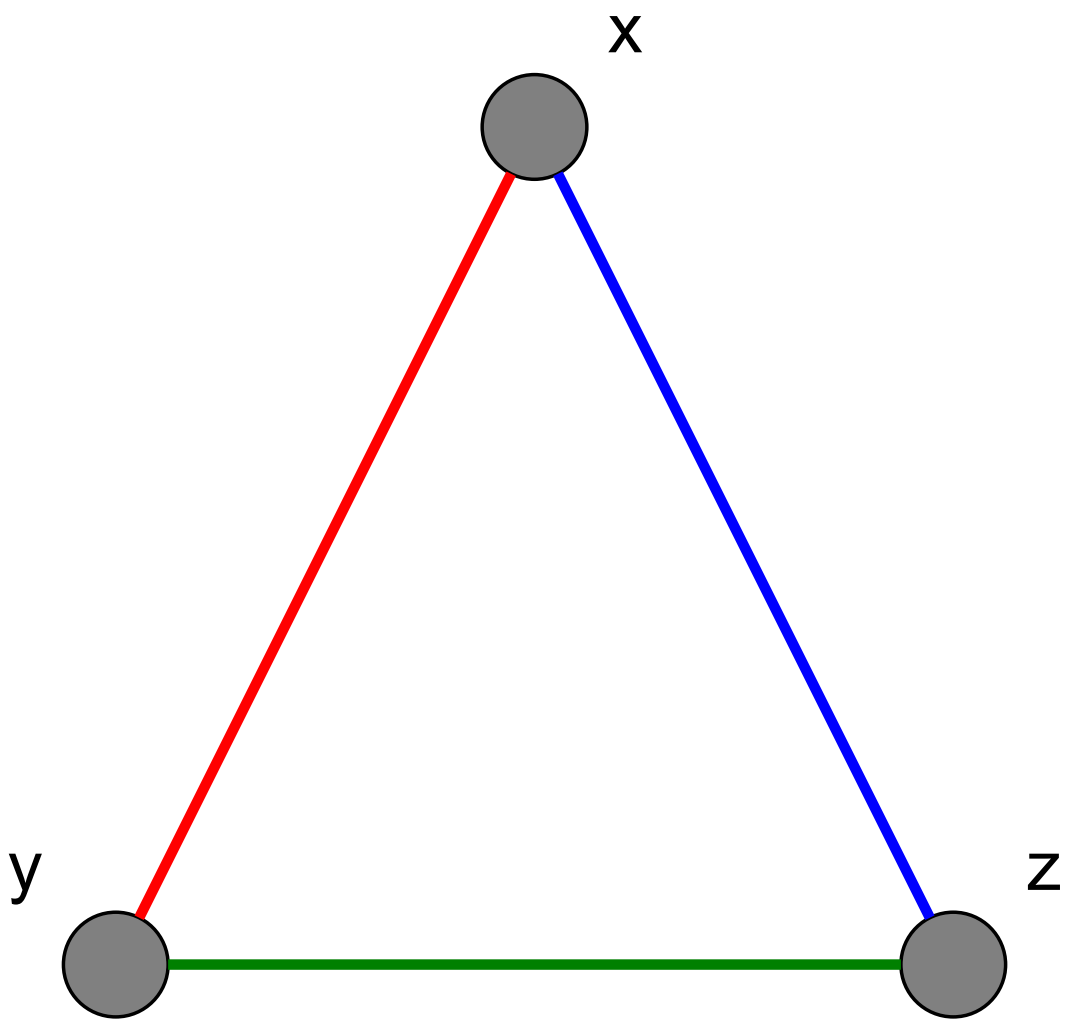
Seja $\mathcal{A} = \{x, y, z\}$ um conjunto de objetos e $\delta : \mathcal{A} \times \mathcal{A} \to \mathbb{R}^+$ uma distância métrica aditiva para $\mathcal{A}$. Logo podemos construir uma árvore aditiva $T = (V, E, d)$, com $V = \{x, y, z, c\}$, $E = \{\{x, c\}, \{y, c\}, \{z, c\}\}$, com pesos para as arestas dados pelas seguintes fórmulas:

$$d(x, c) = \frac{\delta(x, y) + \delta(x, z) - \delta(y, z)}{2}$$

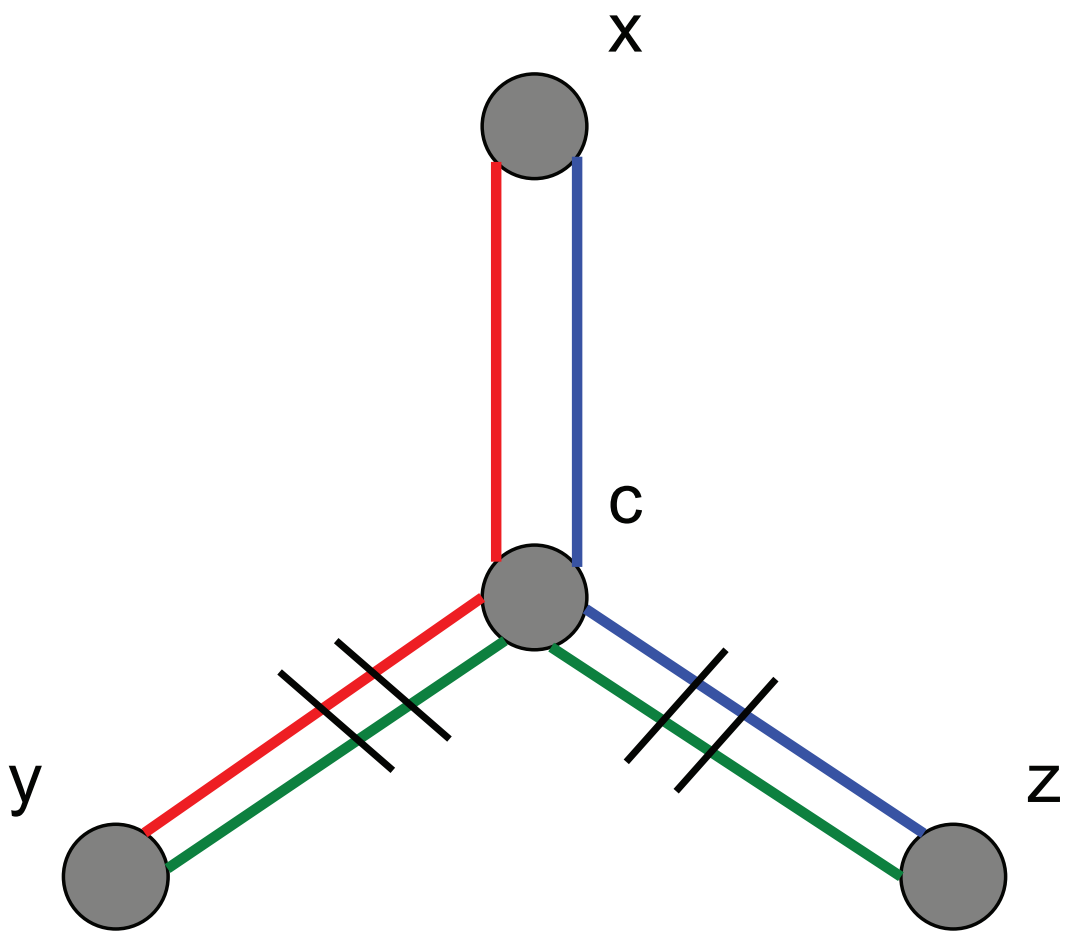
$$d(y, c) = \frac{\delta(x, y) + \delta(y, z) - \delta(x, z)}{2}$$

$$d(z, c) = \frac{\delta(x, z) + \delta(y, z) - \delta(x, y)}{2}$$

Árvore Aditiva



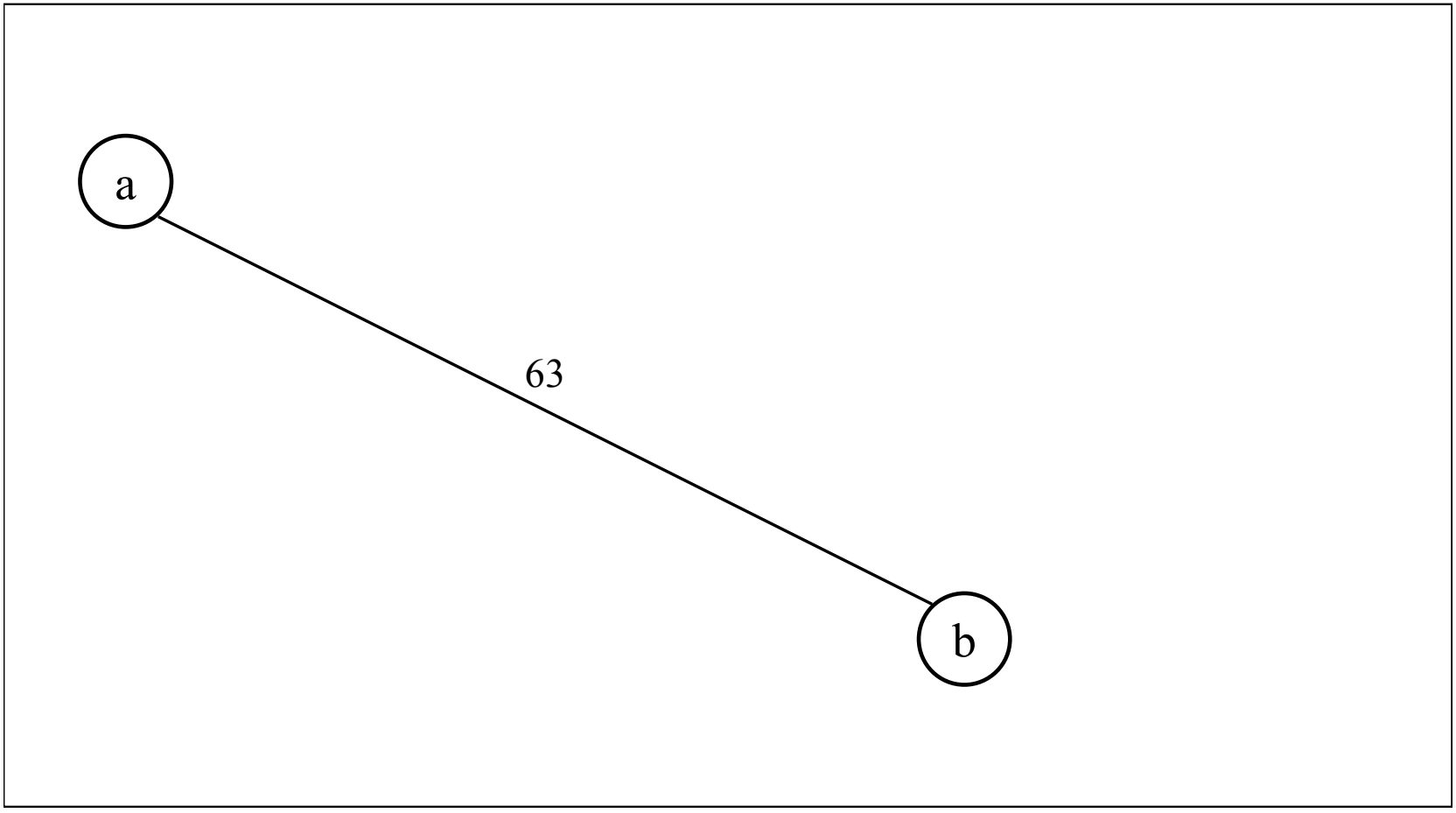
Árvore Aditiva



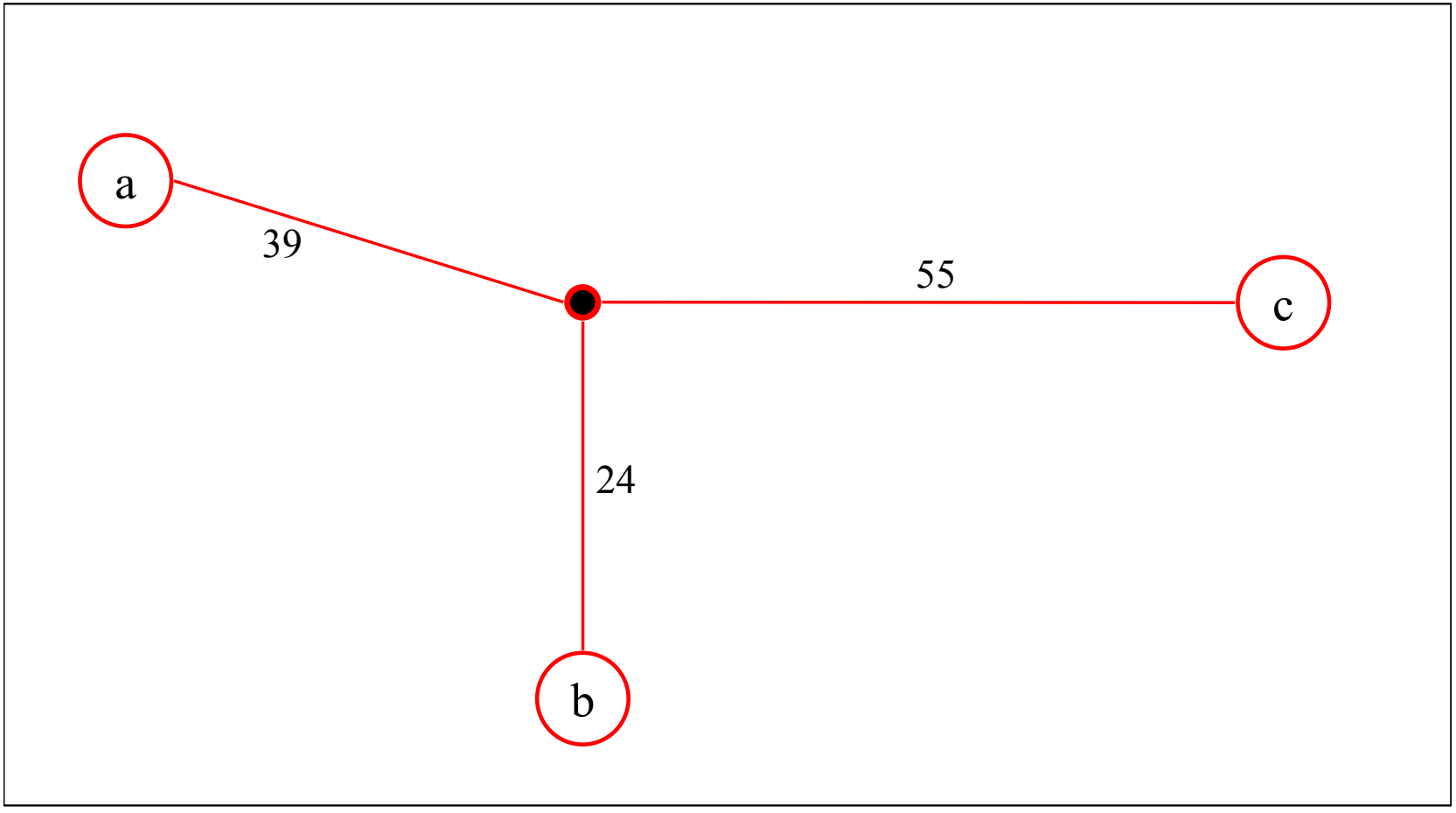
Árvore Aditiva

	a	b	c	d	e	f	g
a	0	63	94	111	67	23	107
b	63	0	79	96	16	58	92
c	94	79	0	47	83	89	43
d	111	96	47	0	100	106	20
e	67	16	83	100	0	62	96
f	23	58	89	106	62	0	102
g	107	92	43	20	96	102	0

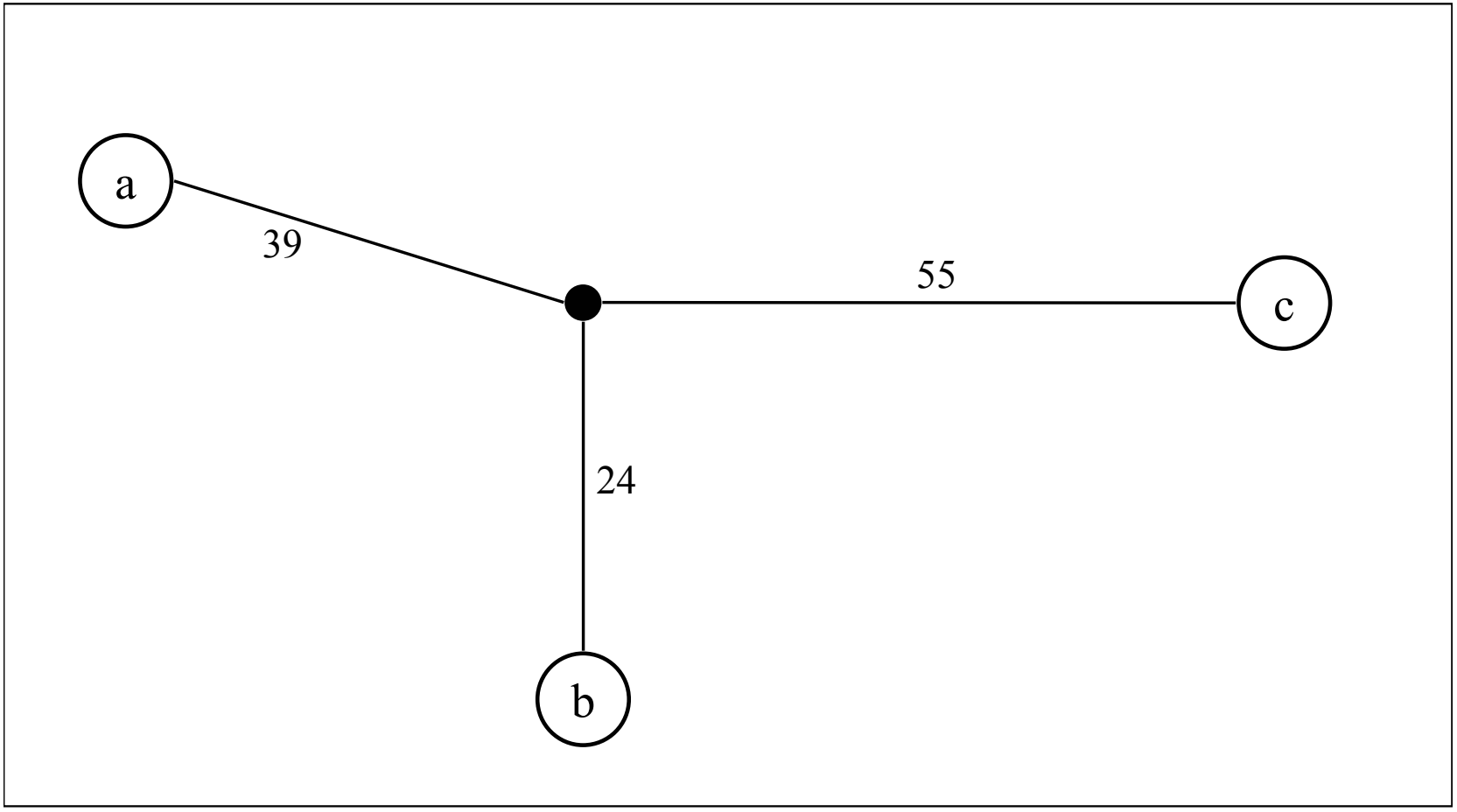
Árvore Aditiva



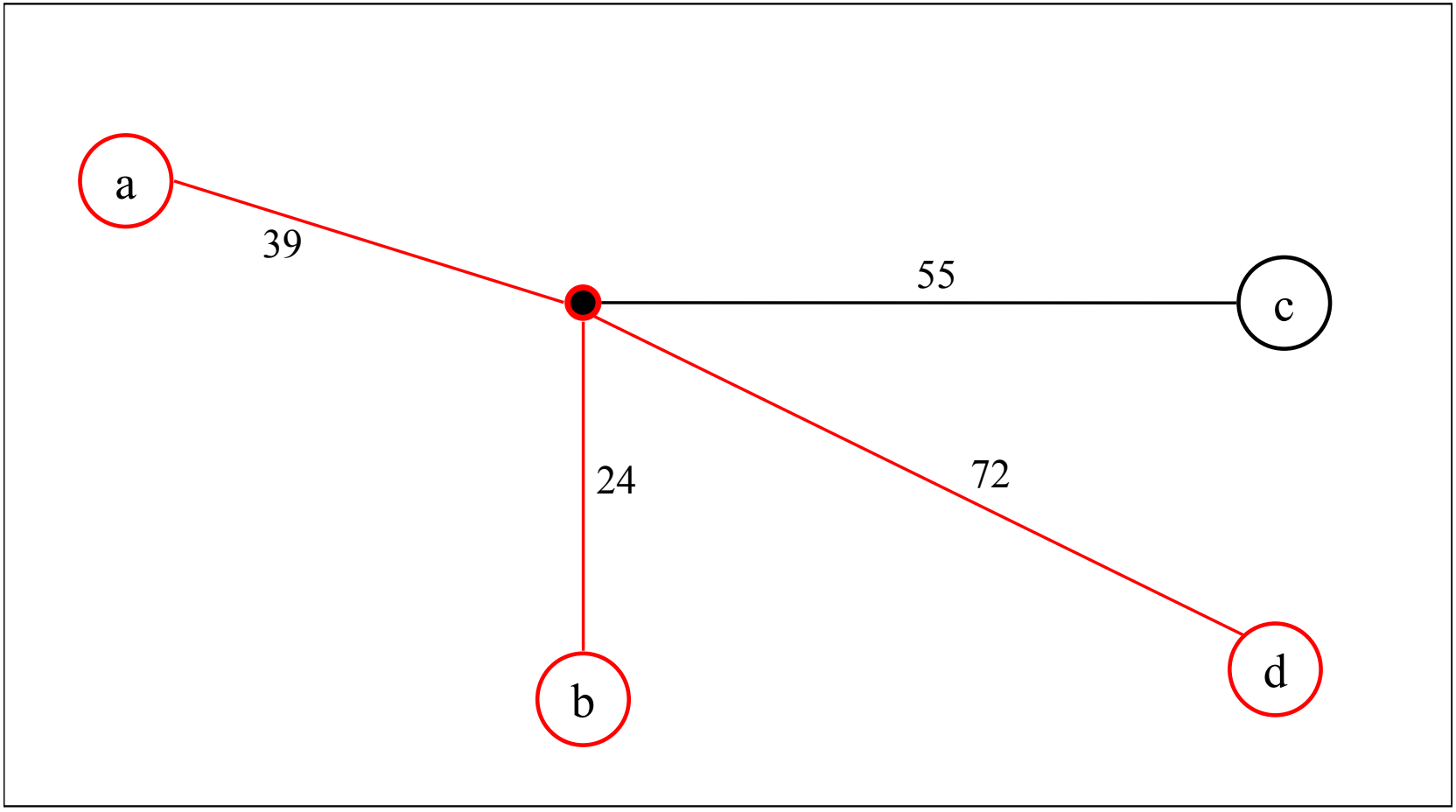
Árvore Aditiva



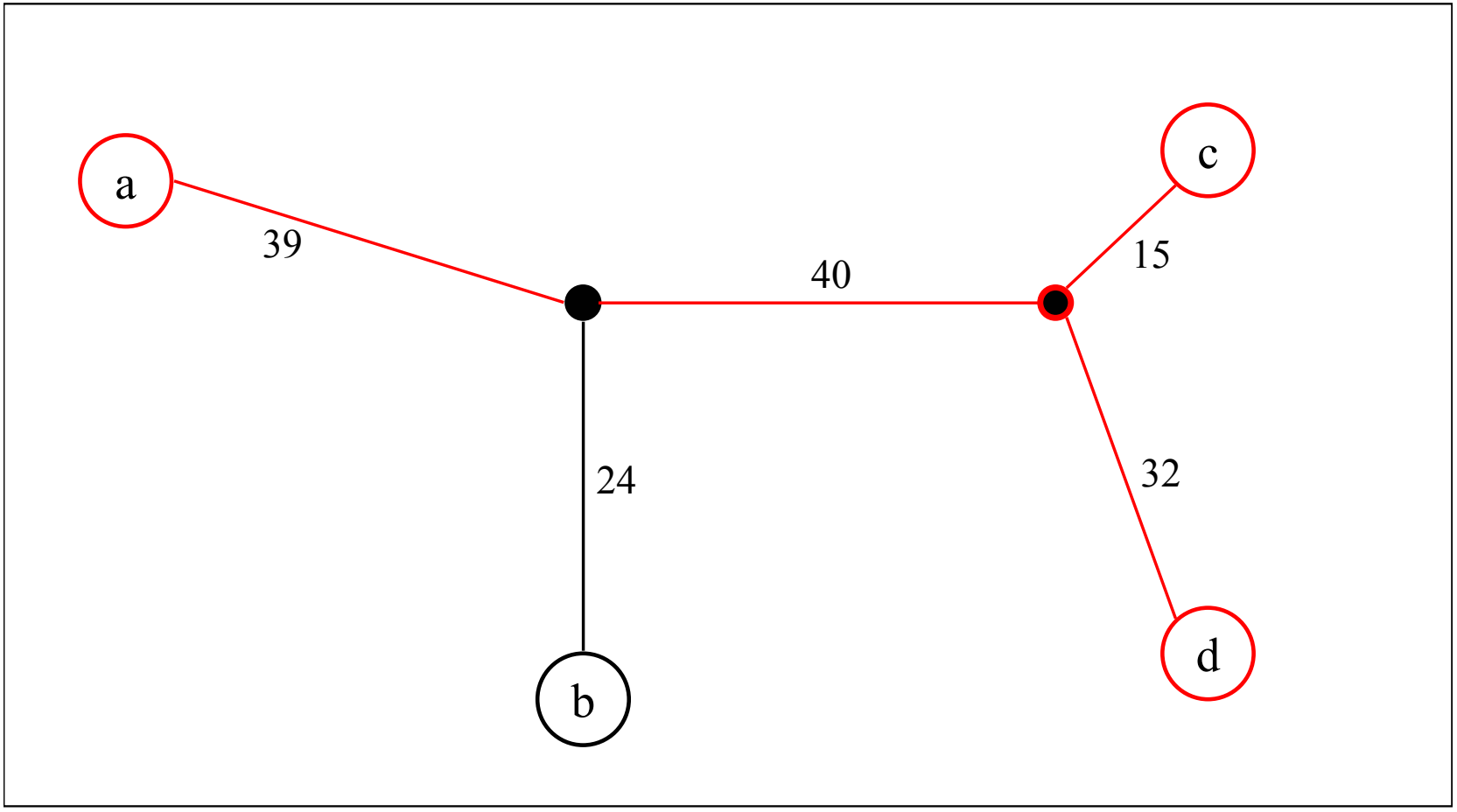
Árvore Aditiva



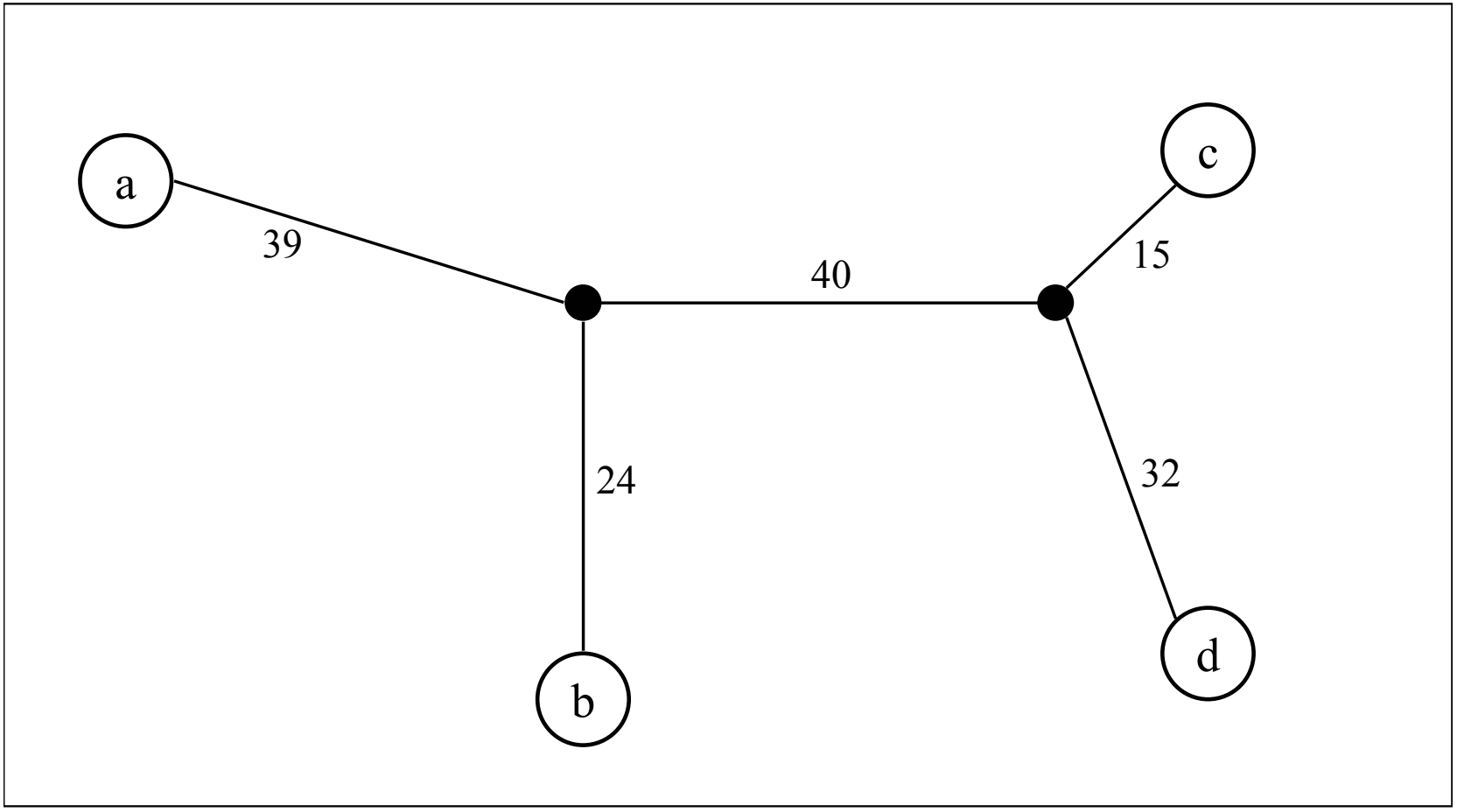
Árvore Aditiva



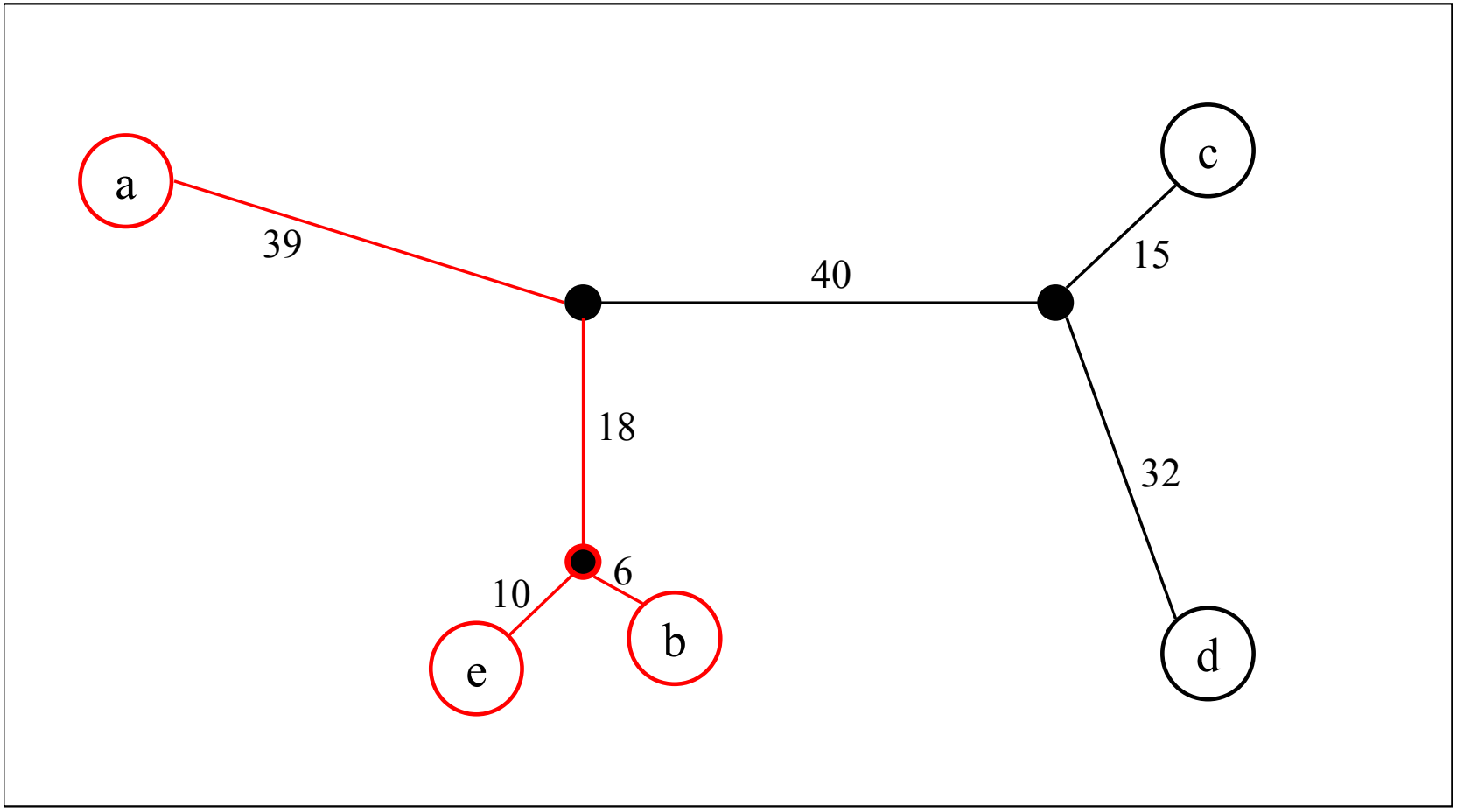
Árvore Aditiva



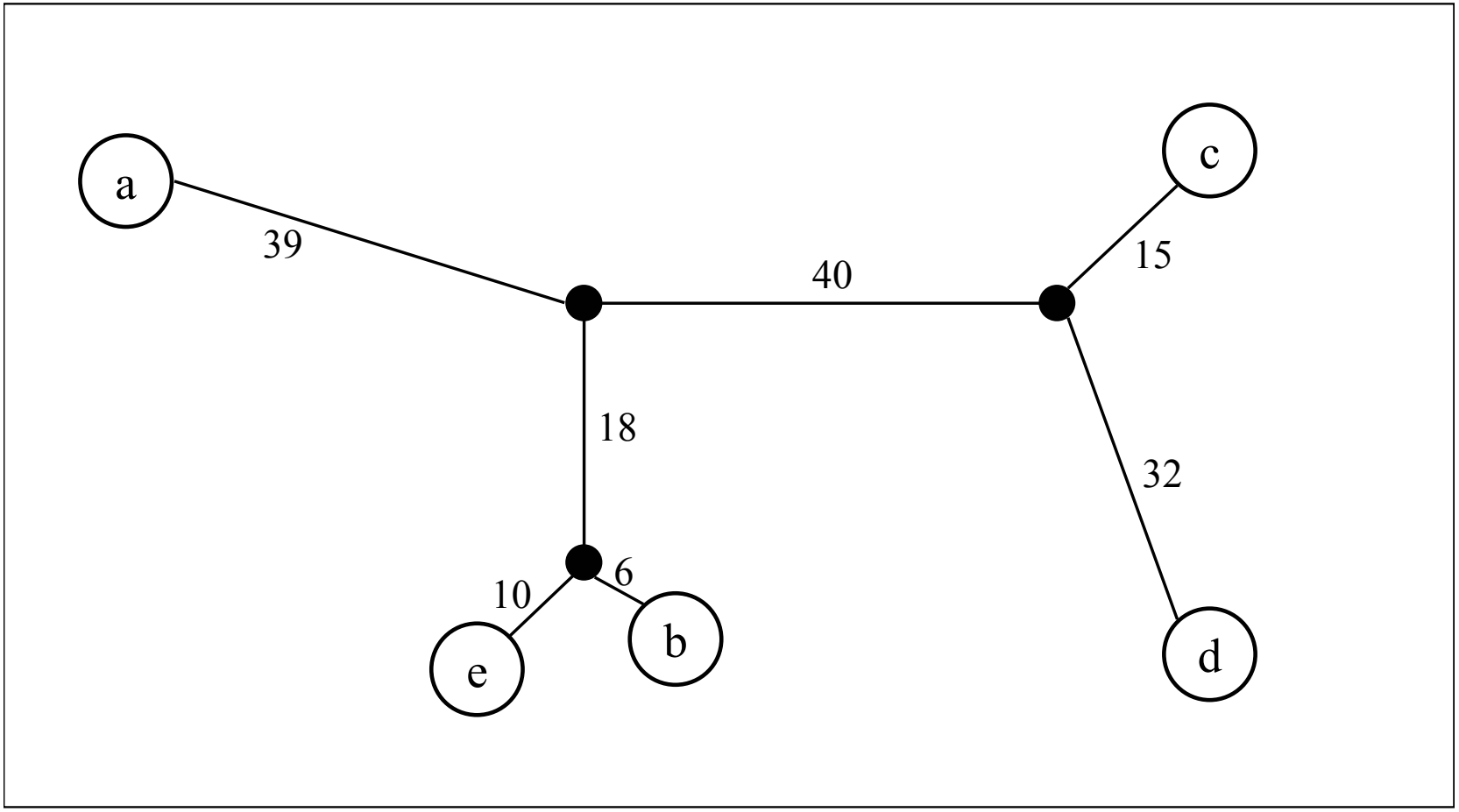
Árvore Aditiva



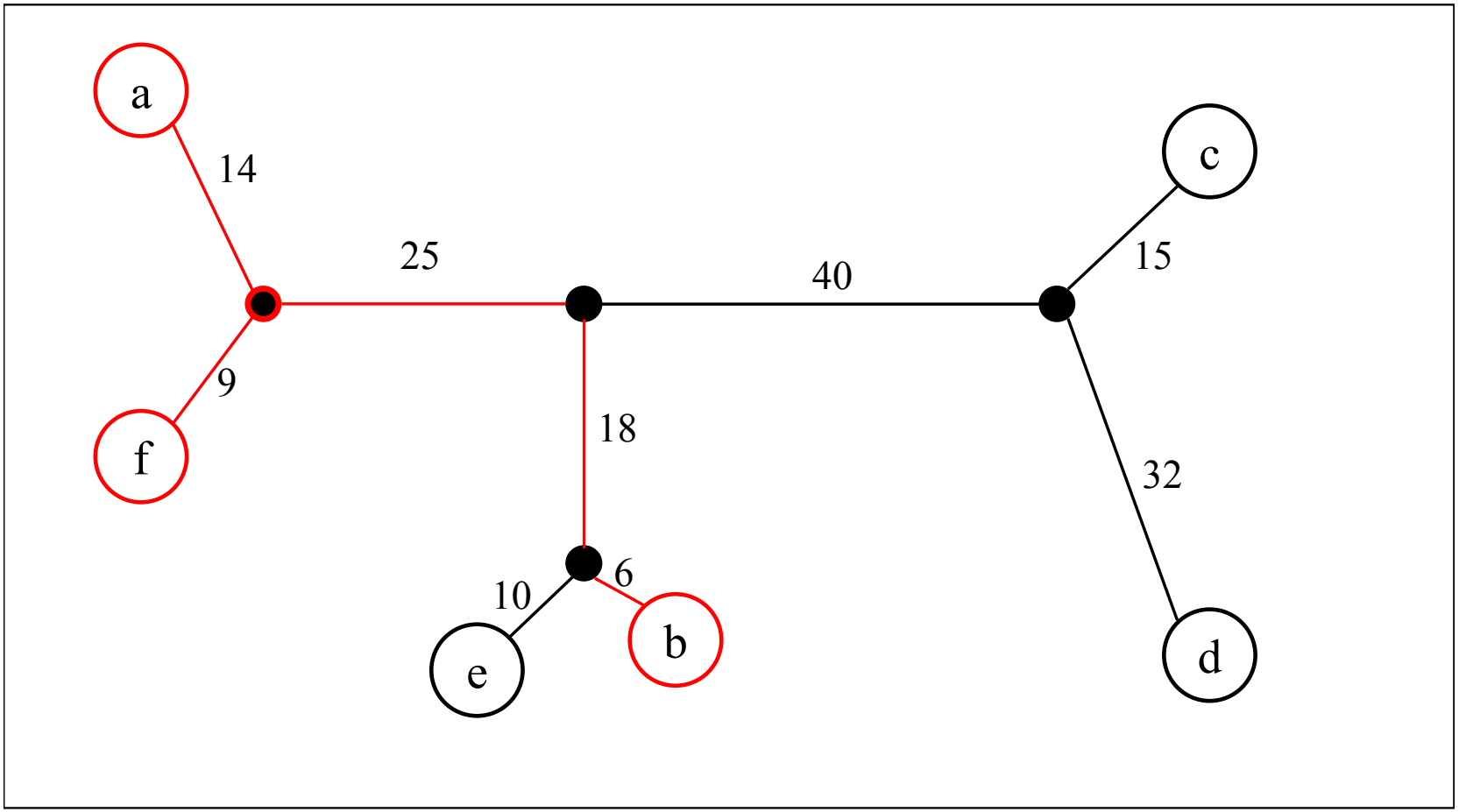
Árvore Aditiva



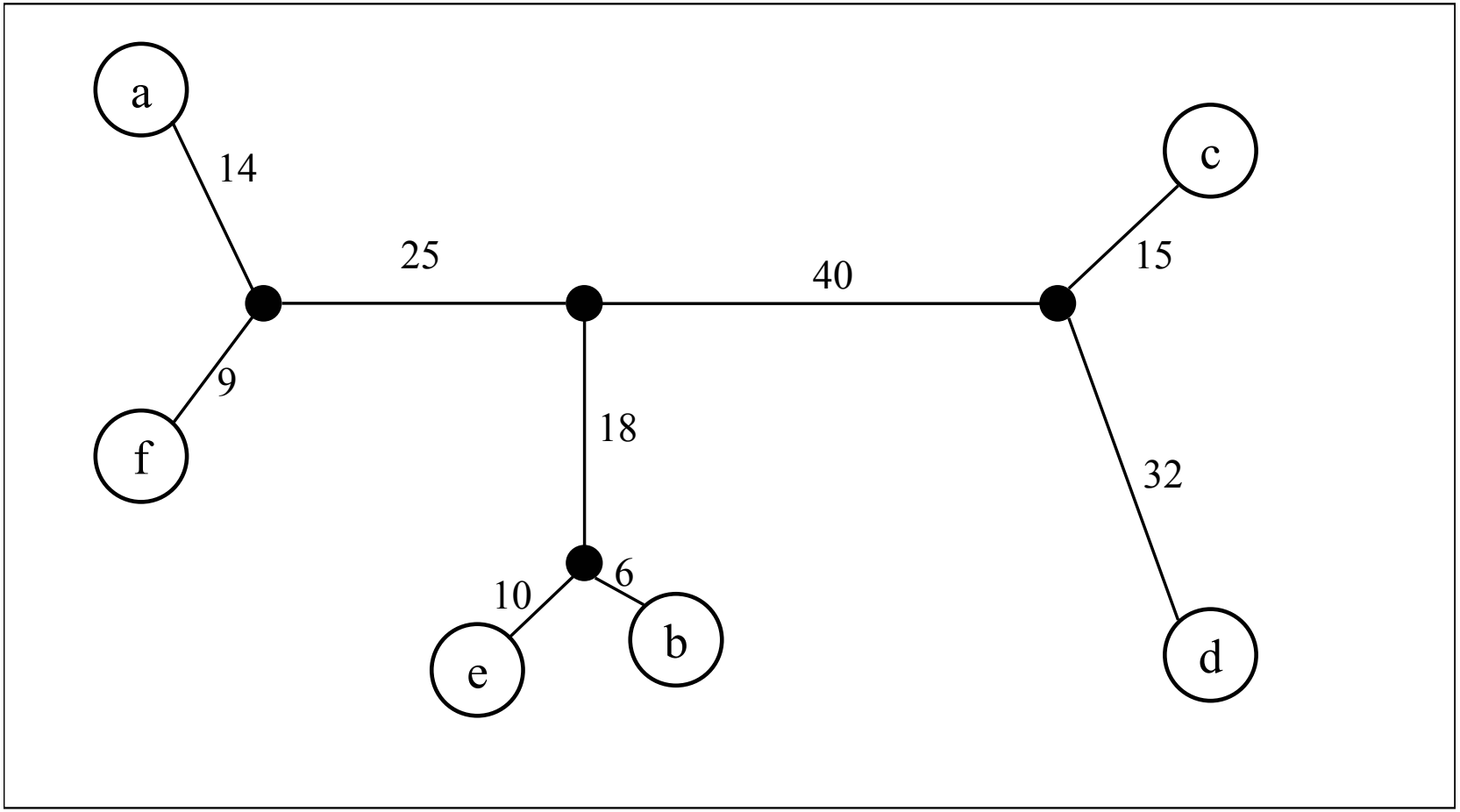
Árvore Aditiva



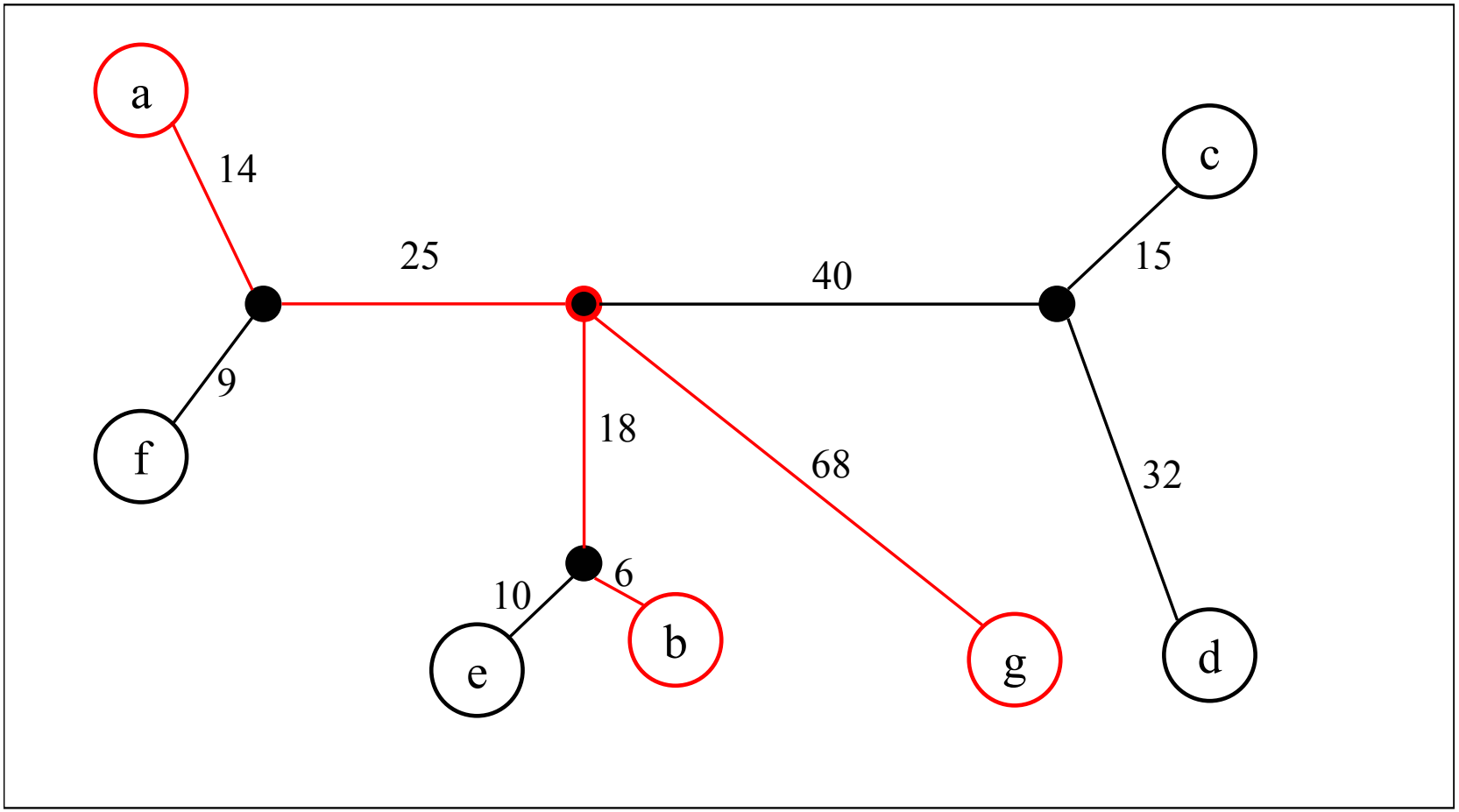
Árvore Aditiva



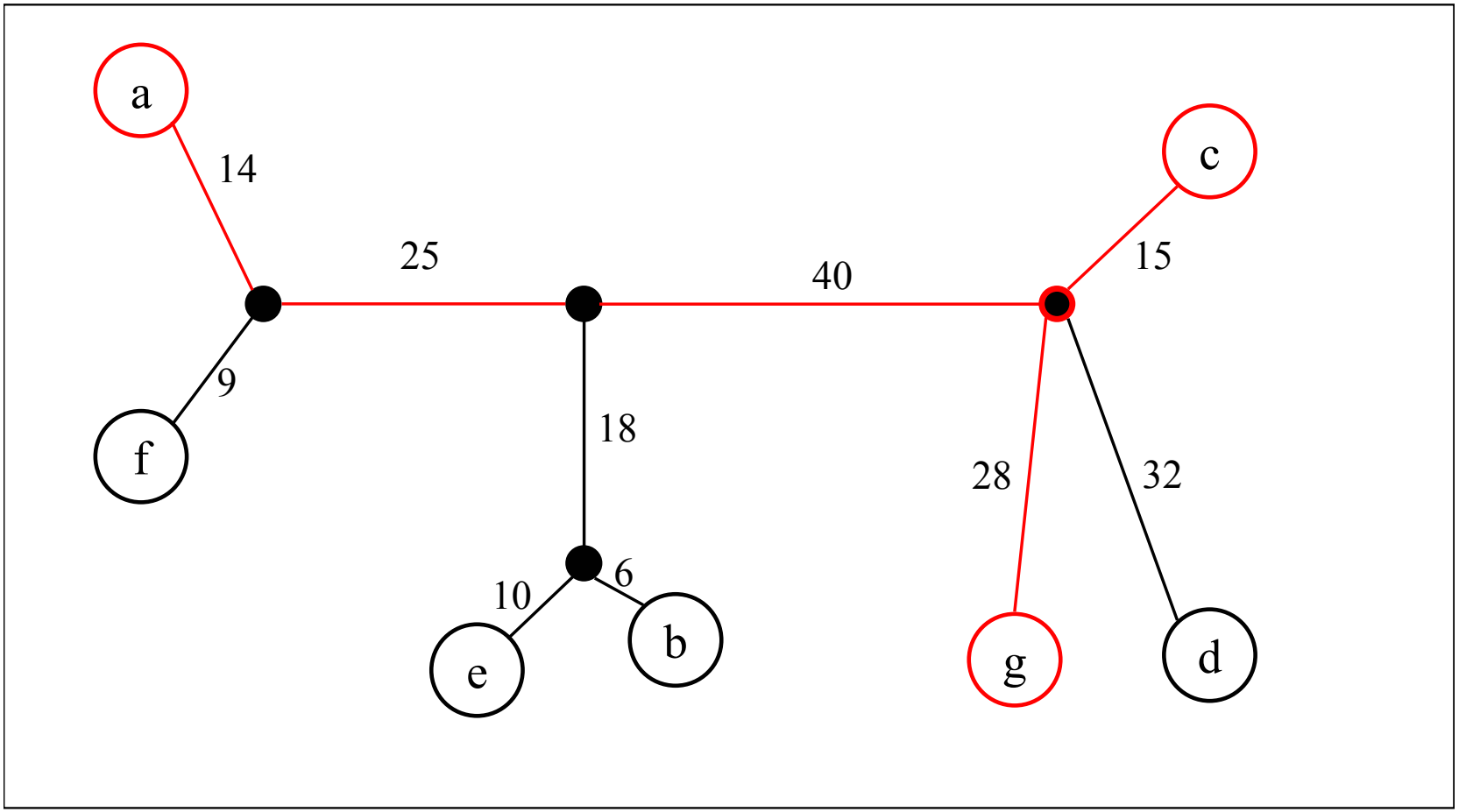
Árvore Aditiva



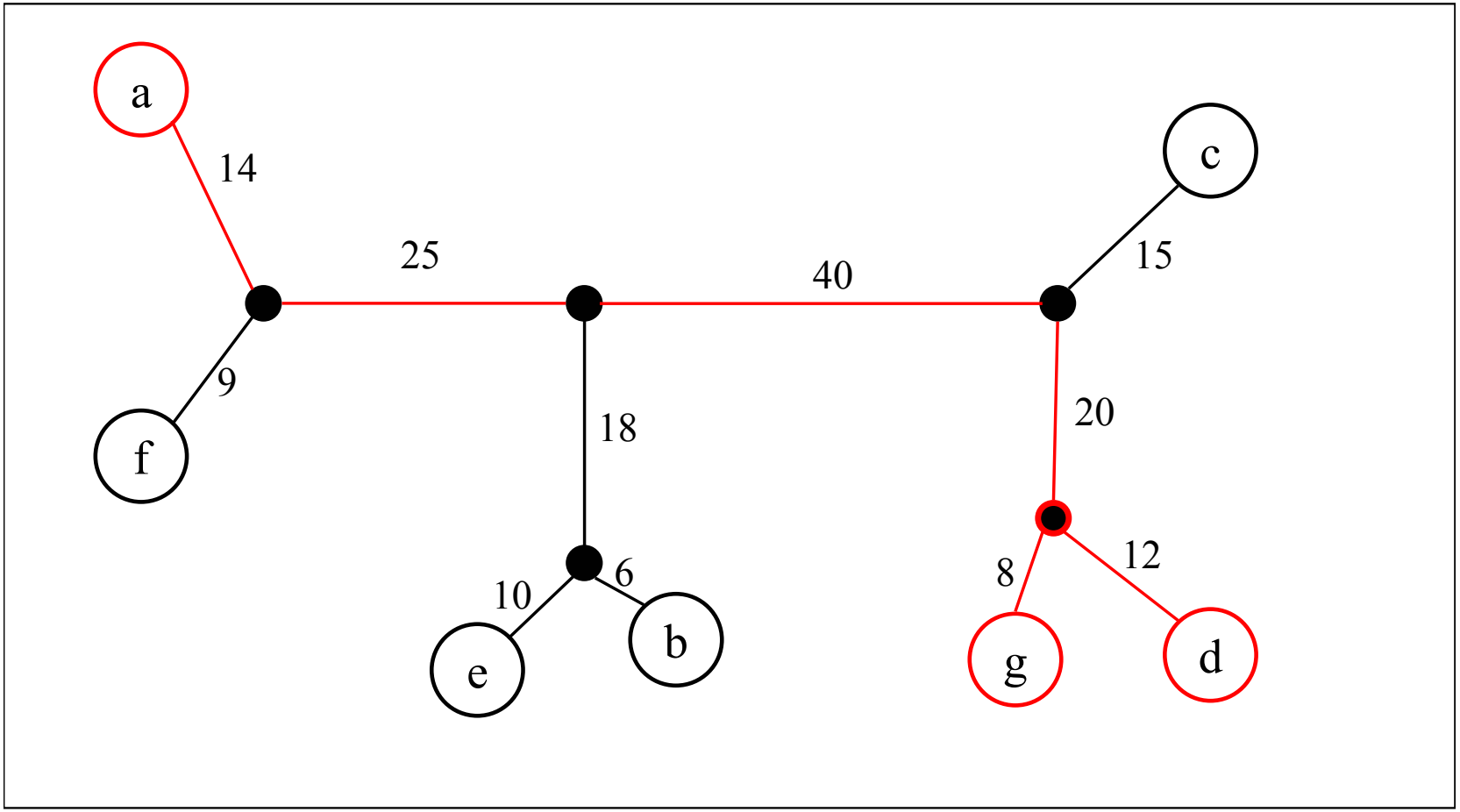
Árvore Aditiva



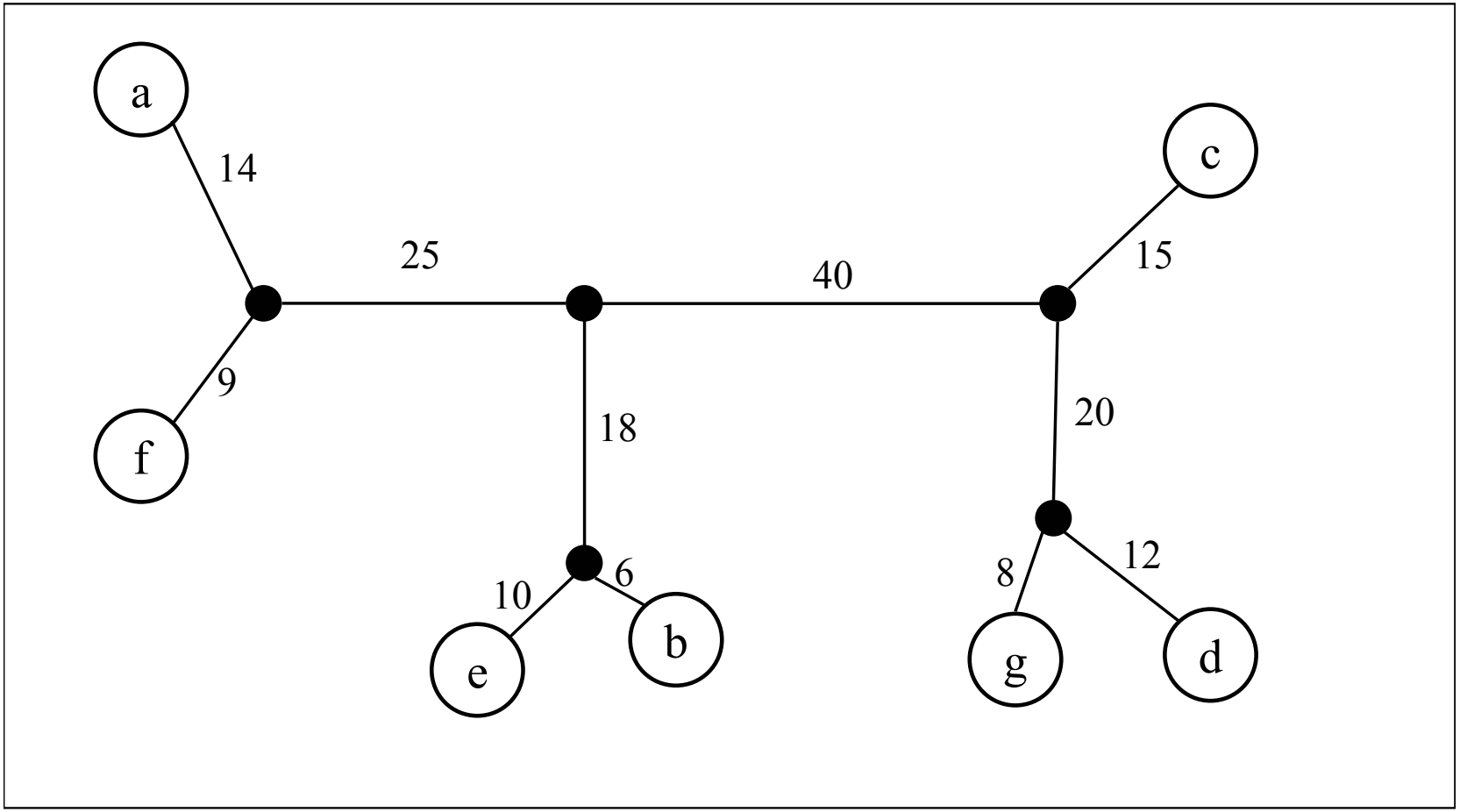
Árvore Aditiva



Árvore Aditiva



Árvore Aditiva



Algoritmo para Construção de Árvores Aditivas

- A partir de uma árvore formada por 3 vértices (quaisquer), adicione novos vértices, um a um, seguindo os seguintes passos:
    1. Escolha dois vértices quaisquer da árvore previamente construída (chame estes vértices de x e y).
    2. Calcule onde o novo vértice z deverá ser incluído, em relação ao caminho entre x e y.
    3. Se a inserção do novo vértice gerar um novo vértice interno c, entre os vértices c_1 e c_2, remova a aresta (c_1, c_2), insira os vértices c e z e as arestas (c_1, c), (c, c_2) e (c, z).
    4. Caso contrário, se existir um vértice y' da árvore previamente construída (e ainda não descartado na inserção do vértice corrente), chame-o de y e volte ao passo 2.
    5. Caso contrário, insira o vértice z e aresta (c, z), onde c é o nó interno do caminho entre x e y onde z deve ser incluído.

Algoritmo para Construção de Árvores Aditivas

- Complexidade (pior caso): $\sum_{k=4}^{n}(k-2)\Theta(k) = \Theta(n^3)$.
- Algoritmo proposto por Waterman, Smith, Singh e Beyer, em 1977.
- Annete Dobson, em 1974, provou que, dada uma matriz aditiva, existe uma única árvore aditiva (a menos de contrações de caminhos).

Como Verificar se uma Matriz de Distância é Aditiva

- Método Ingênuo:
  - Faça o teste para todo conjunto de 4 elementos, conforme teorema previamente visto.
  - Complexidade: $\binom{n}{4} \times \Theta(1) = \Theta(n^4)$.

Como Verificar se uma Matriz de Distância é Aditiva

- Abordagem Alternativa:
  - Dada uma matriz de distância qualquer, construa uma árvore, de acordo com o algoritmo para matrizes aditivas previamente estudado.
    - Complexidade: $\Theta(n^3)$.
  - Para cada vértice da árvore, calcule a distância para os demais vértices da árvore, usando algoritmo de caminhos mínimos para grafo acíclicos.
    - Complexidade: $n \times \Theta(n) = \Theta(n^2)$.
  - Para cada par de vértice da árvore, compare as distância computadas através da árvore com as distâncias da matriz original.
    - Complexidade: $\Theta(n^2)$.
  - Total: $\Theta(n^3)$.

Árvores Aditivas Compactas

Árvores Aditivas Compactas

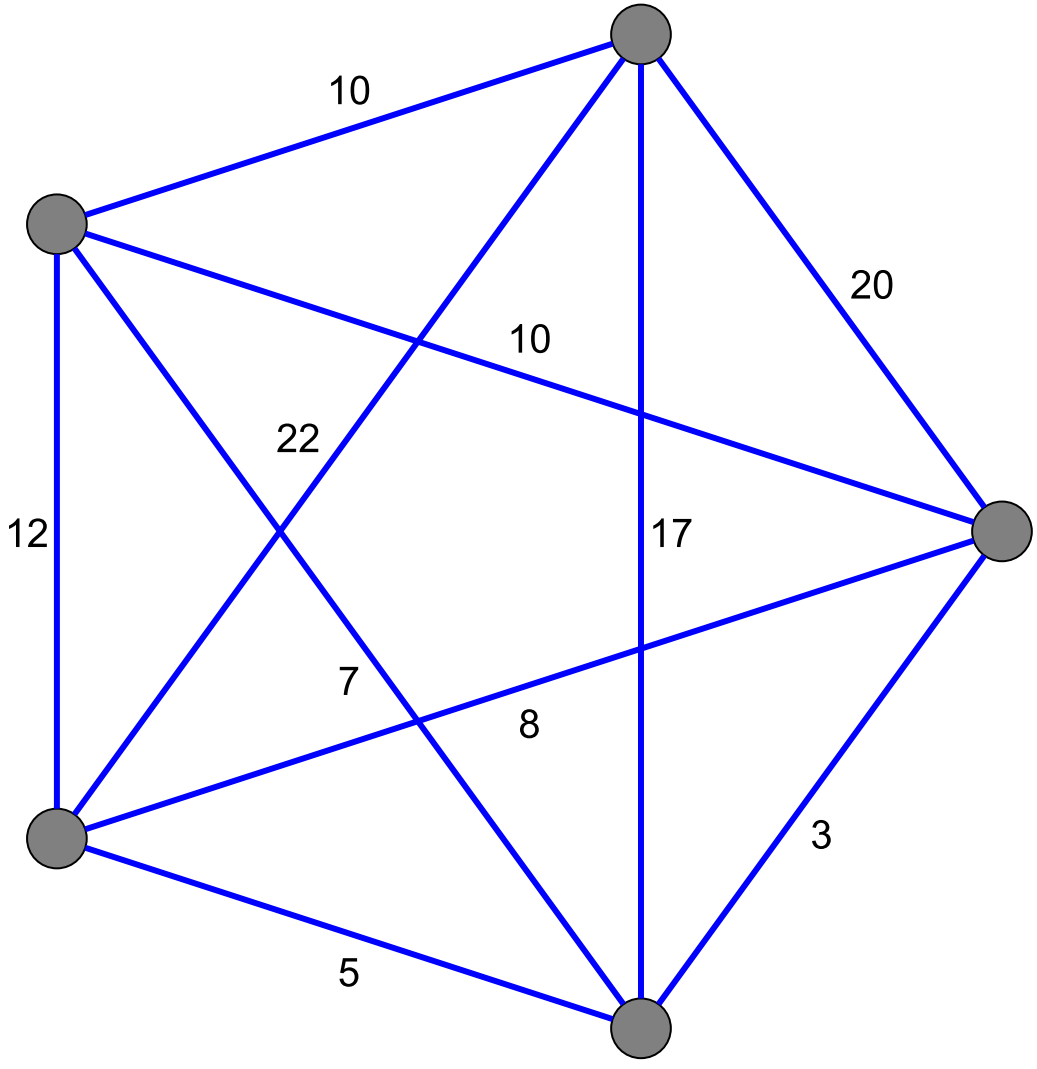
Definição

Uma árvore aditiva $T = (V, E, d)$ é chamada compacta se $\mathcal{A} = V$.

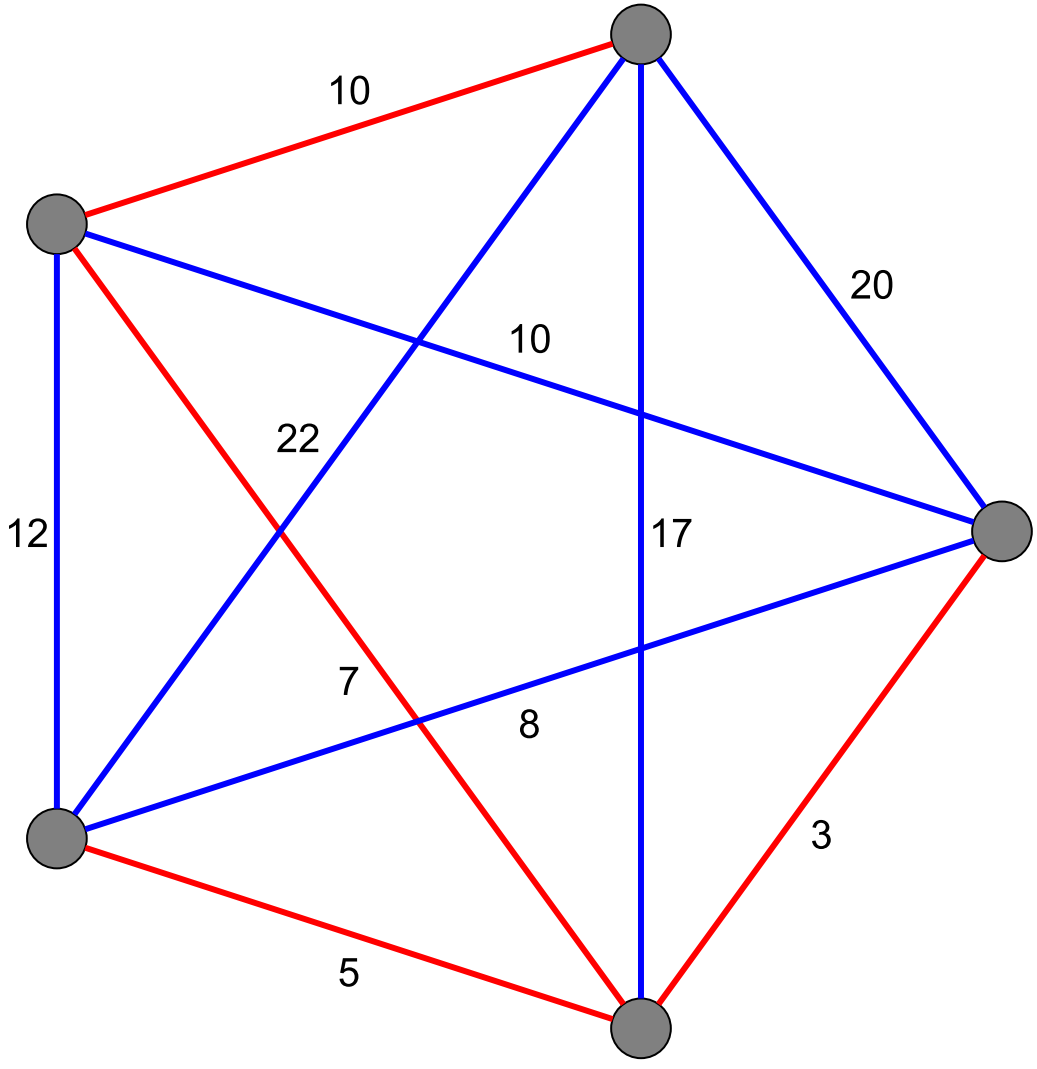
Teorema

Seja $G(V, E)$ o grafo completo onde os vértices representam os objetos de $\mathcal{A}$ e as arestas representam as distâncias métricas entre todos os pares de objetos. O grafo $G(V, E)$ é chamado Grafo de Distâncias. Se existe uma árvore compacta aditiva $T = (V, E', d)$, com $E' \subseteq E$, para $\mathcal{A}$ com respeito a δ, então T é a única Árvore Geradora Mínima do grafo $G(V, E)$.

Árvores Aditivas Compactas



Árvores Aditivas Compactas



Como Construir Árvores Aditivas Compactas

- Como construir uma Árvore Aditiva Compacta (caso ela exista):
  - Execute o algoritmo de Prim para Árvore Geradora Mínima: $\Theta(n^2)$.
  - Usando algoritmo de caminhos mínimos para grafos acíclicos, calcule a distância entre todos os pares de vértices da Árvore Geradora Mínima: $n \times \Theta(n) = \Theta(n^2)$.
  - Para cada par de vértice $i, j \in V$, teste se $dist(i, j) = \delta(i, j)$: $\Theta(n^2)$.
  - Complexidade: $\Theta(n^2)$.

Árvores Ultramétricas

Distância Ultramétrica

Definição

Seja $\mathcal{A}$ um conjunto de objetos e $\delta : \mathcal{A} \times \mathcal{A} \to \mathbb{R}^+$ uma métrica para $\mathcal{A}$. Então δ é uma ultramétrica para $\mathcal{A}$ se satisfaz a seguinte condição:

- *Para toda trinca $a, b, c \in \mathcal{A}$, ou $\delta(a, b) \leq \delta(a, c) = \delta(c, b)$ ou $\delta(a, c) \leq \delta(a, b) = \delta(b, c)$ ou $\delta(b, c) \leq \delta(b, a) = \delta(a, c)$.*

Lema

Uma matriz de distância M é ultramétrica se e somente se no grafo completo G correspondente, a aresta de maior peso, em qualquer ciclo, não é única.

Observação

Toda distância ultramétrica é uma distância aditiva.

Árvores Ultramétricas

Definição

Seja $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ um conjunto de objetos. Uma árvore ponderada $T = (V, E, d)$ com raiz r e função de peso associada às arestas $d : E \to \mathbb{R}^+$ é uma árvore ultramétrica para $\mathcal{A}$ se satisfaz as seguintes condições:

- *T é uma árvore aditiva para o conjunto $\mathcal{A}$ e a distância d.*
- *T é uma árvore binária, ou seja, cada vértice interno de T possui exatamente dois filhos.*
- *T possui exatamente n folhas, rotuladas com $\{a_1, a_2, \ldots, a_n\}$.*
- *A soma dos pesos das arestas de qualquer caminho da raiz r a qualquer folha de T é sempre o mesmo.*

Árvores Ultramétricas

Teorema

Seja $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ um conjunto de objetos e $\delta : \mathcal{A} \times \mathcal{A} \to \mathbb{R}^+$ uma função de comparação para os objetos de $\mathcal{A}$, logo existe uma árvore ultramétrica para $\mathcal{A}$ se e somente se δ for ultramétrica.

UPGMA

UPGMA

- Dado um conjunto de objetos $\mathcal{A}$ e uma função ultramétrica δ para $\mathcal{A}$, o algoritmo **UPGMA** (*Unweighted Pair Group Method with Arithmetic Mean*), proposto por Robert Sokal e Charles Michener em 1958, calcula a árvore ultramétrica para $\mathcal{A}$ em $O(n^3)$.
- A cada iteração, o algoritmo **UPGMA** agrupa os dois objetos mais próximos entre si, de acordo com a função de distância δ.

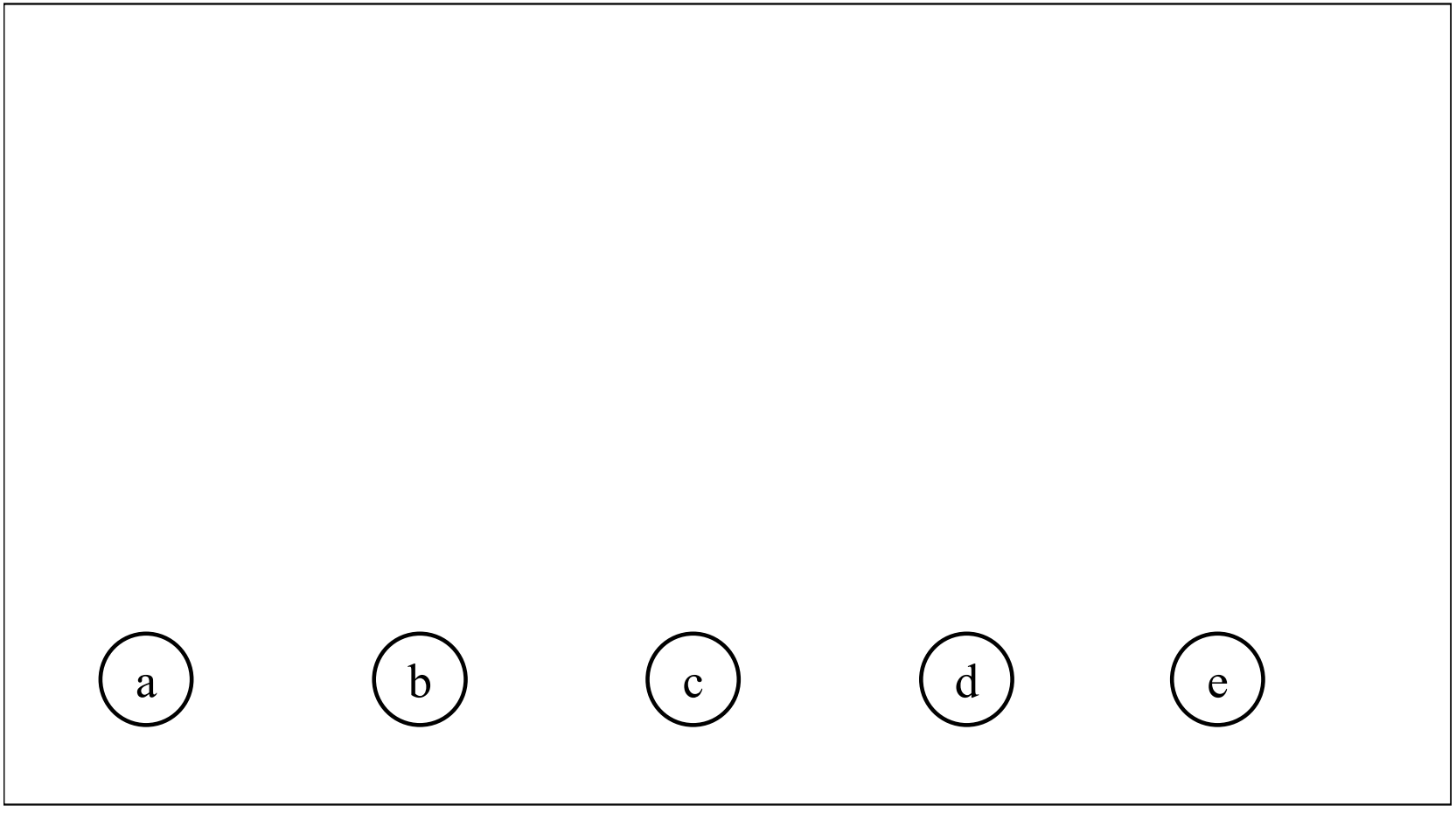
Algoritmo 1: UPGMA

Input: $\mathcal{A}, n, \delta$
$X \leftarrow \{\{a_i\}, \{a_2\}, \ldots, \{a_n\}\}$
for *all* $i, j \in [1..n]$ **do** $dist(\{a_i\}, \{a_j\}) = \delta(a_i, a_j)$;
for *all* $i \in [1..n]$ **do** $height(\{a_i\}) \leftarrow 0$;
$V \leftarrow X$; $E \leftarrow \emptyset$
while $|X| \geq 2$ **do**
    $min \leftarrow \infty$
    for *all* $x_i, x_j \in X$ **do**
        if $x_i \neq x_j$ *and* $dist(x_i, x_j) < min$ **then**
            $min \leftarrow dist(x_i, x_j)$
            $C_1 \leftarrow x_i$; $C_2 \leftarrow x_j$; $D \leftarrow C_1 \cup C_2$;
        end
    end
    $X \leftarrow (X - \{C_1, C_2\}) \cup \{D\}$
    for *all* $C \in X$ **do** $dist(D, C) \leftarrow dist(C, D) \leftarrow (dist(C_1, C) + dist(C_2, C))/2$;
    $V \leftarrow V \cup \{D\}$
    $E \leftarrow E \cup \{(D, C_1), (D, C_2)\}$
    $height(D) \leftarrow dist(C_1, C_2)/2$
    $d(D, C_1) \leftarrow height(D) - height(C_1)$
    $d(D, C_2) \leftarrow height(D) - height(C_2)$
end
return $T = (V, E, d)$

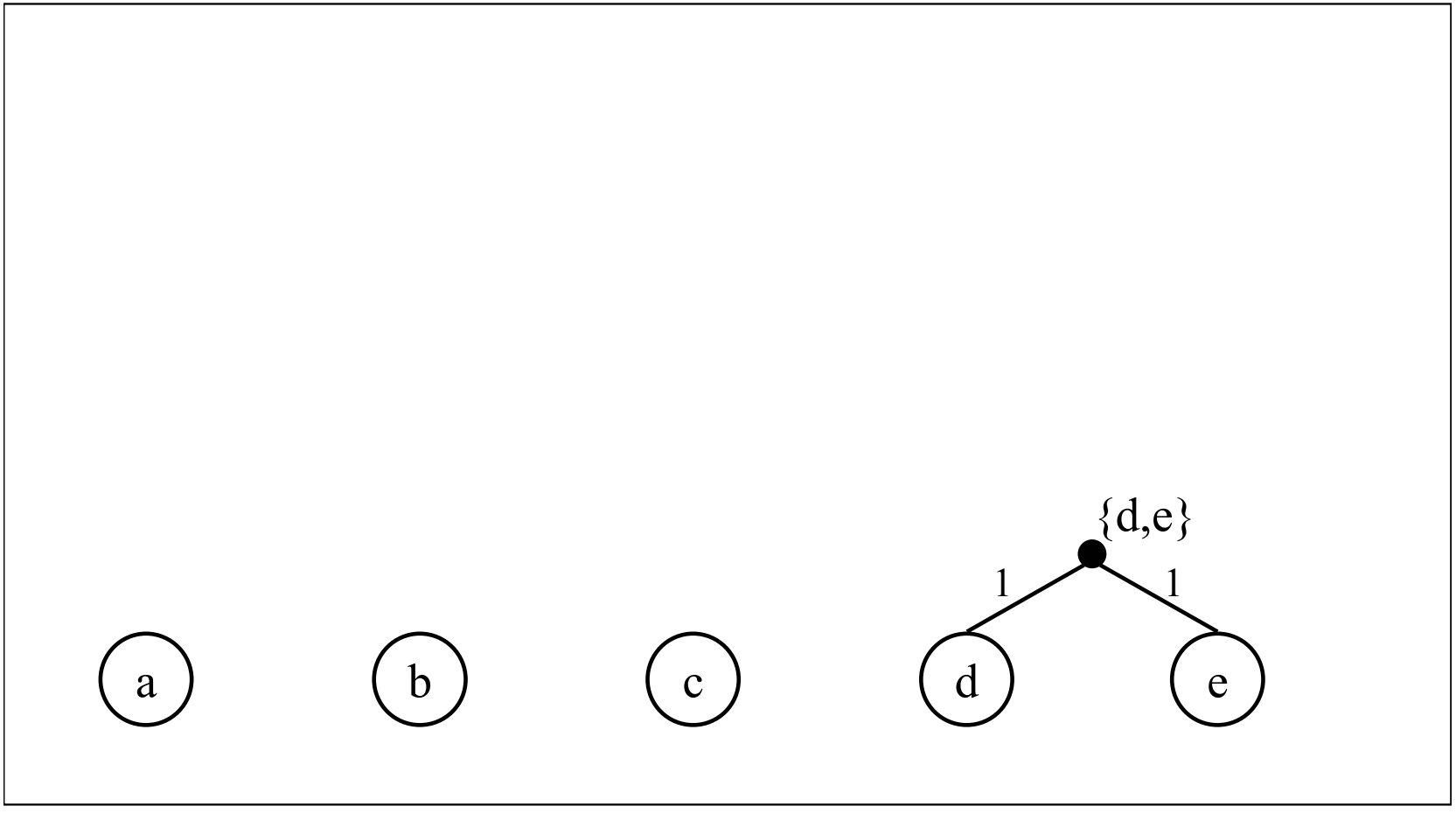
Árvore Ultramétrica

	a	**b**	**c**	**d**	**e**
a	0	12	12	12	12
b	12	0	4	6	6
c	12	4	0	6	6
d	12	6	6	0	2
e	12	6	6	2	0

Árvore Ultramétrica



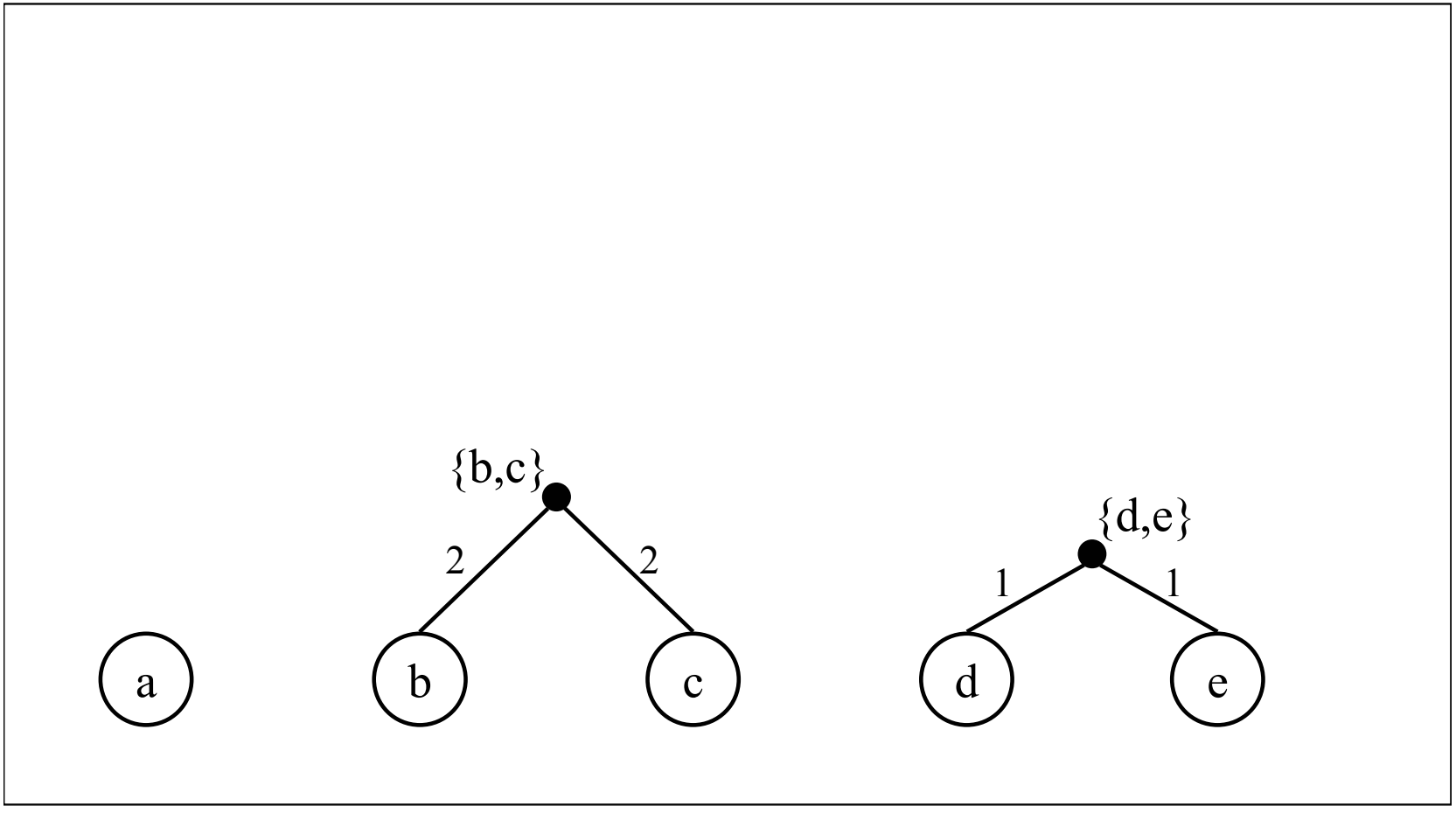
Árvore Ultramétrica



Árvore Ultramétrica

	a	**b**	**c**	**d,e**
a	0	12	12	12
b	12	0	4	6
c	12	4	0	6
d,e	12	6	6	0

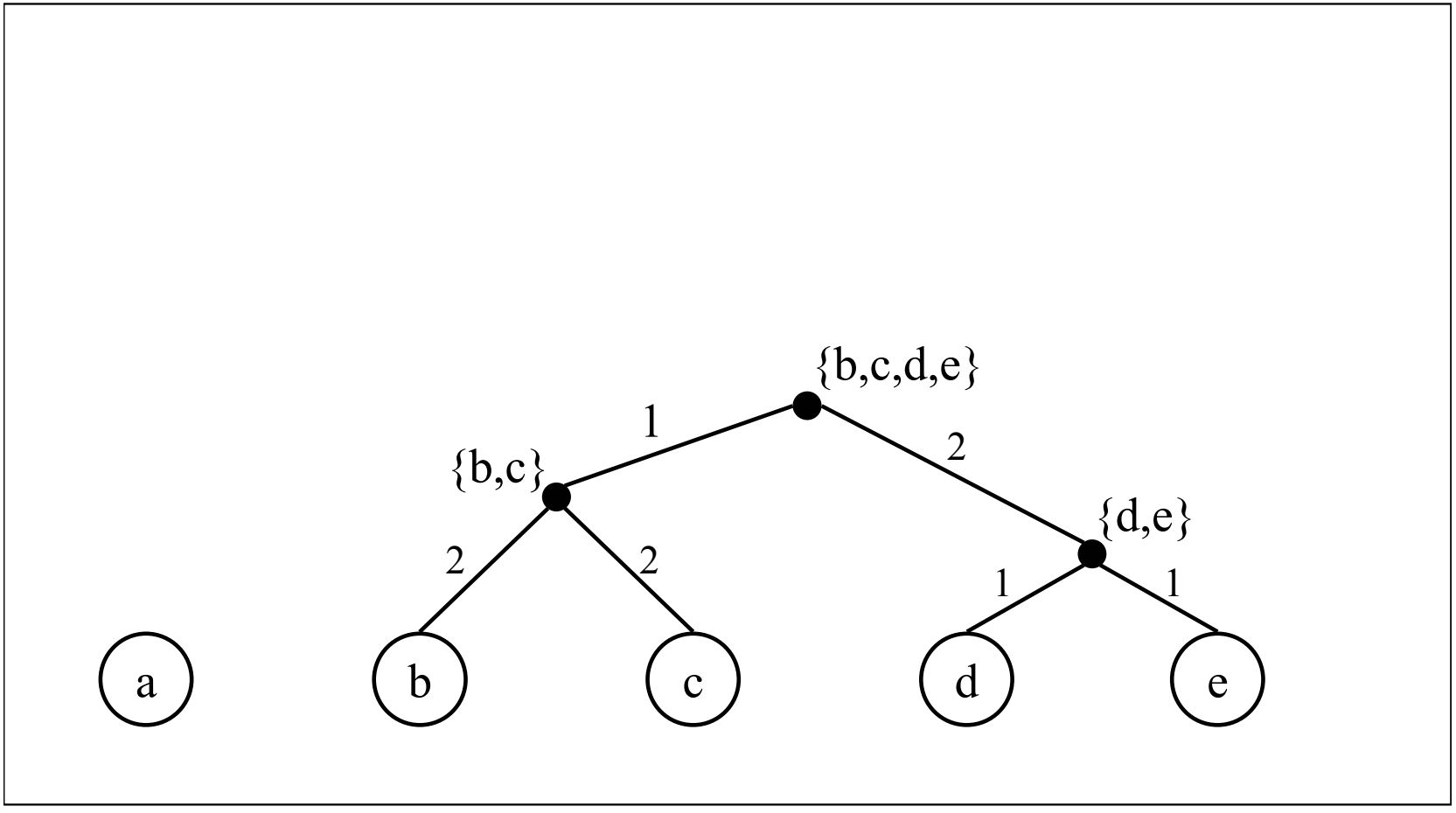
Árvore Ultramétrica



Árvore Ultramétrica

	a	**b,c**	**d,e**
a	0	12	12
b,c	12	0	6
d,e	12	6	0

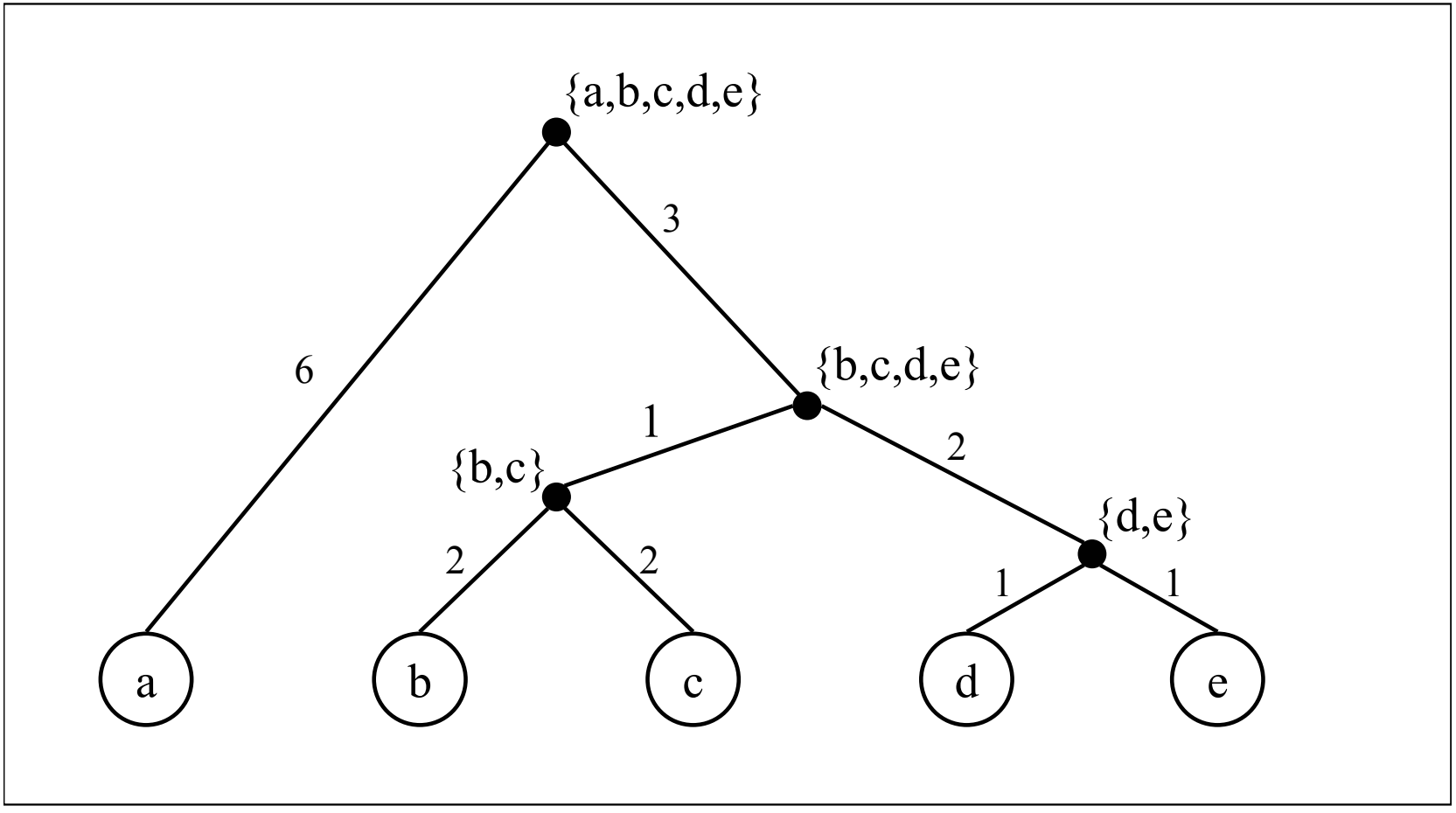
Árvore Ultramétrica



Árvore Ultramétrica

	a	**b,c,d,e**
a	0	12
b,c,d,e	12	0

Árvore Ultramétrica



Neighbor-Joining

Neighbor-Joining

- Algoritmo para construção de árvores filogenéticas sem raiz.
- Se a matriz for aditiva, o algoritmo constrói uma árvore aditiva.
- Para matrizes não aditivas, geralmente produz boas árvores (em termos topológicos).
- Para cada objeto, o algoritmo calcula uma medida $u(x)$ de "separação" entre o elemento x e todos os demais elementos do conjunto.

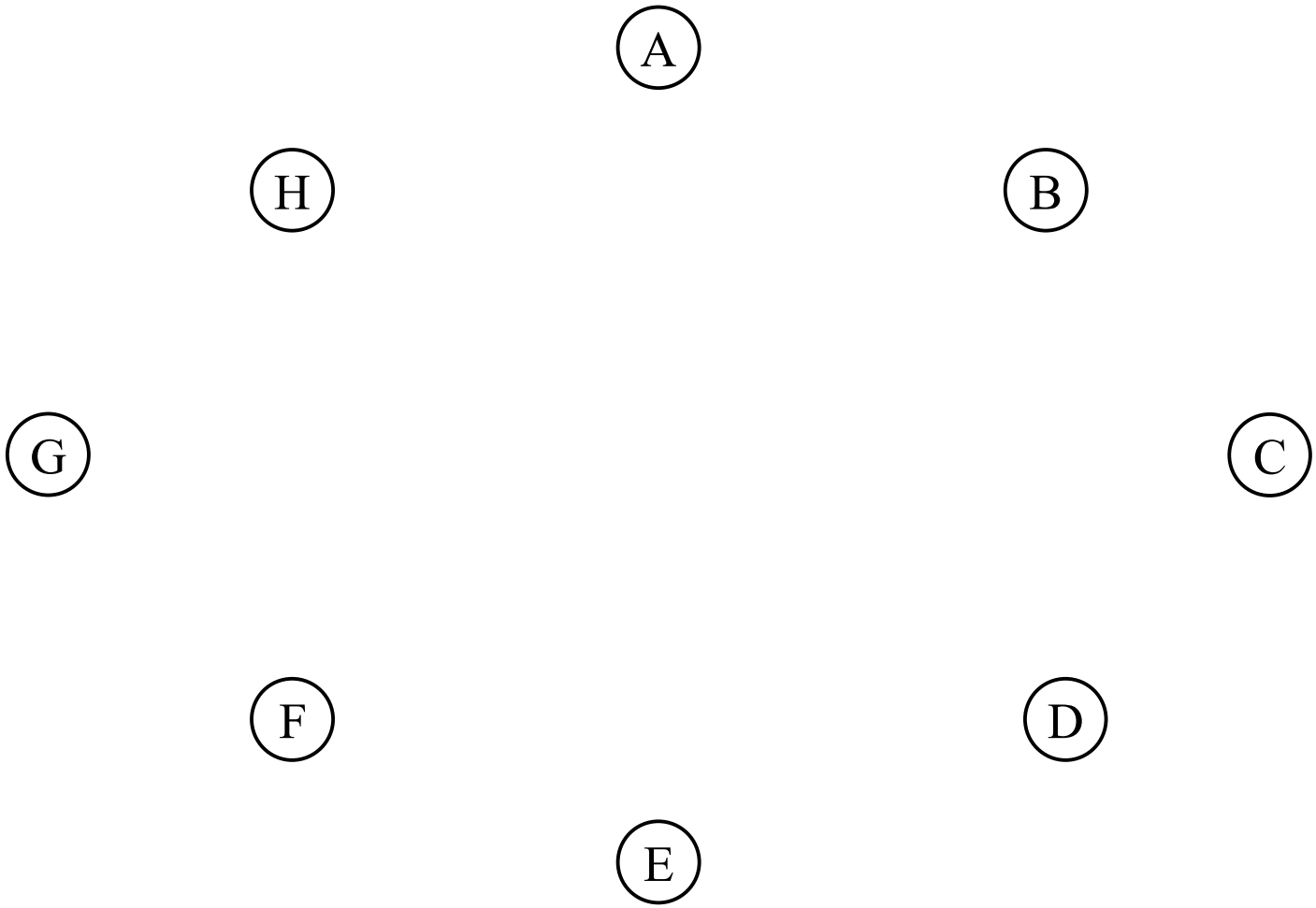
Neighbor-Joining

- A cada iteração, o algoritmo `Neighbor-Joining` tenta agrupar os dois objetos mais próximos entre si, de acordo com a função de distância δ, e ao mesmo tempo mais "separados" dos objetos restantes, de acordo com a função de "separação" u, ou seja:
  - Desejamos agrupar o par (i, j) de objetos, tal que o valor $S(i, j) = \delta(i, j) - u(i) - u(j)$ seja o menor possível.
- O método `Neighbor-Joining` foi proposto em 1987 por Naruya Saitou e Masatoshi Nei.
- Em 1988, James Studier e Karl Keppler mostraram que é possível implementar o método `Neighbor-Joining` com complexidade $O(n^3)$.

Algoritmo 2: Neighbor-Joining

Input: $\mathcal{A}, n, \delta$
$X \leftarrow \{\{a_1\}, \{a_2\}, \ldots, \{a_n\}\}$
for *all* $i, j \in [1..n]$ **do** $d(\{a_i\}, \{a_j\}) \leftarrow \delta(a_i, a_j)$;
$V \leftarrow X$; $E \leftarrow \emptyset$
while $|X| > 2$ **do**
  for *all* $x \in X$ **do** $u(x) \leftarrow \frac{1}{|X|-2} \sum_{x' \in X} d(x, x')$;
  $min \leftarrow \infty$
  for *all* $x_i, x_j \in X$ **do**
    if $x_i \neq x_j$ *and* $d(x_i, x_j) - u(x_i) - u(x_j) < min$ **then**
      $min \leftarrow d(x_i, x_j) - u(x_i) - u(x_j)$
      $C_1 \leftarrow x_i$; $C_2 \leftarrow x_j$; $D \leftarrow C_1 \cup C_2$
    end
  end
  $X \leftarrow (X - \{C_1, C_2\}) \cup \{D\}$
  for *all* $C \in X$ **do** $d(D, C) \leftarrow d(C, D) \leftarrow (d(C_1, C) + d(C_2, C) - d(C_1, C_2))/2$;
  $V \leftarrow V \cup \{D\}$; $E \leftarrow E \cup \{(D, C_1), (D, C_2)\}$
  $dist(D, C_1) \leftarrow (d(C_1, C_2) + u(C_1) - u(C_2))/2$
  $dist(D, C_2) \leftarrow (d(C_1, C_2) + u(C_2) - u(C_1))/2$
end
$E \leftarrow E \cup \{(C_1, C_2)\}$
$dist(C_1, C_2) \leftarrow d(C_1, C_2)$
return $T = (V, E, dist)$

Neighbor-Joining



Neighbor-Joining

d	A	B	C	D	E	F	G	H	u
A		7	8	11	13	16	13	17	14,2
B	7		5	8	10	13	10	14	11,2
C	8	5		5	7	10	7	11	8,8
D	11	8	5		8	11	8	12	10,5
E	13	10	7	8		5	6	10	9,8
F	16	13	10	11	5		9	13	12,8
G	13	10	7	8	6	9		8	10,2
H	17	14	11	12	10	13	8		14,2
u	14,2	11,2	8,8	10,5	9,8	12,8	10,2	14,2	

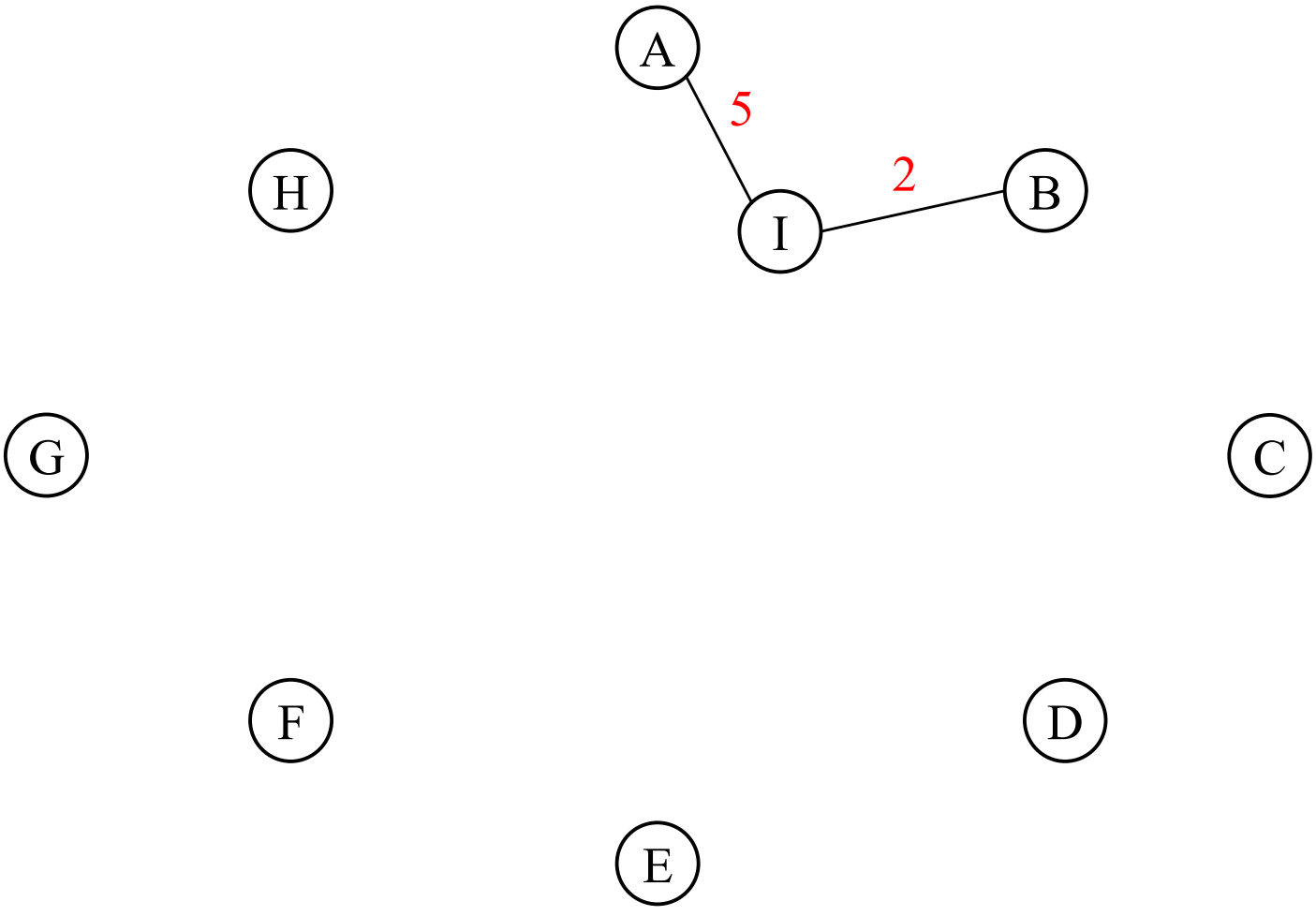
Neighbor-Joining

S	A	B	C	D	E	F	G	H	u
A		-18,3	-15,0	-13,7	-11,0	-11,0	-11,3	-11,3	14,2
B	-18,3		-15,0	-13,7	-11,0	-11,0	-11,3	-11,3	11,2
C	-15,0	-15,0		-14,3	-11,7	-11,7	-12,0	-12,0	8,8
D	-13,7	-13,7	-14,3		-12,3	-12,3	-12,7	-12,7	10,5
E	-11,0	-11,0	-11,7	-12,3		-17,7	-14,0	-14,0	9,8
F	-11,0	-11,0	-11,7	-12,3	-17,7		-14,0	-14,0	12,8
G	-11,3	-11,3	-12,0	-12,7	-14,0	-14,0		-16,3	10,2
H	-11,3	-11,3	-12,0	-12,7	-14,0	-14,0	-16,3		14,2
u	14,2	11,2	8,8	10,5	9,8	12,8	10,2	14,2	

Neighbor-Joining

d	A	B	C	D	E	F	G	H	u
A		7	8	11	13	16	13	17	14,2
B	7		5	8	10	13	10	14	11,2
C	8	5		5	7	10	7	11	8,8
D	11	8	5		8	11	8	12	10,5
E	13	10	7	8		5	6	10	9,8
F	16	13	10	11	5		9	13	12,8
G	13	10	7	8	6	9		8	10,2
H	17	14	11	12	10	13	8		14,2
u	14,2	11,2	8,8	10,5	9,8	12,8	10,2	14,2	
I	5	2	3	6	8	11	8	12	

Neighbor-Joining



Neighbor-Joining

d	C	D	E	F	G	H	I	u
C		5	7	10	7	11	3	8,6
D	5		8	11	8	12	6	10,0
E	7	8		5	6	10	8	8,8
F	10	11	5		9	13	11	11,8
G	7	8	6	9		8	8	9,2
H	11	12	10	13	8		12	13,2
I	3	6	8	11	8	12		9,6
u	8,6	10,0	8,8	11,8	9,2	13,2	9,6	

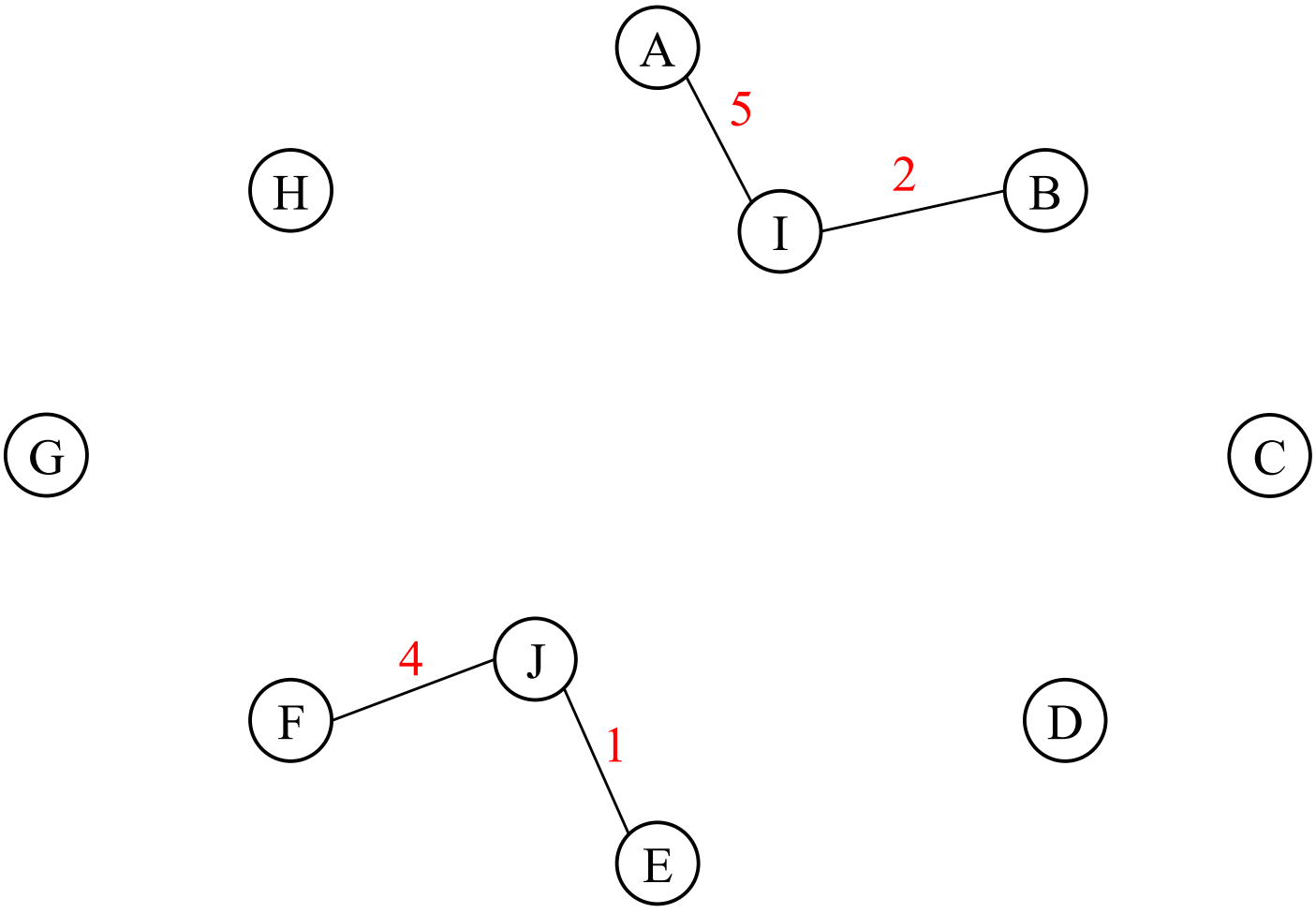
Neighbor-Joining

S	C	D	E	F	G	H	I	u
C		-13,6	-10,4	-10,4	-10,8	-10,8	-15,2	8,6
D	-13,6		-10,8	-10,8	-11,2	-11,2	-13,6	10,0
E	-10,4	-10,8		-15,6	-12,0	-12,0	-10,4	8,8
F	-10,4	-10,8	-15,6		-12,0	-12,0	-10,4	11,8
G	-10,8	-11,2	-12,0	-12,0		-14,4	-10,8	9,2
H	-10,8	-11,2	-12,0	-12,0	-14,4		-10,8	13,2
I	-15,2	-13,6	-10,4	-10,4	-10,8	-10,8		9,6
u	8,6	10,0	8,8	11,8	9,2	13,2	9,6	

Neighbor-Joining

d	C	D	E	F	G	H	I	u
C		5	7	10	7	11	3	8,6
D	5		8	11	8	12	6	10,0
E	7	8		5	6	10	8	8,8
F	10	11	5		9	13	11	11,8
G	7	8	6	9		8	8	9,2
H	11	12	10	13	8		12	13,2
I	3	6	8	11	8	12		9,6
u	8,6	10,0	8,8	11,8	9,2	13,2	9,6	
J	6	7	1	4	5	9	7	

Neighbor-Joining



Neighbor-Joining

d	C	D	G	H	I	J	u
C		5	7	11	3	6	8,0
D	5		8	12	6	7	9,5
G	7	8		8	8	5	9,0
H	11	12	8		12	9	13,0
I	3	6	8	12		7	9,0
J	6	7	5	9	7		8,5
u	8,0	9,5	9,0	13,0	9,0	8,5	

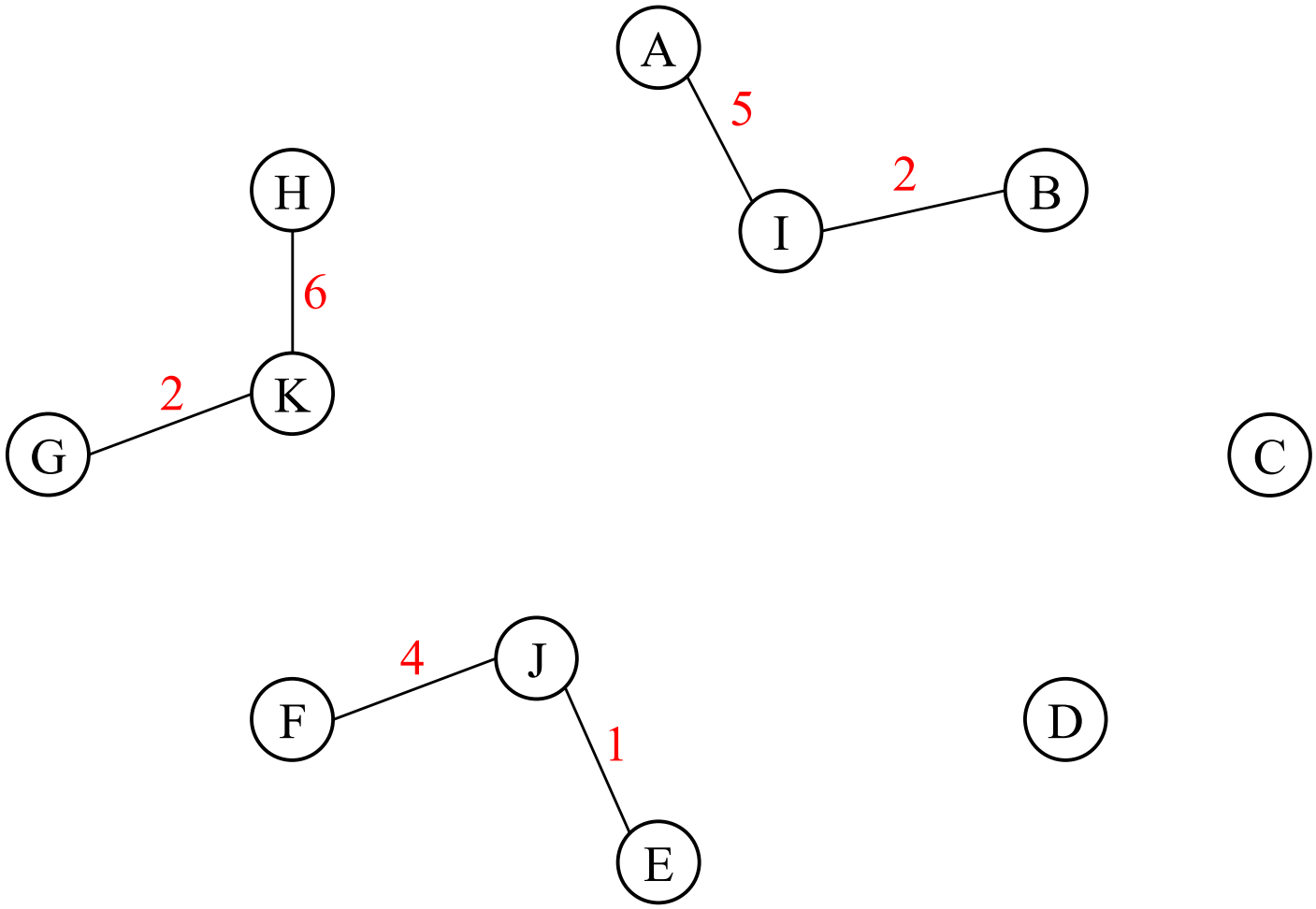
Neighbor-Joining

S	C	D	G	H	I	J	u
C		-12,5	-10,0	-10,0	-14,0	-10,5	8,0
D	-12,5		-10,5	-10,5	-12,5	-11,0	9,5
G	-10,0	-10,5		-14,0	-10,0	-12,5	9,0
H	-10,0	-10,5	-14,0		-10,0	-12,5	13,0
I	-14,0	-12,5	-10,0	-10,0		-10,5	9,0
J	-10,5	-11,0	-12,5	-12,5	-10,5		8,5
u	8,0	9,5	9,0	13,0	9,0	8,5	

Neighbor-Joining

d	C	D	G	H	I	J	u
C		5	7	11	3	6	8,0
D	5		8	12	6	7	9,5
G	7	8		8	8	5	9,0
H	11	12	8		12	9	13,0
I	3	6	8	12		7	9,0
J	6	7	5	9	7		8,5
u	8,0	9,5	9,0	13,0	9,0	8,5	
K	5	6	2	6	6	3	

Neighbor-Joining



Neighbor-Joining

d	C	D	I	J	K	u
C		5	3	6	5	6,3
D	5		6	7	6	8,0
I	3	6		7	6	7,3
J	6	7	7		3	7,7
K	5	6	6	3		6,7
u	6,3	8,0	7,3	7,7	6,7	

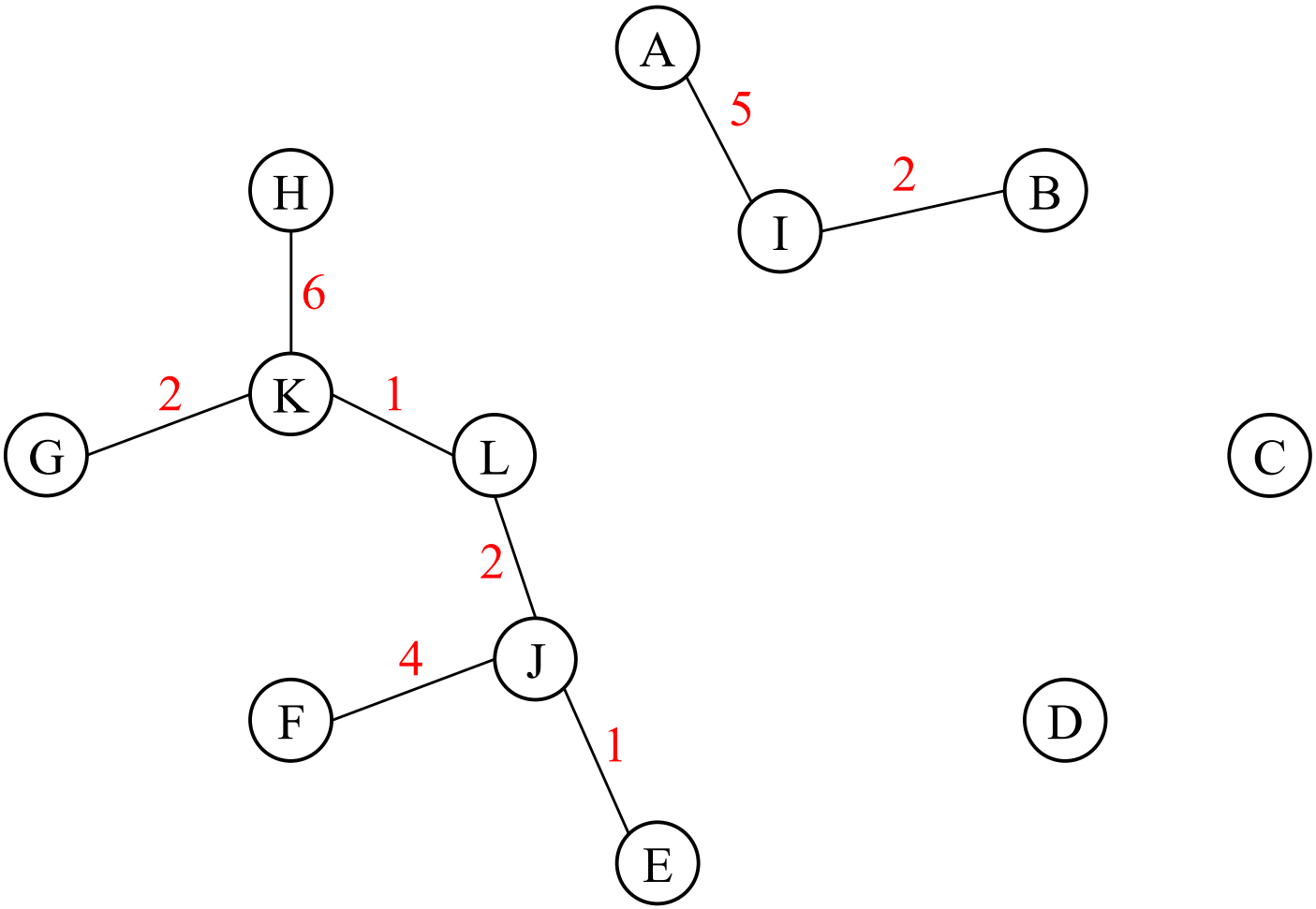
Neighbor-Joining

S	C	D	I	J	K	u
C		-9,3	-10,7	-8,0	-8,0	6,3
D	-9,3		-9,3	-8,7	-8,7	8,0
I	-10,7	-9,3		-8,0	-8,0	7,3
J	-8,0	-8,7	-8,0		-11,3	7,7
K	-8,0	-8,7	-8,0	-11,3		6,7
u	6,3	8,0	7,3	7,7	6,7	

Neighbor-Joining

d	C	D	I	J	K	u
C		5	3	6	5	6,3
D	5		6	7	6	8,0
I	3	6		7	6	7,3
J	6	7	7		3	7,7
K	5	6	6	3		6,7
u	6,3	8,0	7,3	7,7	6,7	
L	4	5	5	2	1	

Neighbor-Joining



Neighbor-Joining

d	C	D	I	L	u
C		5	3	4	6,0
D	5		6	5	8,0
I	3	6		5	7,0
L	4	5	5		7,0
u	6,0	8,0	7,0	7,0	

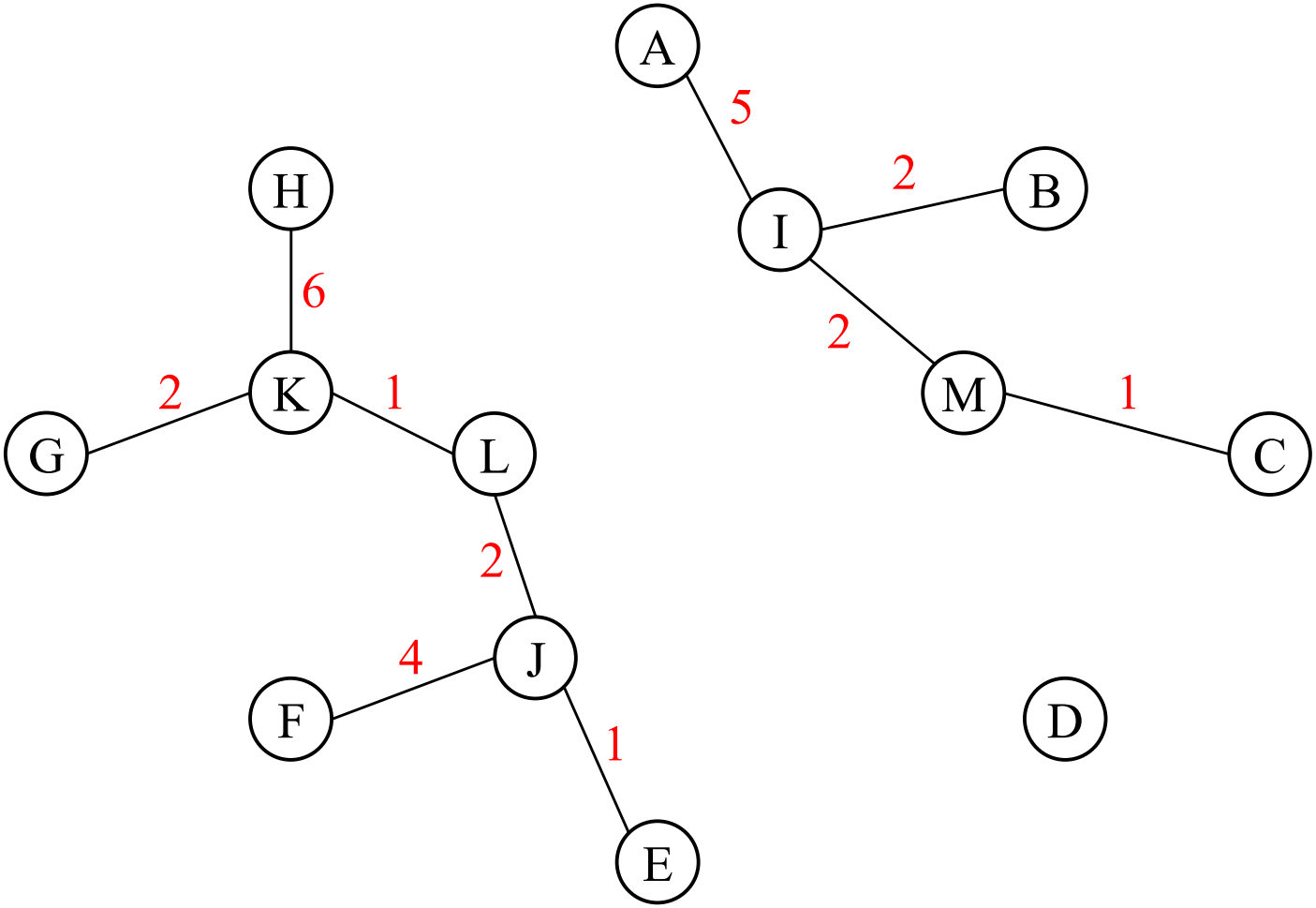
Neighbor-Joining

S	C	D	I	L	u
C		-9,0	-10,0	-9,0	6,0
D	-9,0		-9,0	-10,0	8,0
I	-10,0	-9,0		-9,0	7,0
L	-9,0	-10,0	-9,0		7,0
u	6,0	8,0	7,0	7,0	

Neighbor-Joining

d	C	D	I	L	u
C		5	3	4	6,0
D	5		6	5	8,0
I	3	6		5	7,0
L	4	5	5		7,0
u	6,0	8,0	7,0	7,0	
M	1	4	2	3	

Neighbor-Joining



Neighbor-Joining

d	D	L	M	u
D		5	4	9,0
L	5		3	8,0
M	4	3		7,0
u	9,0	8,0	7,0	

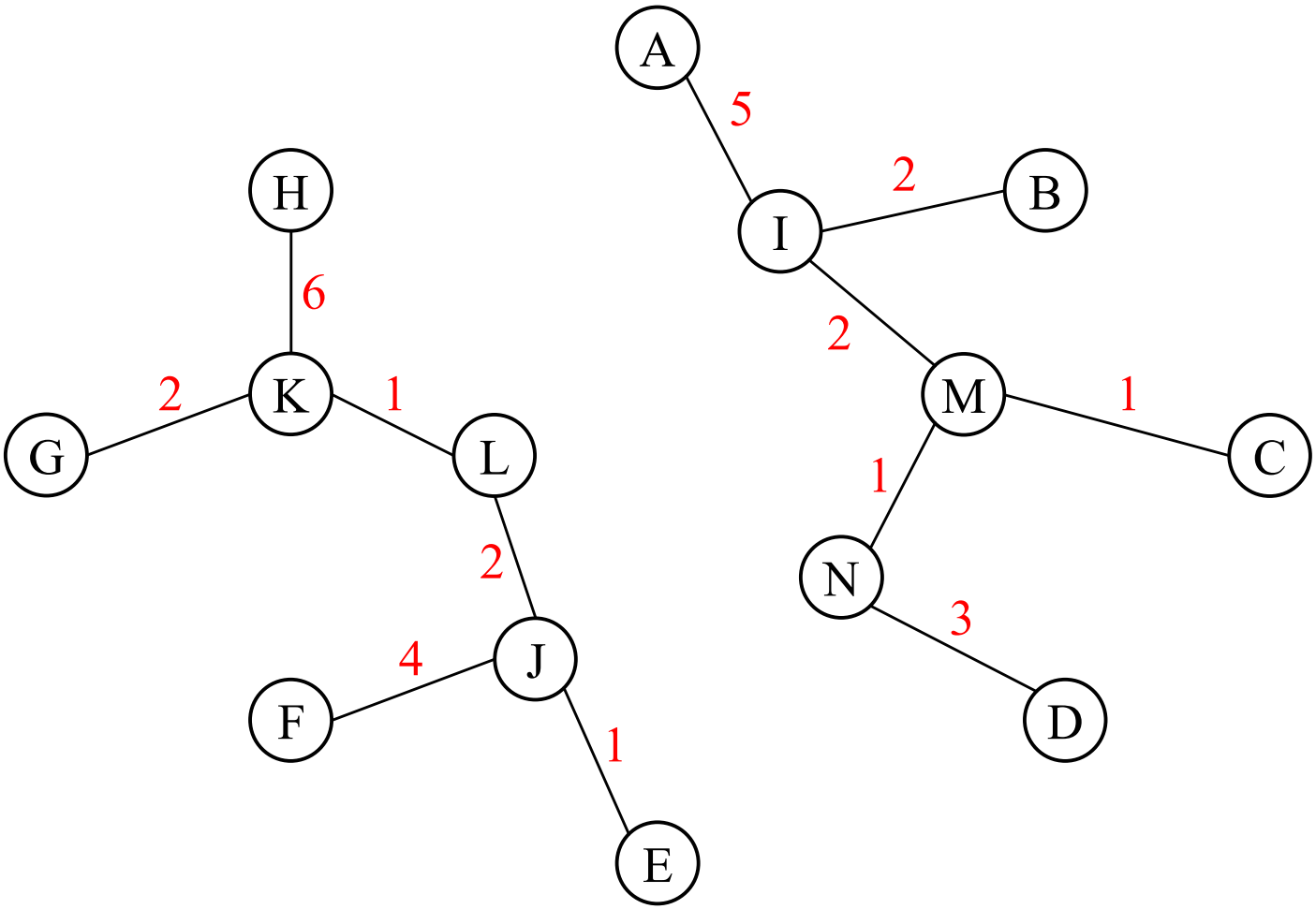
Neighbor-Joining

S	D	L	M	u
D		-12,0	-12,0	9,0
L	-12,0		-12,0	8,0
M	-12,0	-12,0		7,0
u	9,0	8,0	7,0	

Neighbor-Joining

d	D	L	M	u
D		5	4	9,0
L	5		3	8,0
M	4	3		7,0
u	9,0	8,0	7,0	
N	3	2	1	

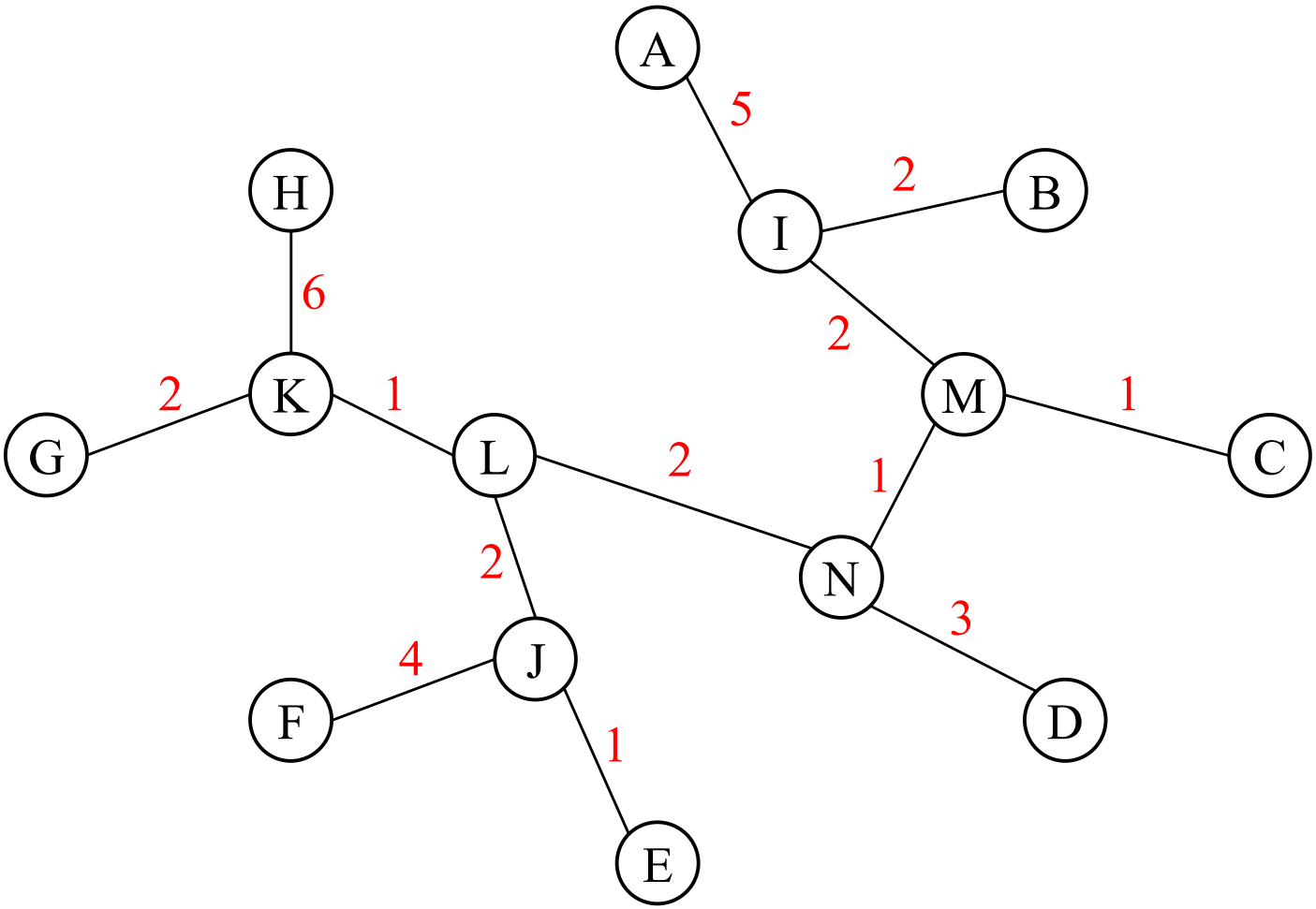
Neighbor-Joining



Neighbor-Joining

d	L	N
L		2
N	2	

Neighbor-Joining



Árvores Filogenéticas para Matrizes não Aditivas

- Em muitos casos práticos a matriz de distância não é aditiva.
- Nestes casos, estamos interessados em encontrar a melhor árvore filogenética em relação a matriz de distância.
- Existem muitas formas possíveis de definir "a melhor árvore filogenética" em relação a uma matriz de distância, por exemplo, a árvore que satizfaça a seguinte expressão:

$$min \sum_{i,j \in \mathcal{A}} (dist(i,j) - \delta(i,j))^2$$

- William Day, em 1987, provou que o problema de encontrar a melhor árvore filogenética, sob várias medidas diferentes, é um problema $\mathcal{NP}$-Completo.

Sanduíche Ultramétrico

Sanduíche Ultramétrico

Definição

Seja M^l e M^h duas matrizes de distâncias entre os objetos $\mathcal{A}$, contendo, respectivamente, limites inferiores e superiores para as distâncias entre os pares de objetos de $\mathcal{A}$. Ou seja, para todo par $i, j \in \mathcal{A}$, temos que:

$$M^l[i,j] \leq \delta(i,j) \leq M^h[i,j]$$

Sanduíche Ultramétrico

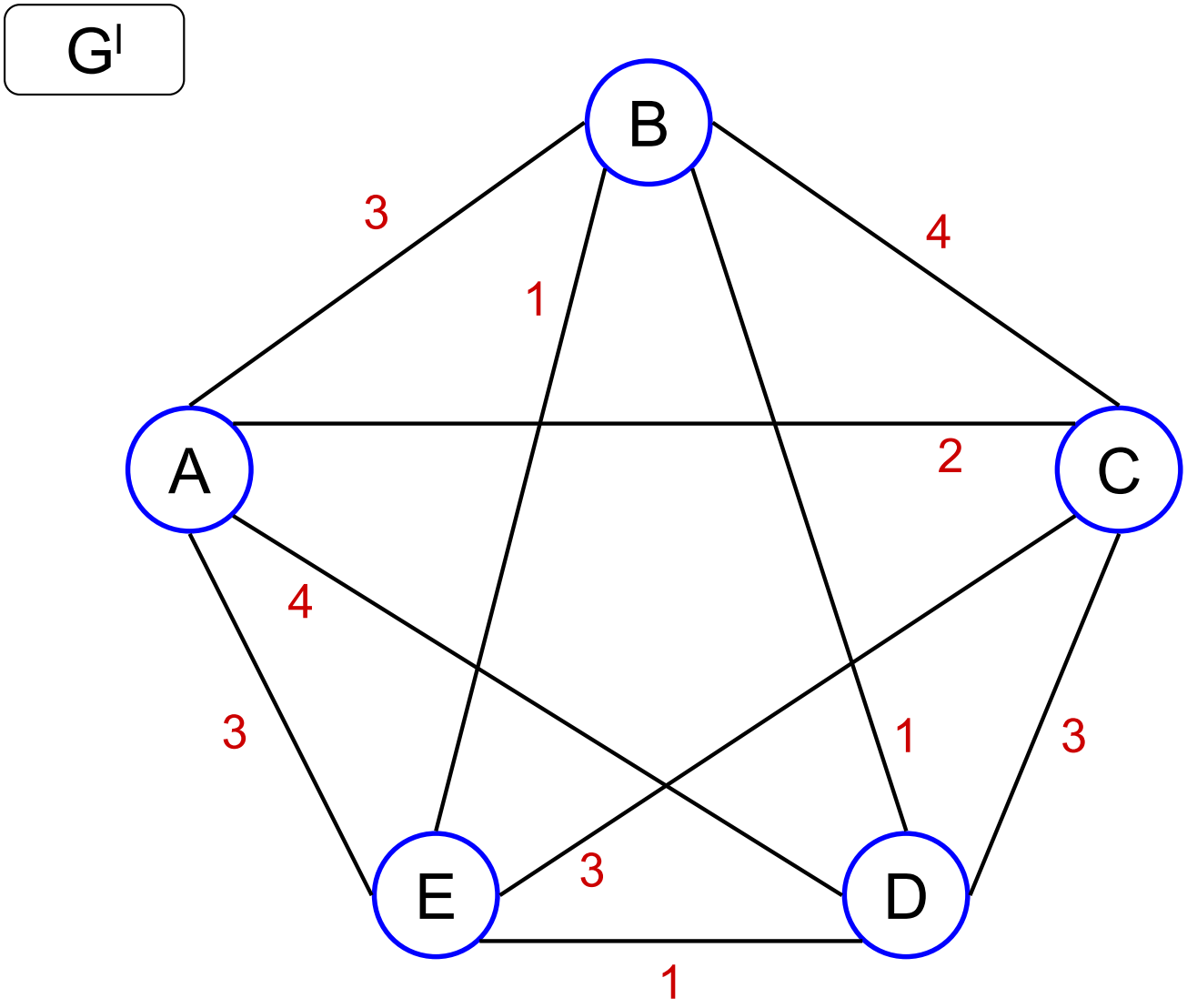
Definição

Seja T uma árvore geradora mínima para o grafo G^h construído a partir da matriz M^h. O corte mais pesado (cut-weight) de uma aresta e de T é dado por:

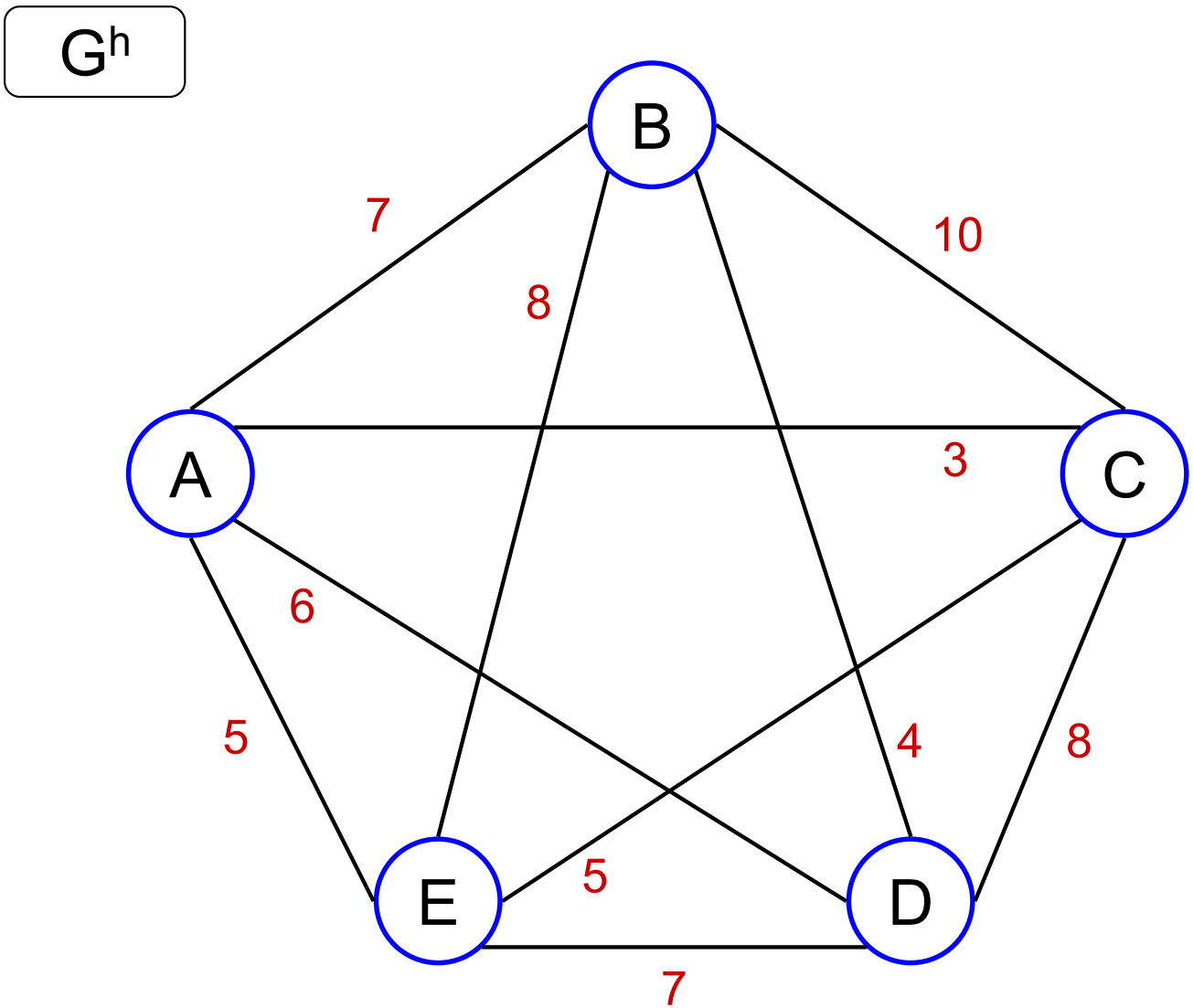
$$CW[e] = max\{M^l[a,b] | e = (a,b)_{max}\}$$

onde $(a,b)_{max}$ é a aresta mais pesada do caminho entre os vértices a e b na árvore geradora mínima T.

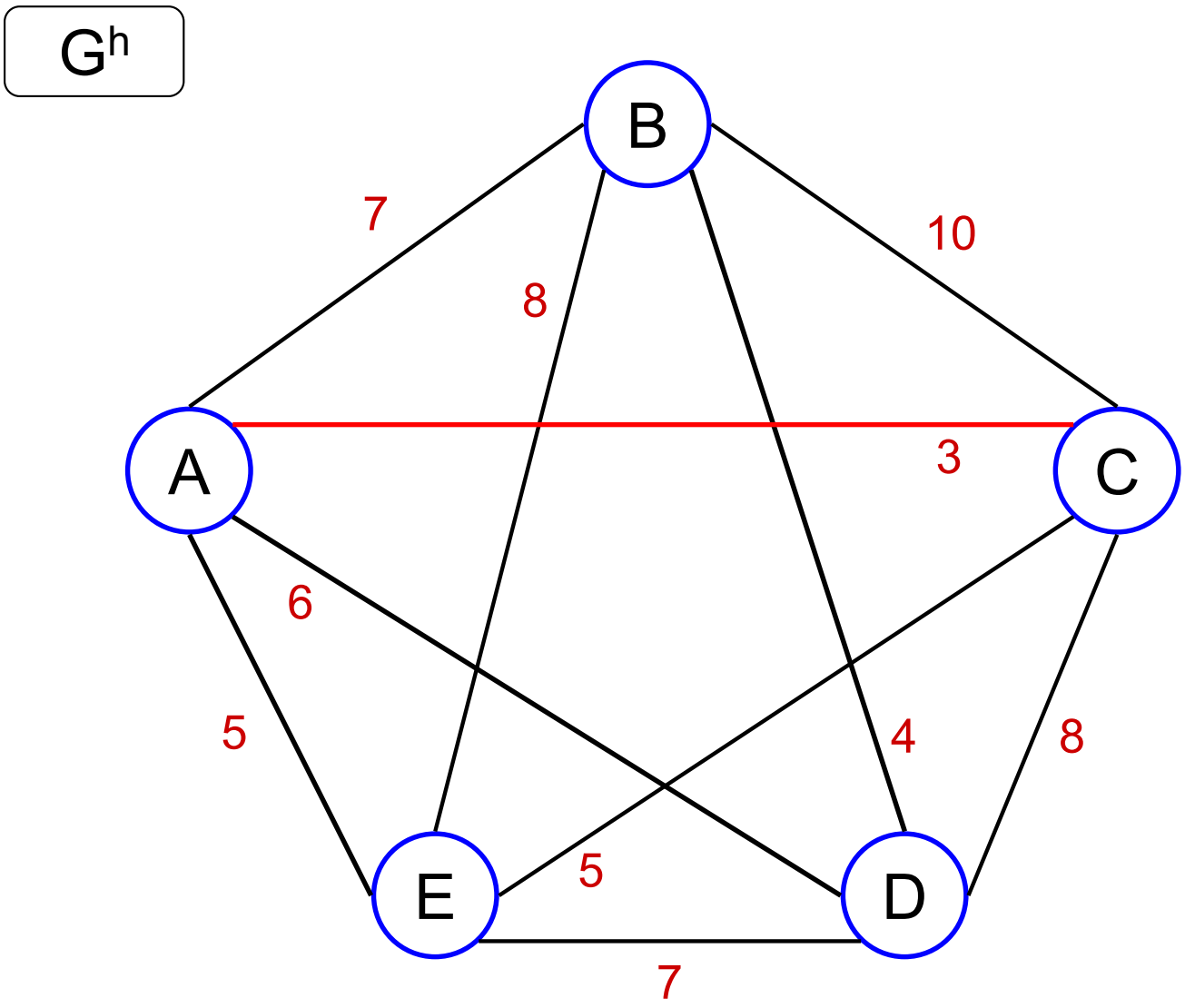
Sanduíche Ultramétrico



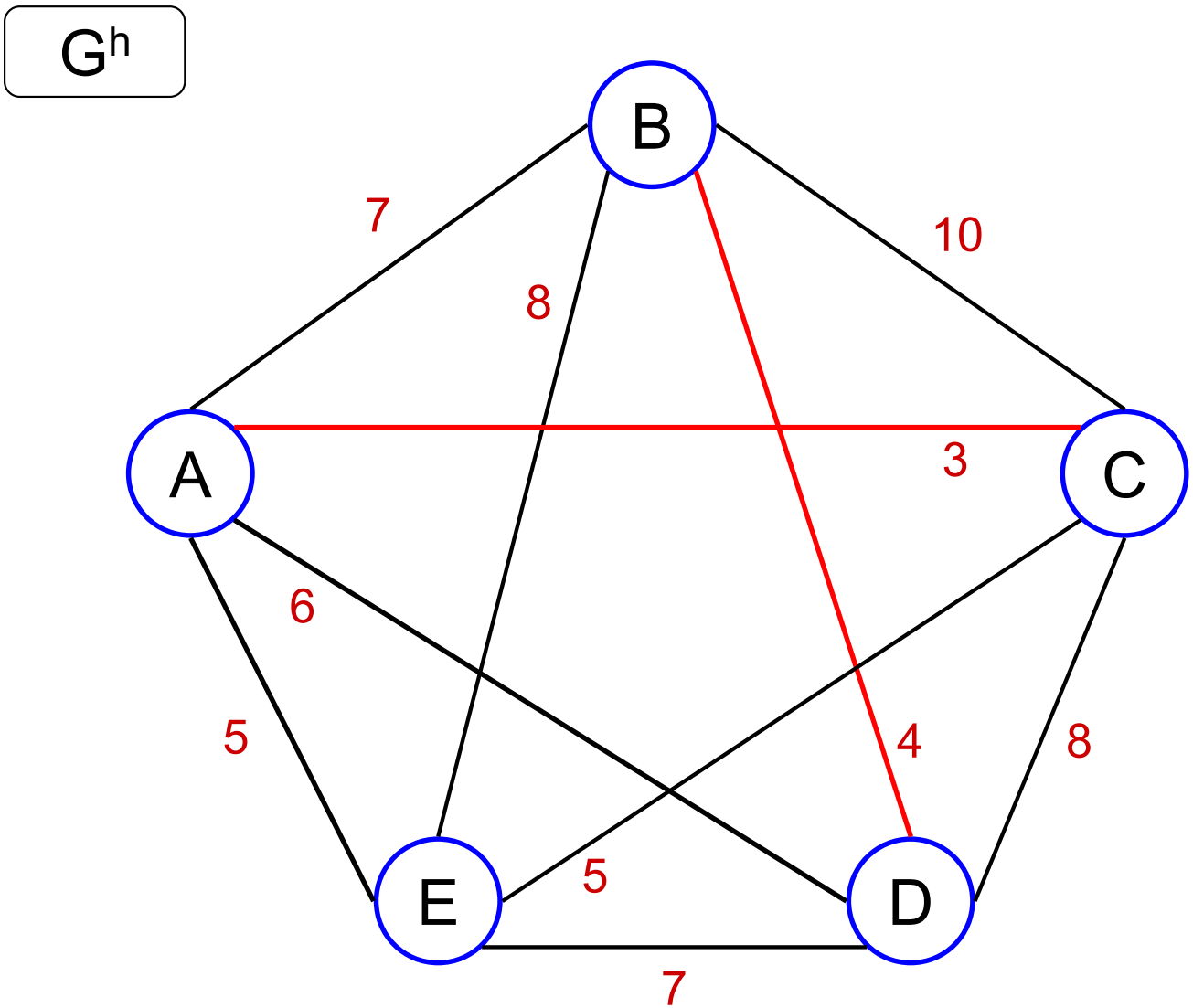
Sanduíche Ultramétrico



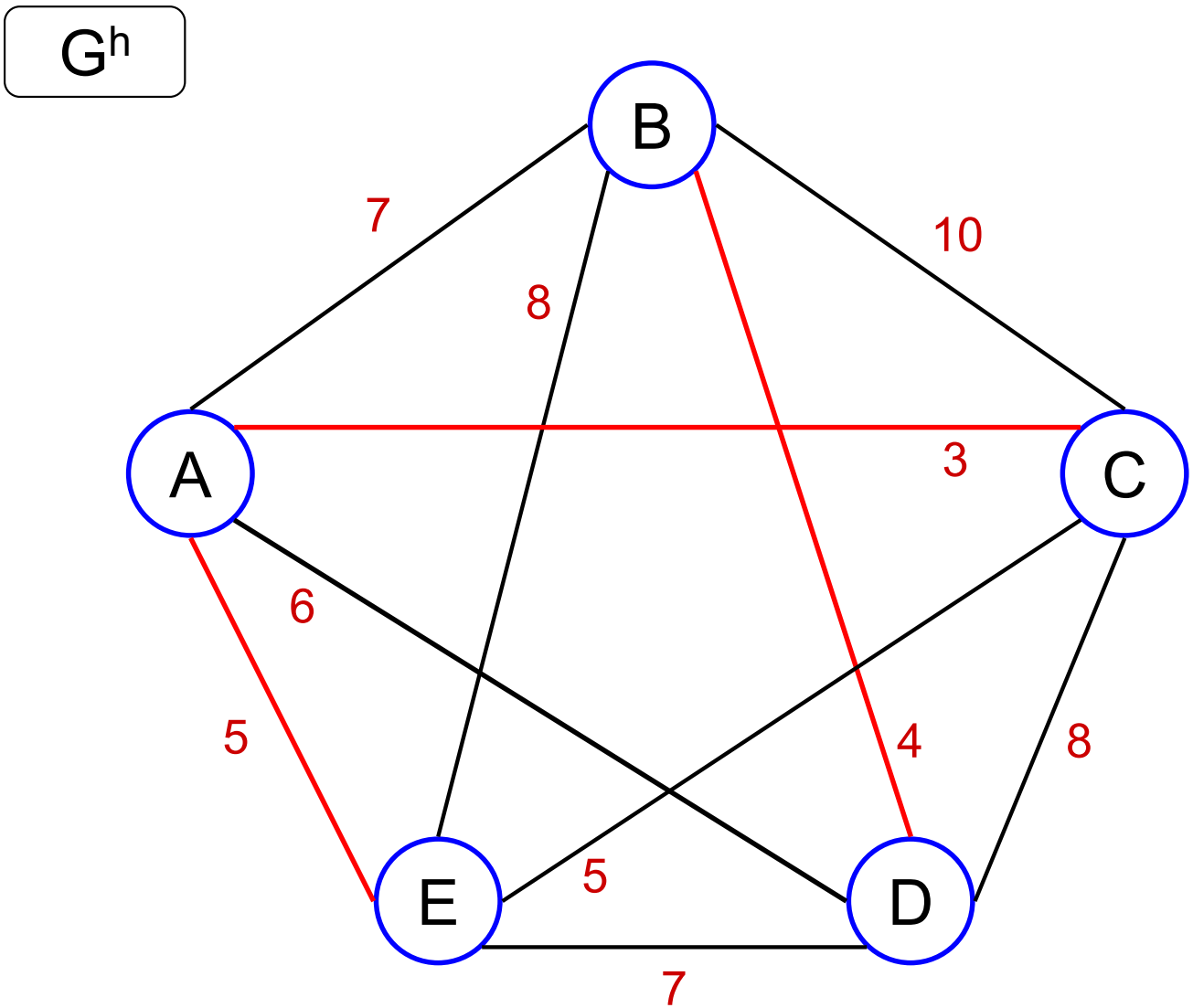
Sanduíche Ultramétrico



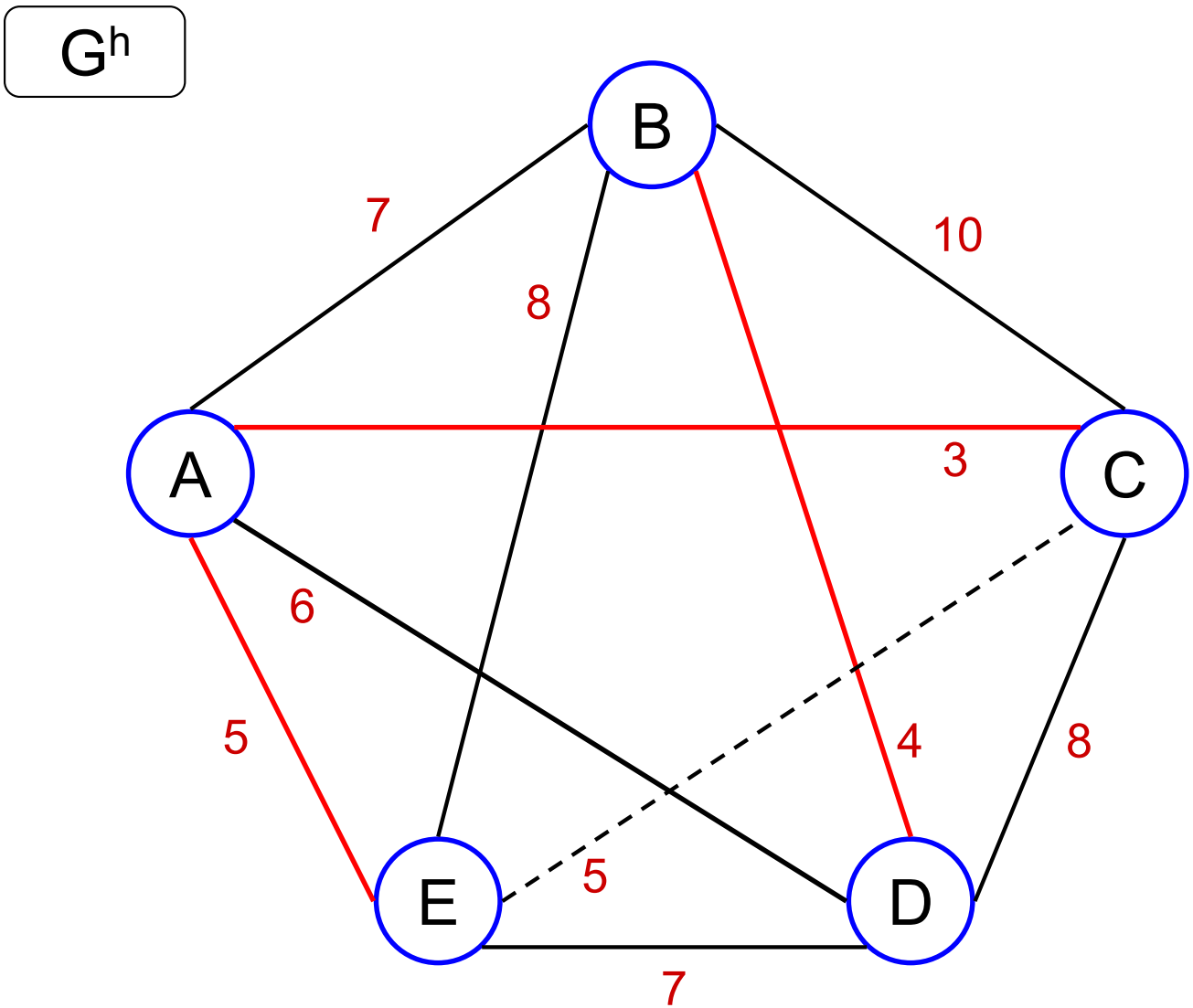
Sanduíche Ultramétrico



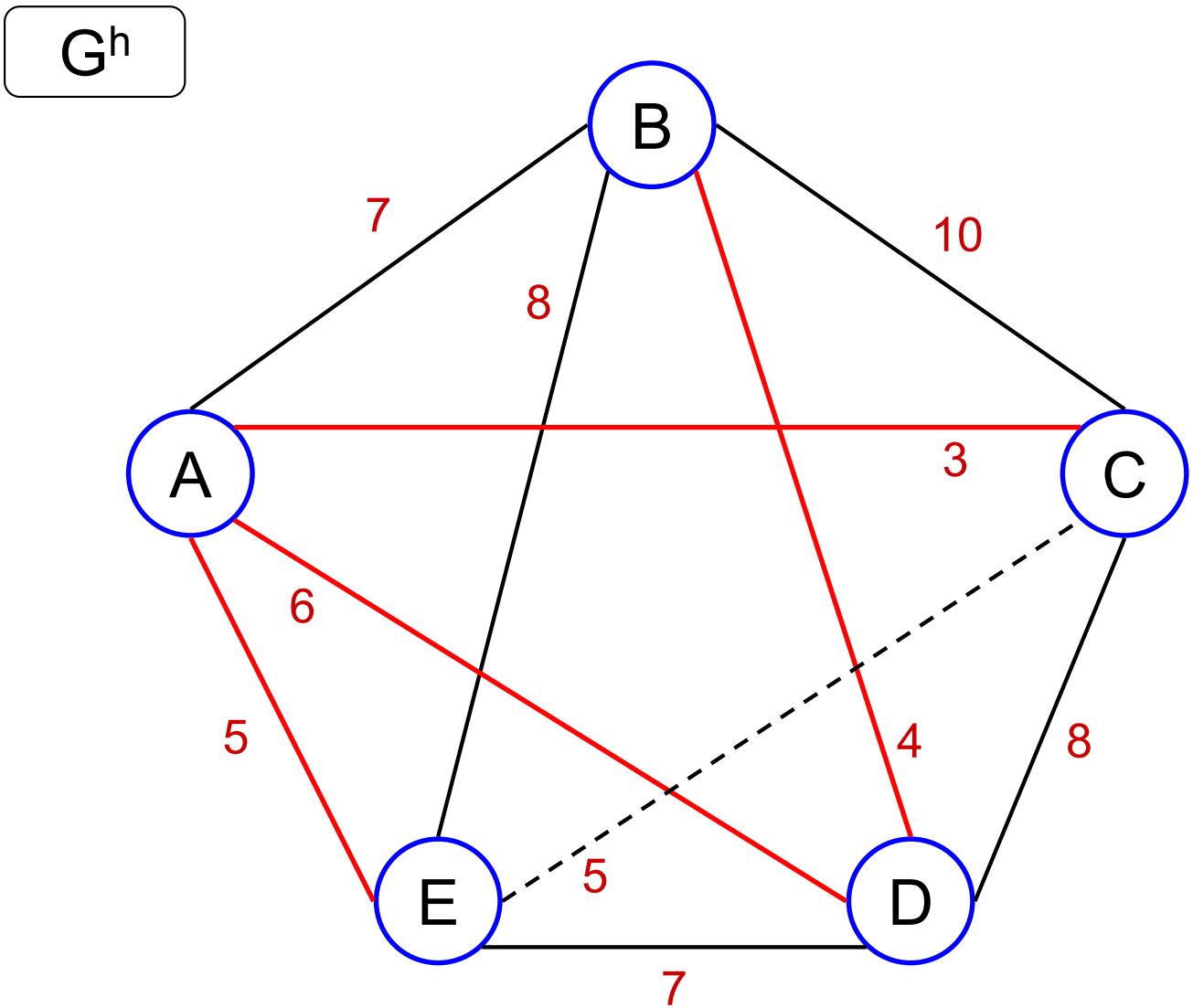
Sanduíche Ultramétrico



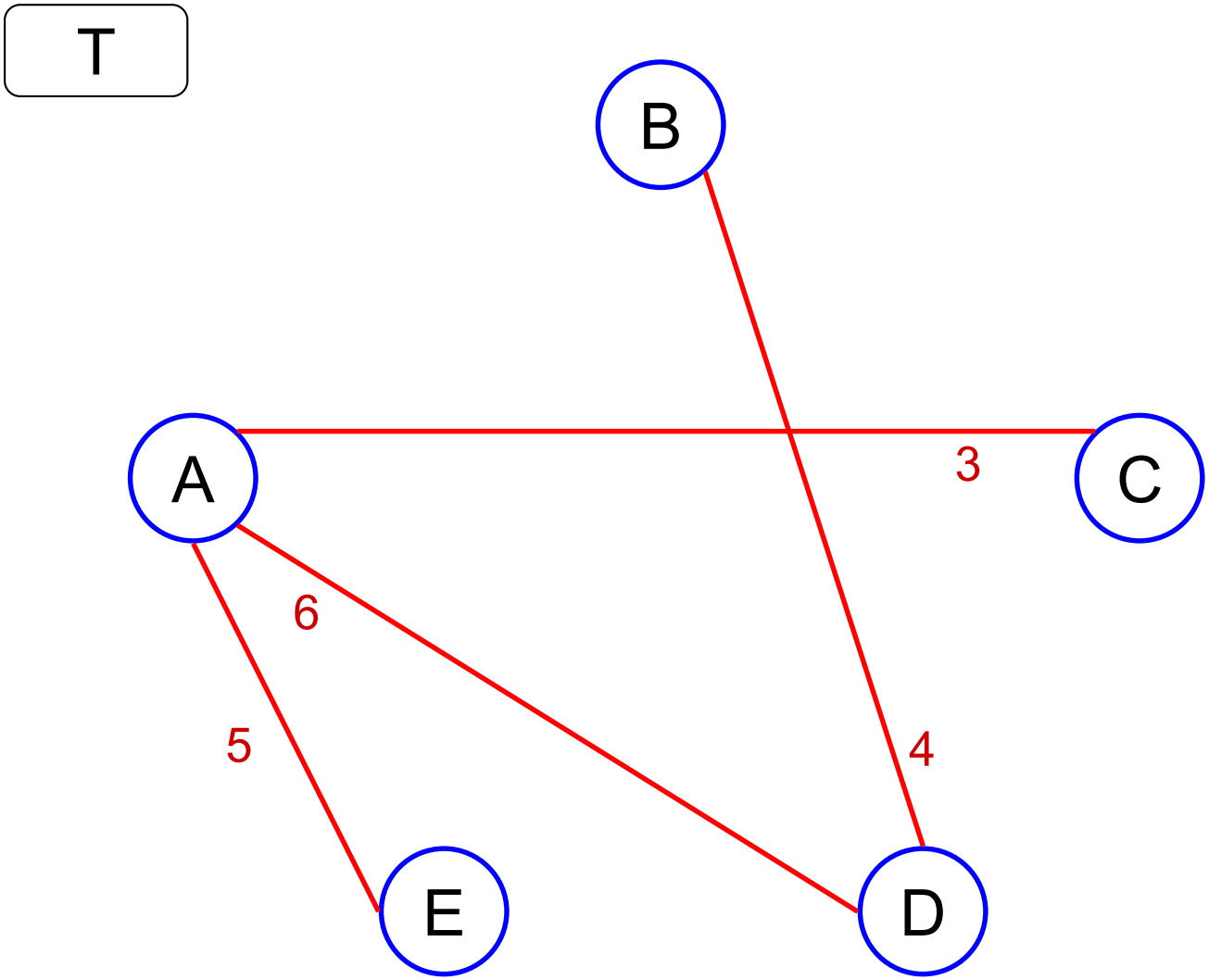
Sanduíche Ultramétrico



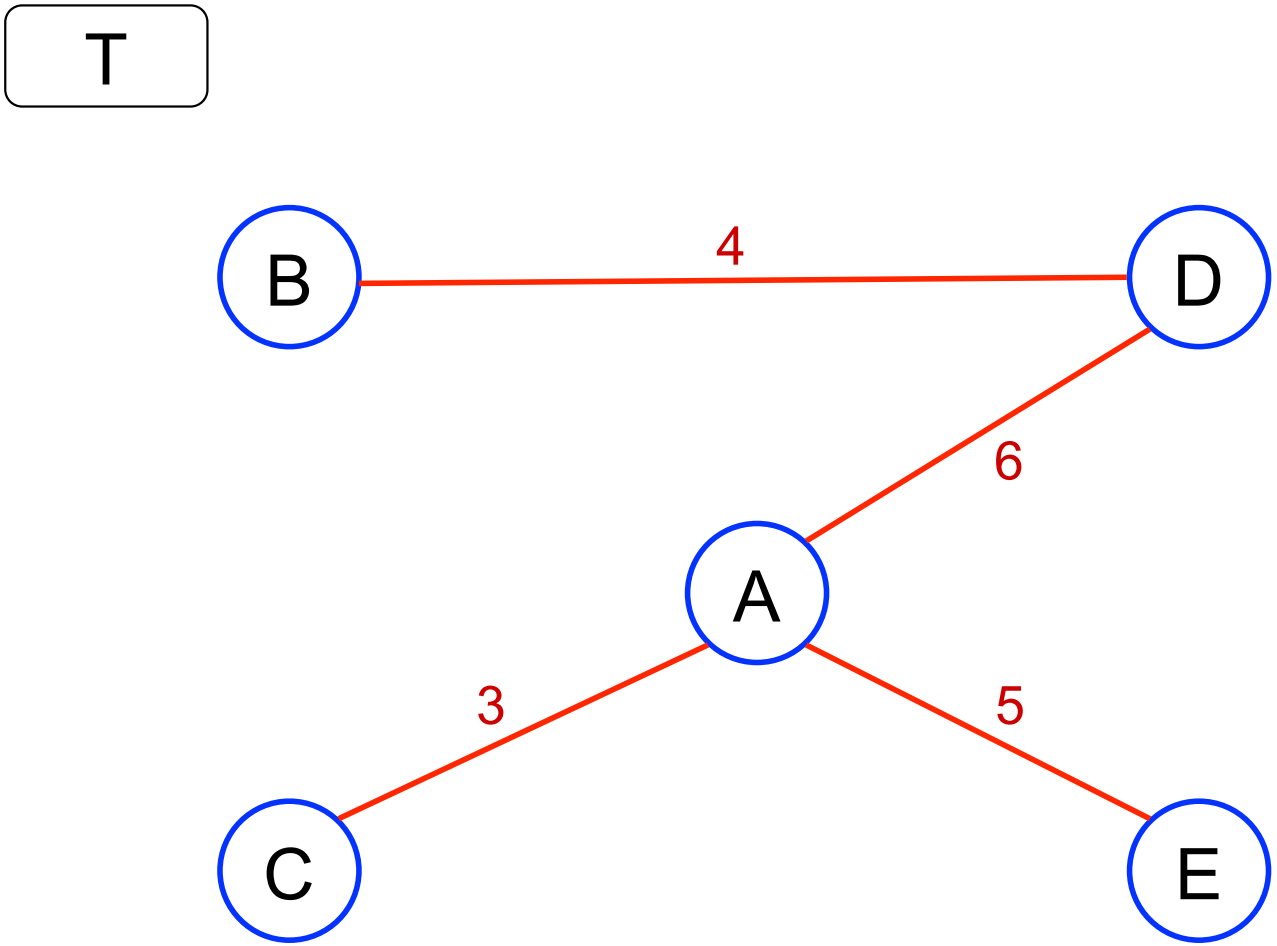
Sanduíche Ultramétrico



Sanduíche Ultramétrico



Sanduíche Ultramétrico

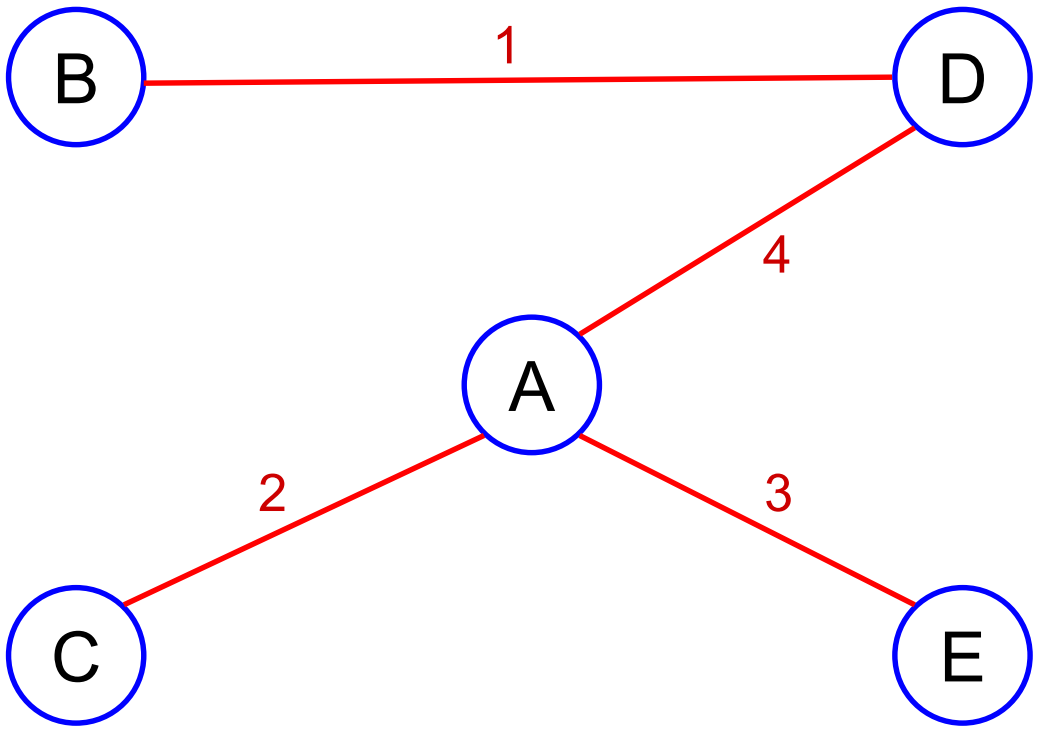


Sanduíche Ultramétrico

- $CW[(B,D)] = max\{M^l[a,b] | (a,b)_{max} = (B,D)\}$
  $= max\{M^l[a,b] | (a,b) = \{(B,D)\}\} = M^l[B,D] = 1$
- $CW[(A,D)] = max\{M^l[a,b] | (a,b)_{max} = (A,D)\}$
  $= max\{M^l[a,b] | (a,b) = \{(A,B), (B,C), (B,E), (A,D), (C,D),$
  $(D,E)\}\} = max\{3,4,1,4,3,1\} = 4$
- $CW[(A,C)] = max\{M^l[a,b] | (a,b)_{max} = (A,C)\}$
  $= max\{M^l[a,b] | (a,b) = \{(A,C)\}\} = M^l[A,C] = 2$
- $CW[(A,E)] = max\{M^l[a,b] | (a,b)_{max} = (A,E)\}$
  $= max\{M^l[a,b] | (a,b) = \{(A,E),(C,E)\}\} = max\{3,3\} = 3$

Sanduíche Ultramétrico

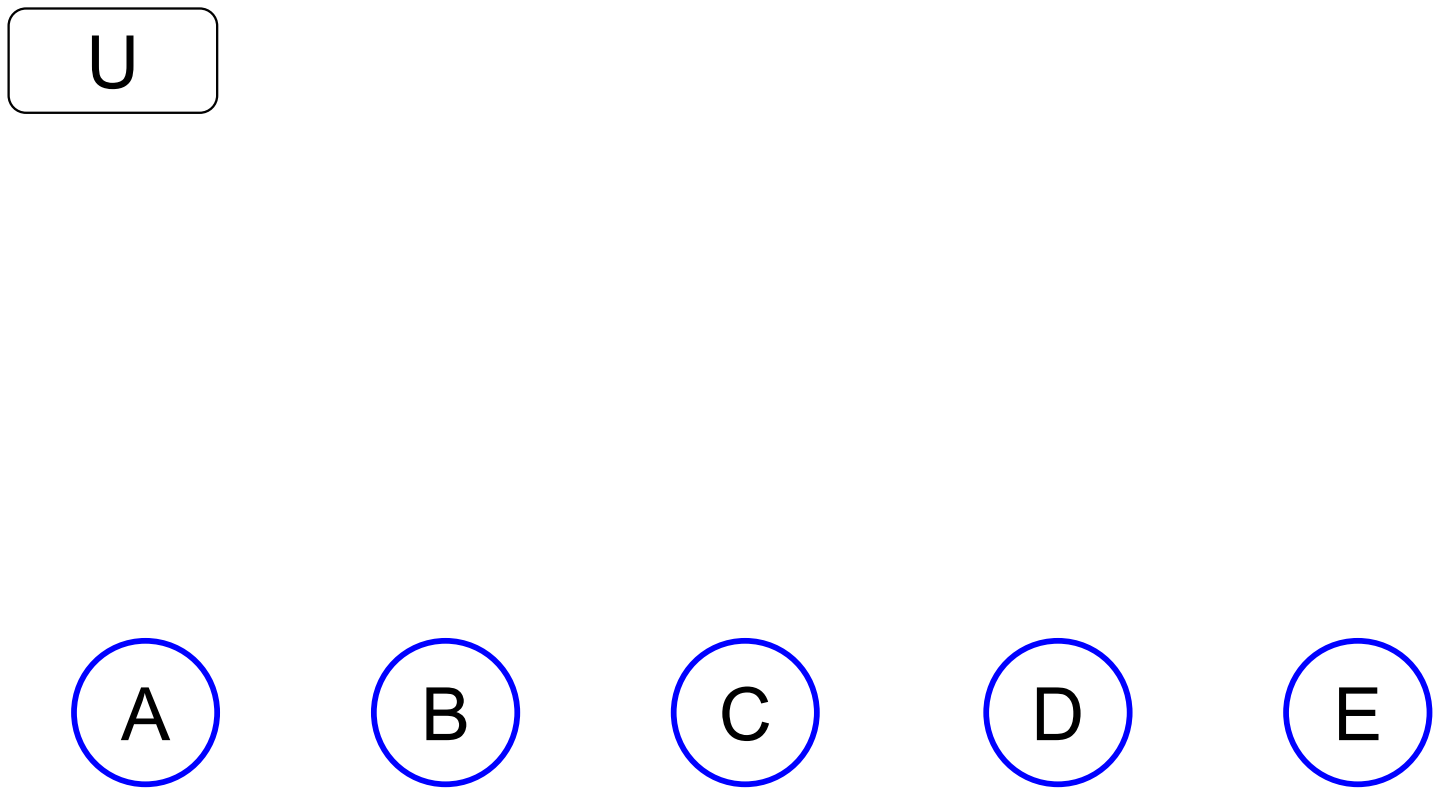
CW



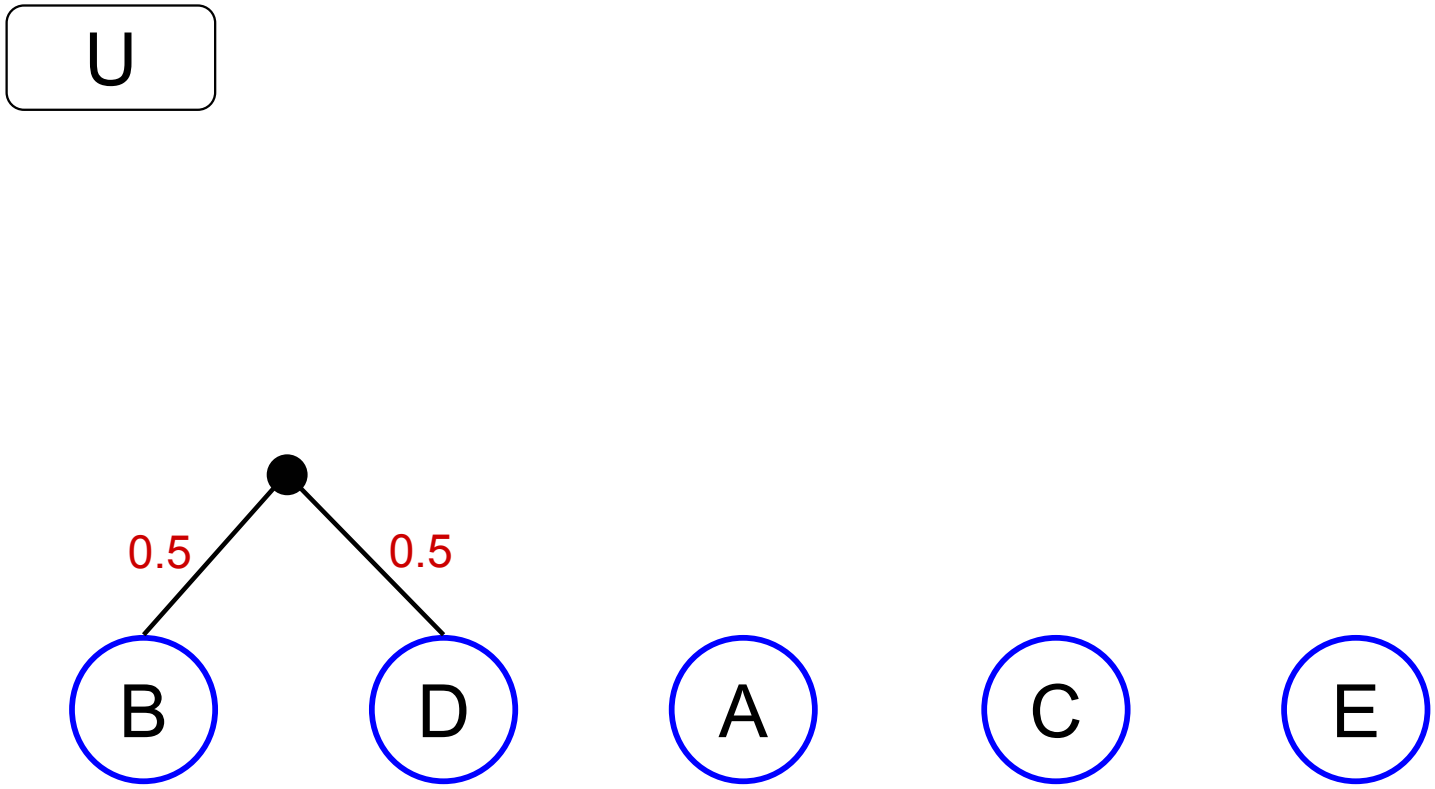
Algoritmo 3: Ultrametric Sandwich

Input: $\mathcal{A}$,T,CW
for *all* $i \in \mathcal{A}$ **do**
    MakeSet(i)
    CreateNode(i)
    $height[i] \leftarrow 0$
end
Sort edges of T in nondecreasing order of cut-weights (CW)
for *all edge* $e = (a, b) \in T$ *in that order* **do**
    $A \leftarrow$ FindSet(a); $B \leftarrow$ FindSet(b)
    if $A \neq B$ **then**
        $u_a \leftarrow$ root of the tree that contains a
        $u_b \leftarrow$ root of the tree that contains b
        CreateNode(U)
        $U.left \leftarrow u_a$
        $U.right \leftarrow u_b$
        $height[U] \leftarrow CW[e]/2$
        $d(U, u_a) \leftarrow height(U) - height(u_a)$
        $d(U, u_b) \leftarrow height(U) - height(u_b)$
        Union(A,B)
    end
end
return U

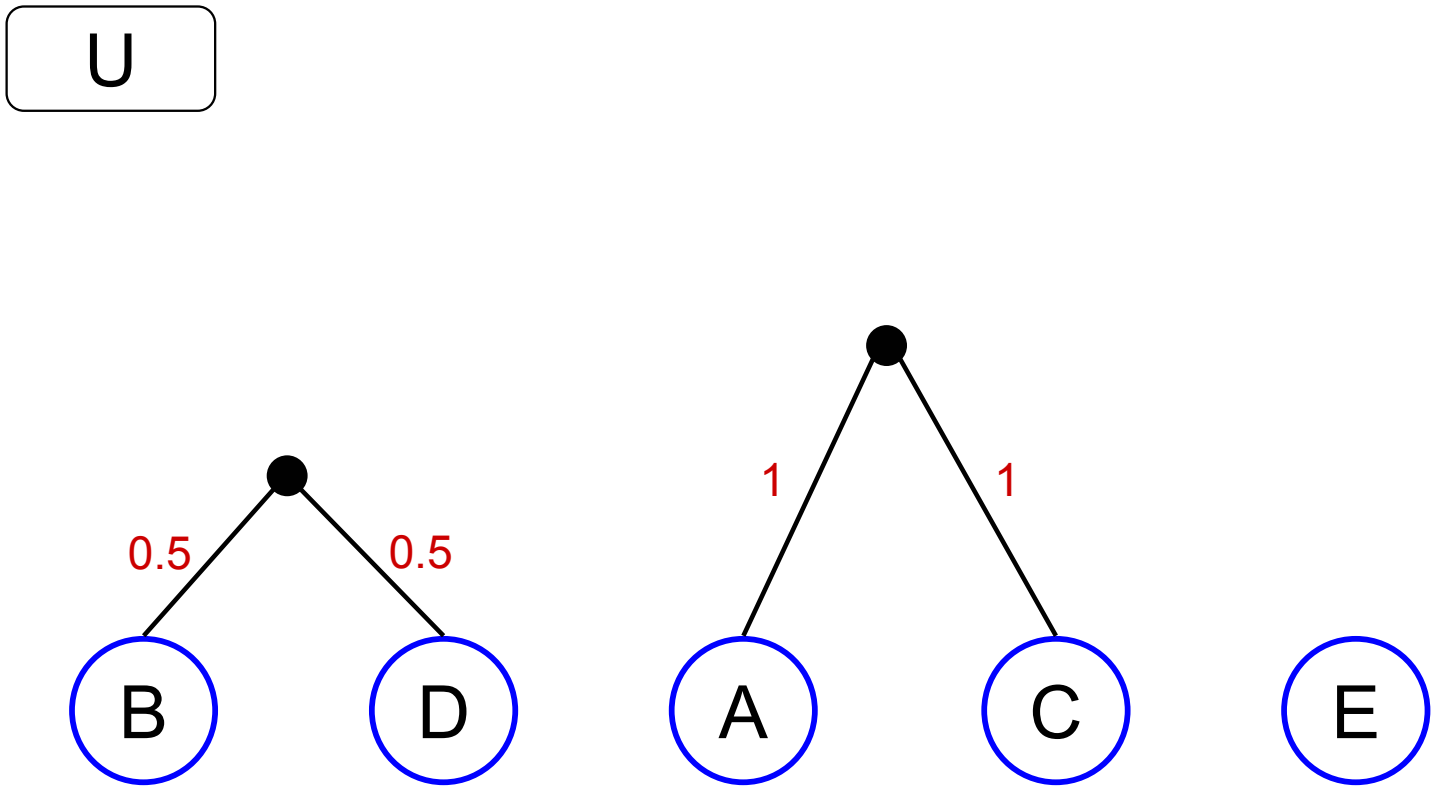
Sanduíche Ultramétrico



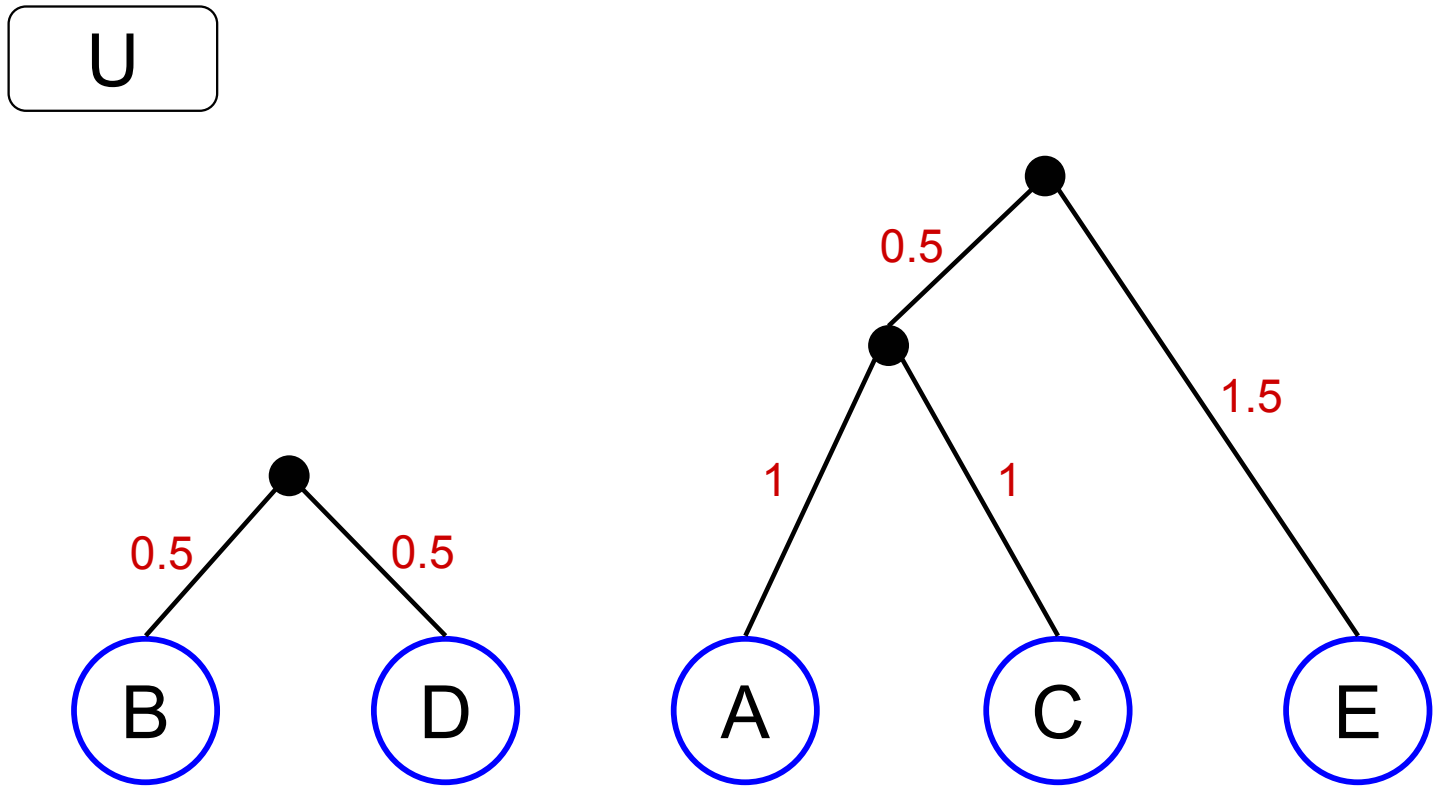
Sanduíche Ultramétrico



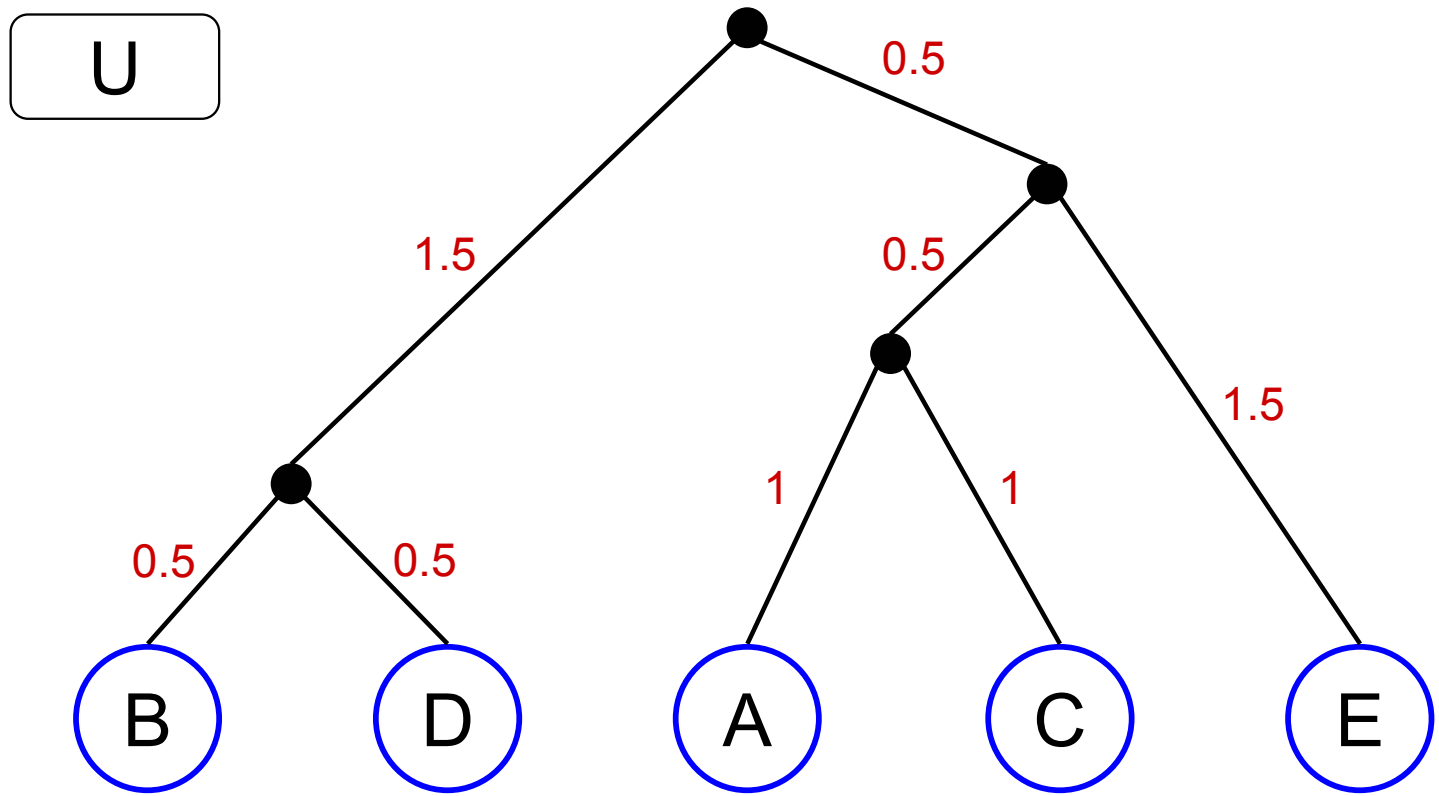
Sanduíche Ultramétrico



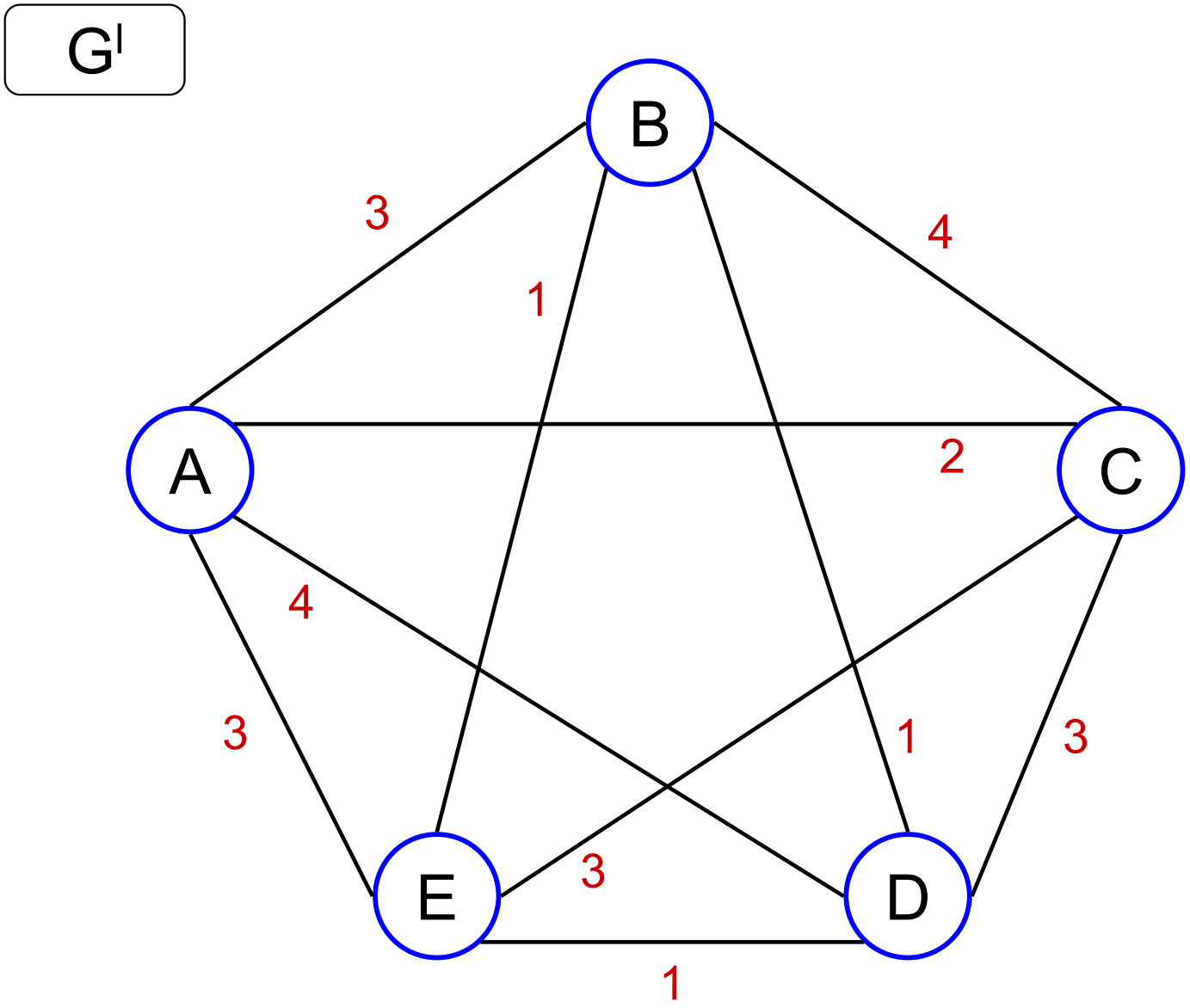
Sanduíche Ultramétrico



Sanduíche Ultramétrico

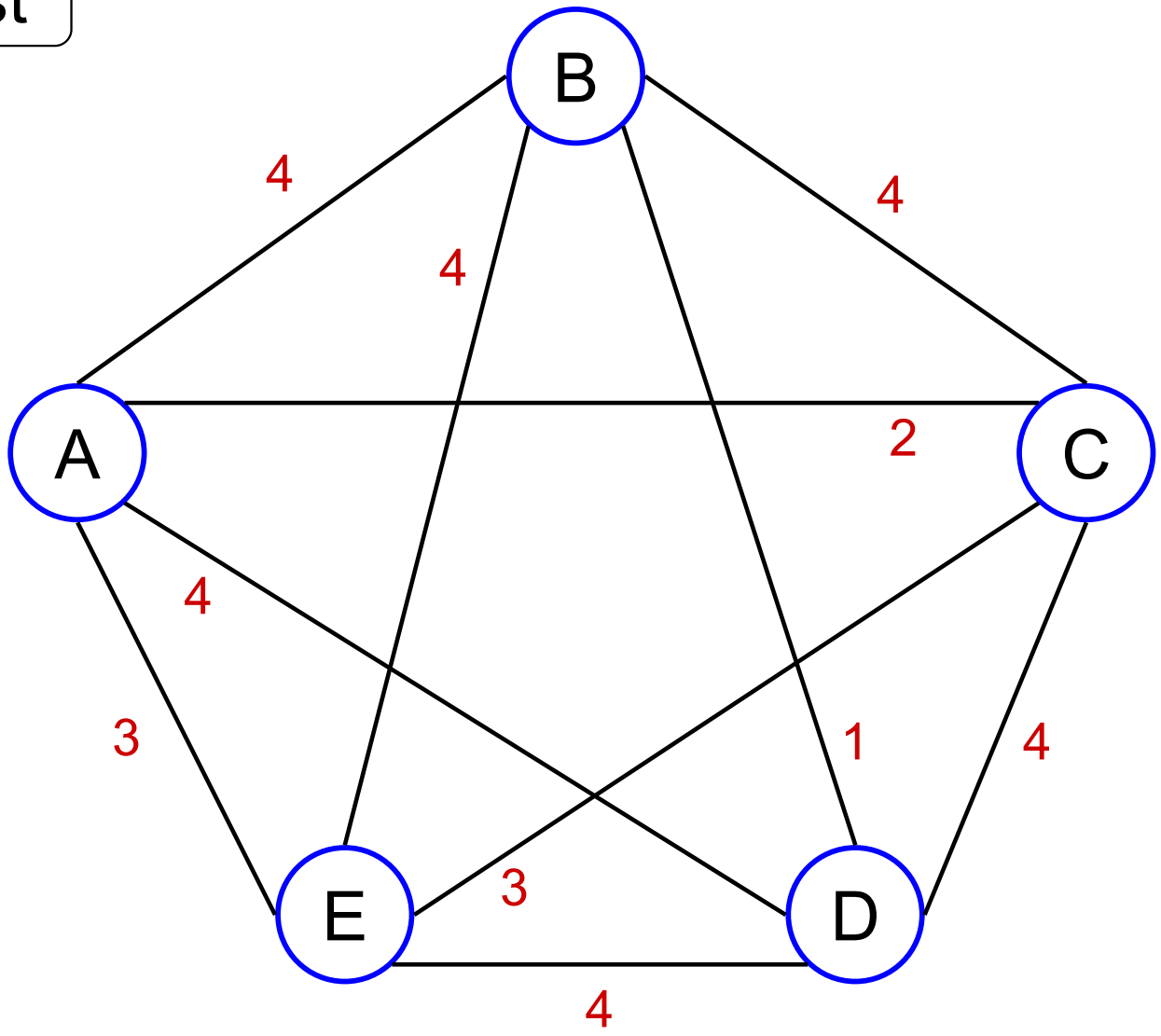


Sanduíche Ultramétrico

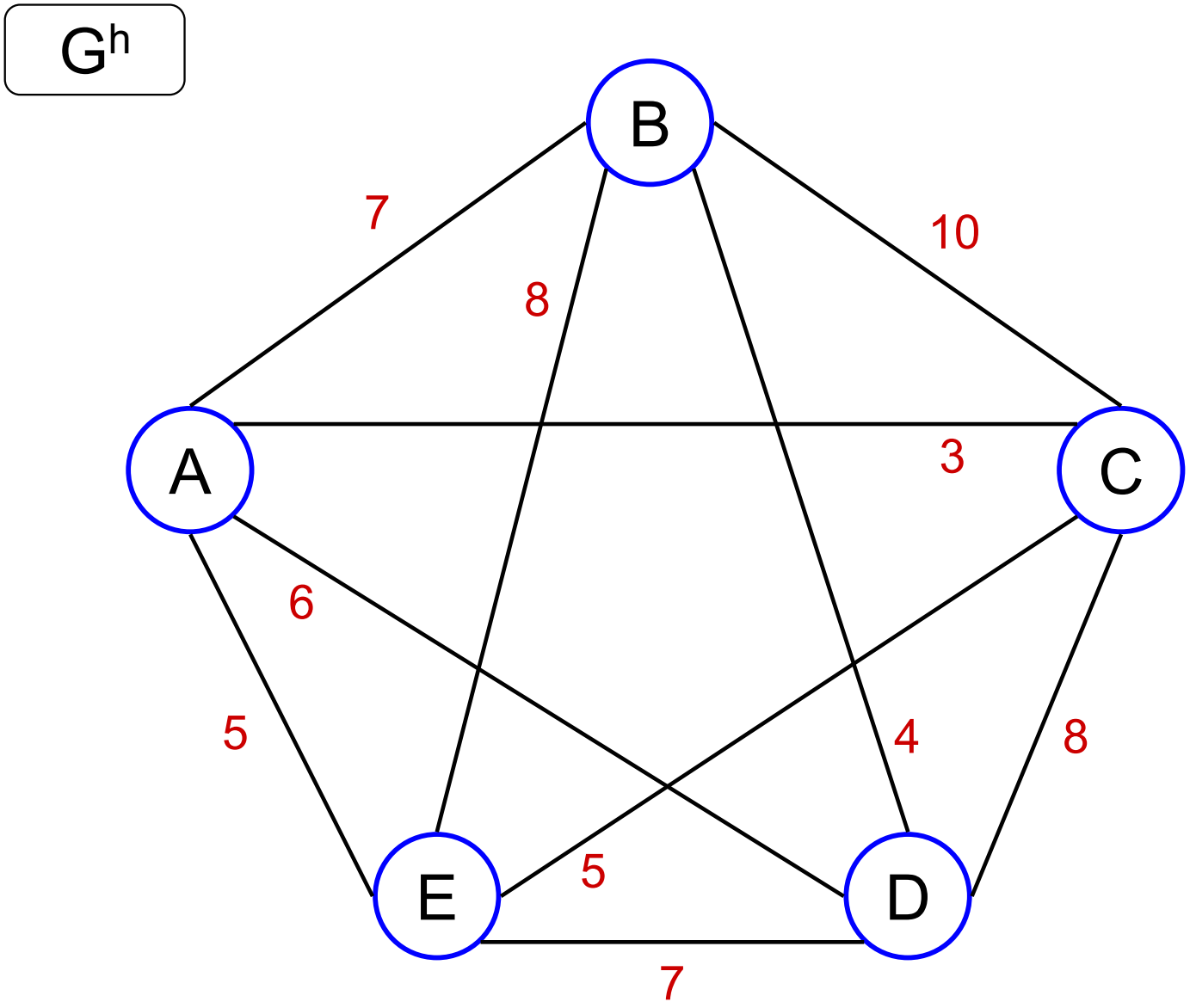


Sanduíche Ultramétrico

dist



Sanduíche Ultramétrico



Complexidade

- Construção da Árvore Geradora Mínima T de G^h:
  - $\Theta(n^2)$, usando o algoritmo de Prim.
- Cálculo de $CW[e]$, para toda aresta e de T:
  - Método ingênuo: para cada aresta e de T, calcule os caminhos mínimos entre todo par u e v de vértices de T e verifique se a aresta de peso máximo do caminho é e:
    $\Theta(n) \times \Theta(n^2) \times \Theta(n) = \Theta(n^4)$.
  - Método ótimo: inicialize $CW[e] = 0$, para toda aresta e de T. Para cada vértice u de T descubra a aresta $e = (u, v)_{max}$ de peso máximo no caminho de u para cada um dos demais vértice de v e atualize o valor de $CW[e]$: $\Theta(n) \times \Theta(n) = \Theta(n^2)$.
- Algoritmo `Ultrametric Sandwich`:
  - $\Theta(n \log n + n\alpha(n)) = \Theta(n \log n)$.
- Total: $\Theta(n^2)$.

Sanduíche Ultramétrico

- Se existe uma árvore ultramétrica tal que $M^l[i,j] \leq d(i,j) \leq M^h[i,j]$, para todo par de elementos $i, j \in \mathcal{A}$, onde $d(i,j)$ é a distância entre i e j na árvore, então o algoritmo visto anteriormente constrói uma árvore com esta propriedade.
- Note que podem existir várias árvores ultramétricas com tal característica.
- Algoritmo proposto por Martin Farach, Sampath Kannan e Tandy Warnow, em 1993.
- Estes mesmo pesquisadores também provaram que o problema de obter uma árvore aditiva (não necessariamente ultramétrica) que satizfaça as restrições de "sanduíche" entre duas matrizes de distância é um problema $\mathcal{NP}$-Completo.

Características com Estados Discretos

Características com Estados Discretos

- Dependendo da quantidade de informação que possuímos sobre os estados, podemos classificá-los como ordenados ou não ordenados.
  - Estados Não-Ordenados: não sabemos nada sobre como as características podem mudar de um estado para outro.
  - Estados Ordenados: quando sabemos exatamente quais são as trocas de estados possíveis para cada característica. Exemplos:
    - Ordenação linear: $3 \leftrightarrow 1 \leftrightarrow 4 \leftrightarrow 2$
    - Parcialmente ordenado: $3 \leftrightarrow 1$, $3 \leftrightarrow 5$, $5 \leftrightarrow 2$, $5 \leftrightarrow 4$
- Características onde conhecemos a direção das mudanças de estados são chamadas de orientadas (ou polares).
  - Característica Não Orientada: $3 \leftrightarrow 1 \leftrightarrow 4 \leftrightarrow 2$
  - Característica Orientada: $3 \rightarrow 1 \rightarrow 4 \rightarrow 2$

Características com Estados Binários

- Todas as características só tem dois estados possíveis:
  - 0: ausente.
  - 1: presente.
- Todas as características são independentes entre si.
- Não há nenhuma característica ausente ou presente em todos os objetos.
- Não existem dois ou mais objetos com todas as características no mesmo estado.
- Todas as características evoluem do estado 0 para o estado 1. Após alcançar o estado 1, uma característica nunca retorna ao estado 0.
- Ou seja, os estado são ordenados e orientados.
- A raíz da árvore filogenética representará o ancestral com todas as características ausentes (estado 0 para todas as características).

Filogenia Perfeita para Características com Estados Binários

Definição

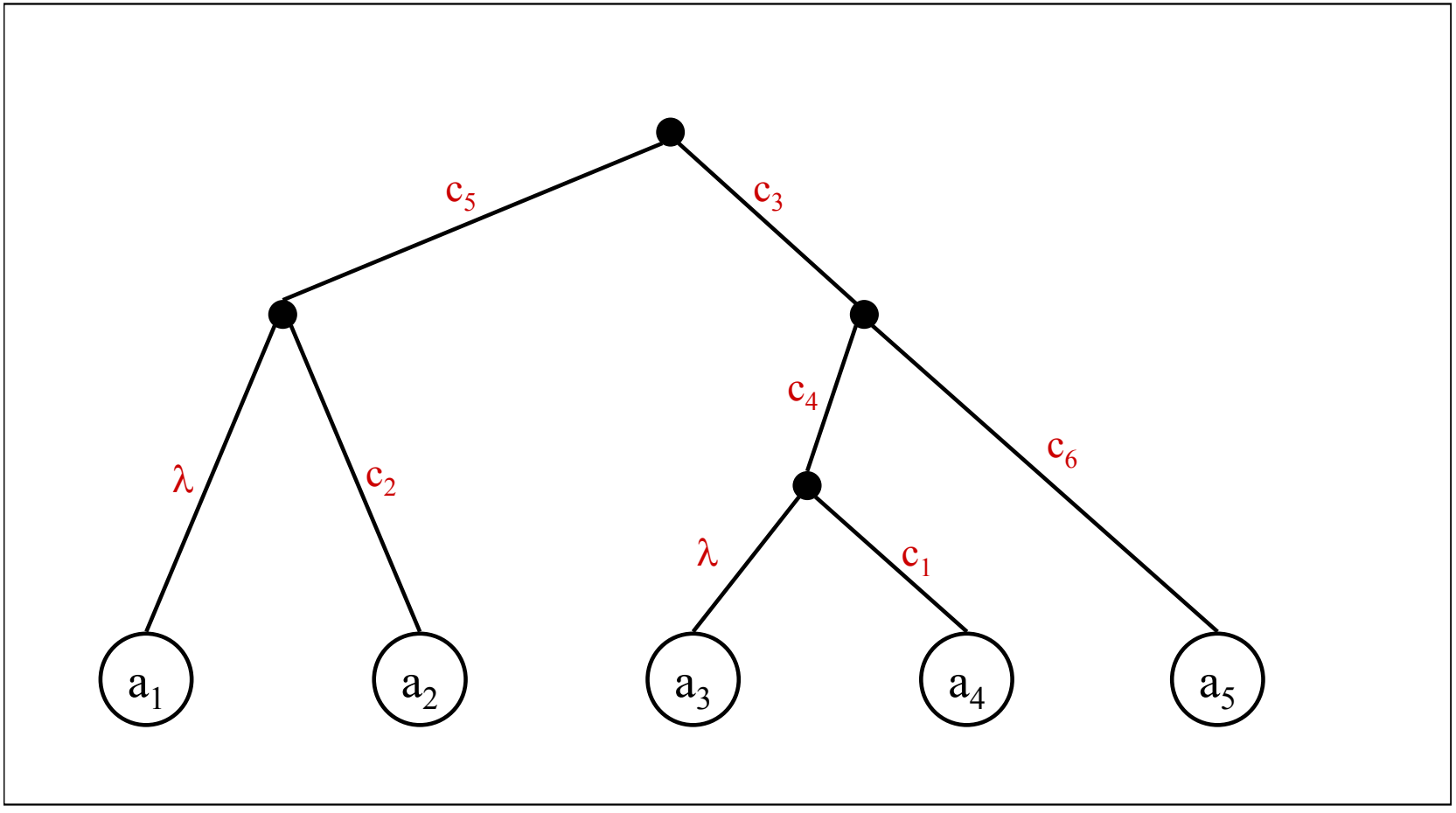
Seja $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ um conjunto de objetos, $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$ um conjunto de características binárias e M uma Matriz de Estados de Caracterísicas para $\mathcal{A}$ e $\mathcal{C}$. Uma árvore filogenética perfeita para M é uma árvore $T = (V, E, d)$ com exatamente n folhas satizfazendo as seguintes condições:

- *As folhas de T correspondem aos objetos de $\mathcal{A}$.*
- *As arestas são rotuladas de acordo com a função $d : E \to \mathcal{C} \cup \{\lambda\}$, onde λ representa o rótulo vazio.*
- *Cada uma das características de $\mathcal{C}$ é atribuída a exatamente uma aresta de T.*
- *Para cada objeto a_i, o conjunto de rótulos do caminho de a_i em T até a raiz corresponde exatamente as características presentes em a_i.*

Filogenia Perfeita para Características com Estados Binários

	c_1	c_2	c_3	c_4	c_5	c_6
a_1	0	0	0	0	1	0
a_2	0	1	0	0	1	0
a_3	0	0	1	1	0	0
a_4	1	0	1	1	0	0
a_5	0	0	1	0	0	1

Filogenia Perfeita para Características com Estados Binários



Existência de Filogenia Perfeita

Definição

Para cada característica c_i de $\mathcal{C}$ seja $\mathcal{A}_i$ o conjunto de objetos de $\mathcal{A}$ tal que o estado da característica c_i seja igual a 1.

- Exemplos: $\mathcal{A}_3 = \{a_3, a_4, a_5\}$, $\mathcal{A}_5 = \{a_1, a_2\}$.

Lema

Uma matriz binária M admite uma filogenia perfeita se e somente se para cada par de características c_i e c_j os conjuntos $\mathcal{A}_i$ e $\mathcal{A}_j$ ou são disjuntos ou um deles contém o outro.

- Complexidade:
  - Testar se dois conjuntos são compatíveis: $O(n)$.
  - Número de pares de conjuntos a serem testados: $O(m^2)$.
  - Total: $O(nm^2)$.

Existência de Filogenia Perfeita

- É possível obter um algoritmo mais eficiente.
- Ideia:
  - Ordenar as colunas da matriz M de características binárias pelo número de 1s (as colunas que possuem a maior quantidade de números 1s devem ficar a esquerda).
  - Construir uma matriz auxiliar L para indicar, para cada valor 1, a posição mais próxima de um valor 1 a sua esquerda na matriz ordenada (na mesma linha).
  - Usar a matriz L para deduzir se existem dois conjuntos A_i e A_j que são incompatíveis entre si (ou seja, não são disjuntos, e nenhum deles contém o outro).

Existência de Filogenia Perfeita

Algoritmo 4: Existence of a Perfect Phylogenetic Tree

Input: $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}, \mathcal{C} = \{c_1, c_2, \ldots, c_m\}, M$
Sort the columns of M in nonincreasing order of numbers of ones
for *all* $i \in [1..n], j \in [1..m]$ **do** $L[i, j] \leftarrow 0$;
for *all* $i \in [1..n]$ **do**
    $k \leftarrow -1$
    for *all* $j \in [1..m]$ **do**
        if $M[i, j] = 1$ **then**
            $L[i, j] \leftarrow k$
            $k \leftarrow j$
        end
    end
end
for *all* $j \in [1..m]$ **do**
    $l \leftarrow 0$
    for *all* $i \in [1..n]$ **do**
        if $L[i, j] \neq 0$ **then**
            if $l = 0$ **then** $l \leftarrow L[i, j]$;
            else if $L[i, j] \neq l$ **then return** `false` ;
        end
    end
end
return `true`

Existência de Filogenia Perfeita

M	c_1	c_2	c_3	c_4	c_5	c_6
a_1	0	0	0	0	1	0
a_2	0	1	0	0	1	0
a_3	0	0	1	1	0	1
a_4	1	0	1	1	0	0
a_5	0	0	1	0	0	1

Existência de Filogenia Perfeita

M	c_3	c_4	c_5	c_6	c_1	c_2
a_1	0	0	1	0	0	0
a_2	0	0	1	0	0	1
a_3	1	1	0	1	0	0
a_4	1	1	0	0	1	0
a_5	1	0	0	1	0	0

Existência de Filogenia Perfeita

L	c_3	c_4	c_5	c_6	c_1	c_2
a_1	0	0	-1	0	0	0
a_2	0	0	-1	0	0	3
a_3	-1	1	0	2	0	0
a_4	-1	1	0	0	2	0
a_5	-1	0	0	1	0	0

Existência de Filogenia Perfeita

L	c_3	c_4	c_5	c_6	c_1	c_2
a_1	0	0	-1	0	0	0
a_2	0	0	-1	0	0	3
a_3	-1	1	0	2	0	0
a_4	-1	1	0	0	2	0
a_5	-1	0	0	1	0	0

Existência de Filogenia Perfeita

L		c_q		c_r		c_j
a_x		≠0		0		q
a_y				≠0		r
a_z				≠0		0

Existência de Filogenia Perfeita

M				c_r		c_j
a_x				0		1
a_y				1		1
a_z				1		0

Existência de Filogenia Perfeita

- Complexidade:
  - Ordenação da matriz de características binárias, de acordo com o número de 1's em cada coluna: $O(nm)$ (para contar o número de 1's em cada coluna) + $O(n + m)$ (para ordenar, usando Counting Sort) $= O(nm)$.
  - Inicialização da matriz L: $O(nm)$.
  - Definição da matriz L: $O(nm)$.
  - Busca por conjuntos A_i e A_j incompatíveis: $O(nm)$.
  - Total: $O(nm)$.
- Algoritmo proposto por Dan Gusfield, em 1991.

Construção de Filogenia Perfeita

Algoritmo 5: Perfect Phylogenetic Tree

Input: $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$, $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$,$M$
Sort the columns of M in nonincreasing order of numbers of ones ($\mathcal{C}' = \{c'_1, c'_2, \ldots, c'_m\}$)
$V \leftarrow \{\mathcal{A}\}$
$E \leftarrow \emptyset$
for *all* $c \in \{c'_1, c'_2, \ldots, c'_m\}$ **do**
    Search for the vertex $X \in V$ representing the smallest superset of $\mathcal{A}_c$
    $V \leftarrow V \cup \{\mathcal{A}_c\}$
    $E \leftarrow E \cup \{(X, \mathcal{A}_c)\}$
    $d(X, \mathcal{A}_j) \leftarrow c$
end
for *all* $i \in [1..n]$ **do**
    if $a_i \notin V$ **then**
        Search for the vertex $X \in V$ representing the smallest set containing a_i
        $V \leftarrow V \cup \{a_i\}$
        $E \leftarrow E \cup \{(X, \{a_i\})\}$
        $d(X, \{a_i\}) \leftarrow \lambda$
    end
end
return $T = (V, E, d)$

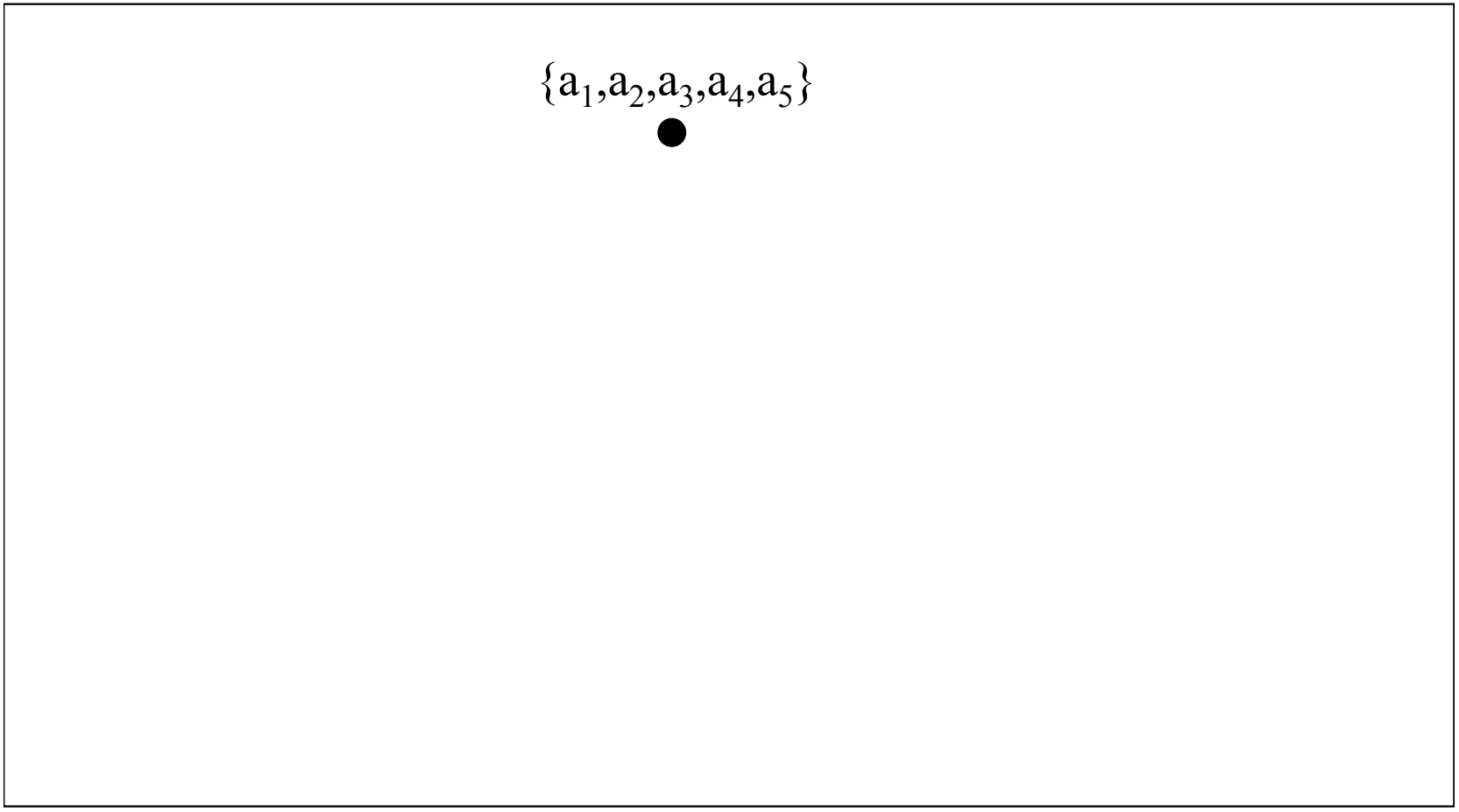
Construção de Filogenia Perfeita

	c_1	c_2	c_3	c_4	c_5	c_6
a_1	0	0	0	0	1	0
a_2	0	1	0	0	1	0
a_3	0	0	1	1	0	0
a_4	1	0	1	1	0	0
a_5	0	0	1	0	0	1

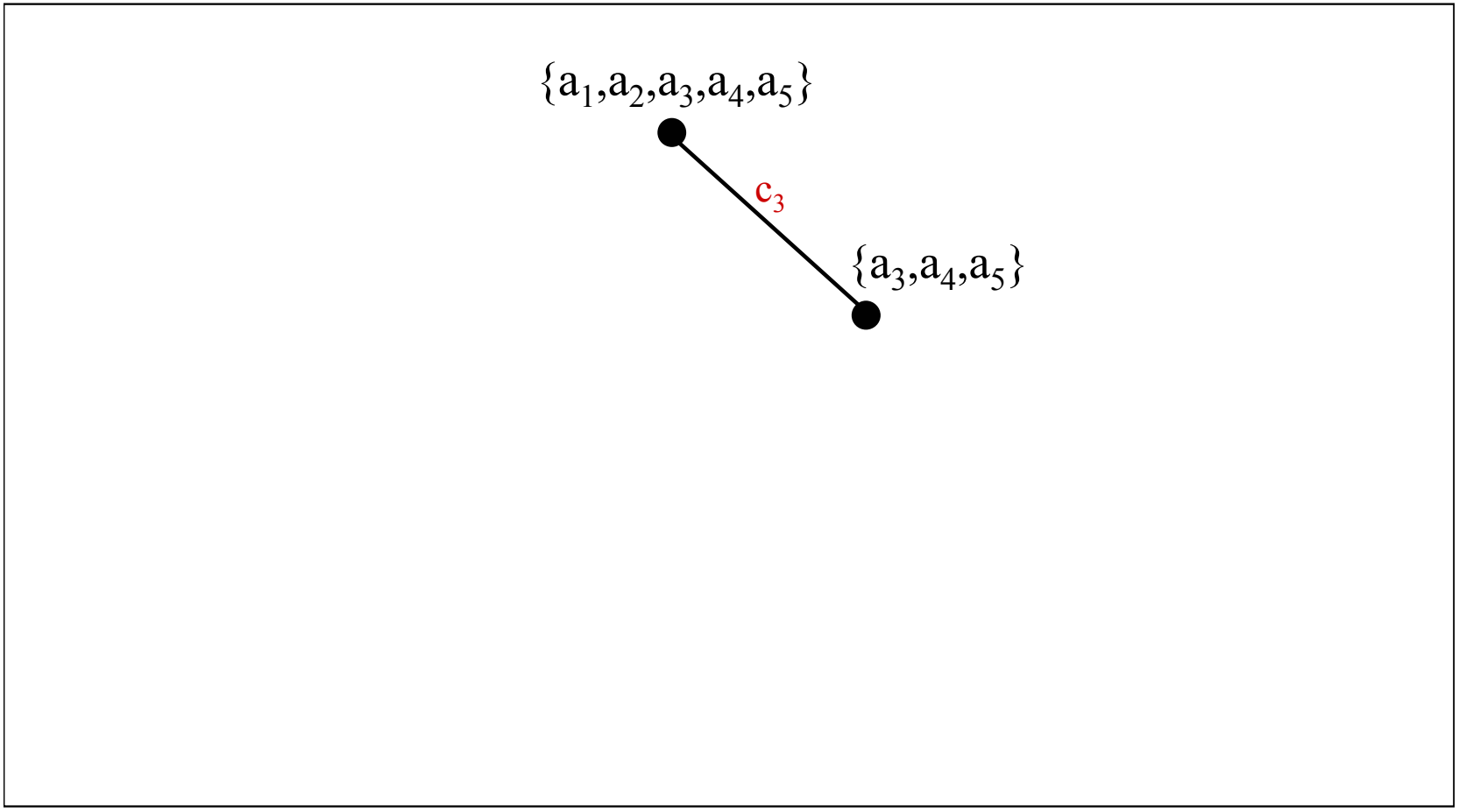
Construção de Filogenia Perfeita

	c_3	c_4	c_5	c_1	c_2	c_6
a_1	0	0	1	0	0	0
a_2	0	0	1	0	1	0
a_3	1	1	0	0	0	0
a_4	1	1	0	1	0	0
a_5	1	0	0	0	0	1

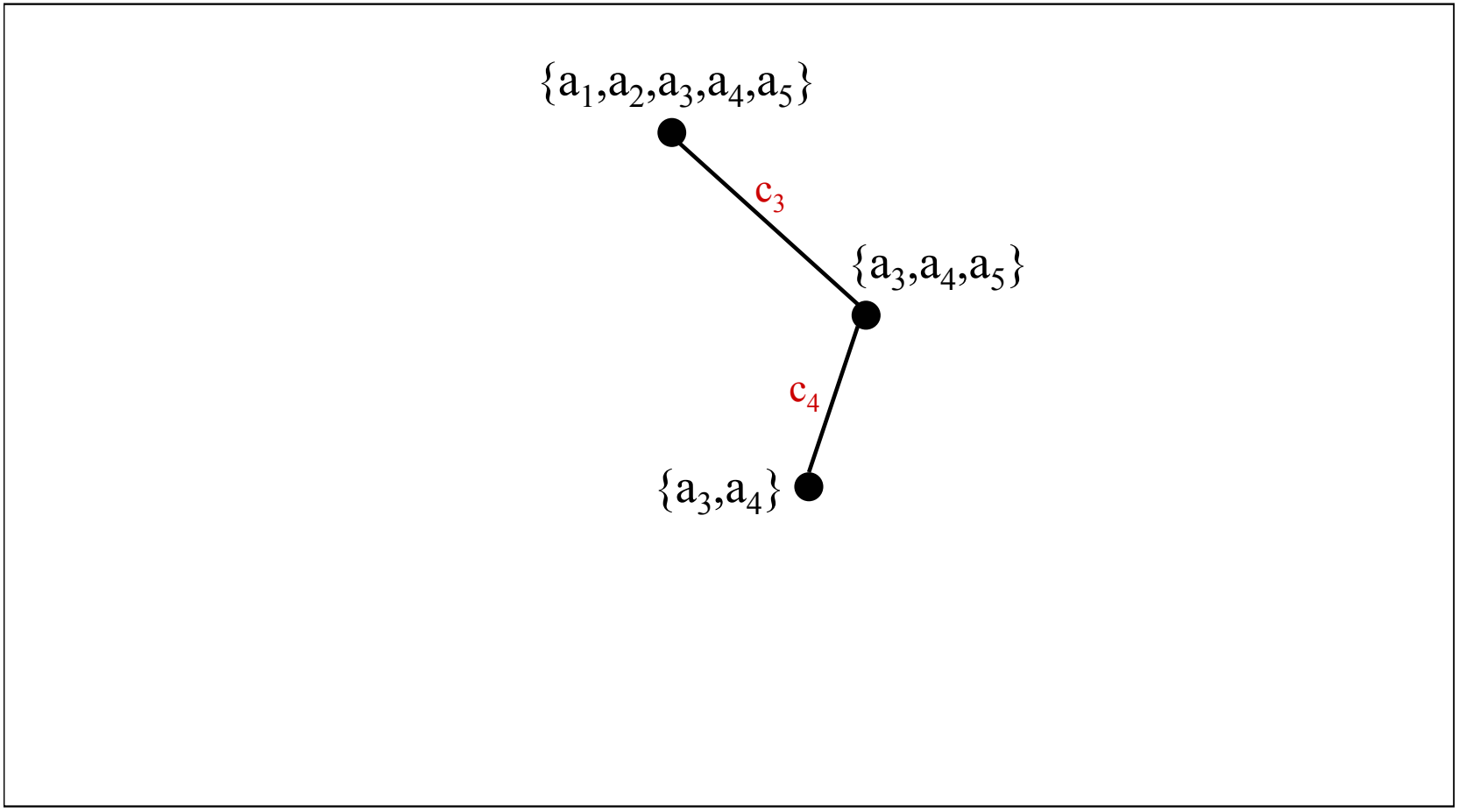
Construção de Filogenia Perfeita



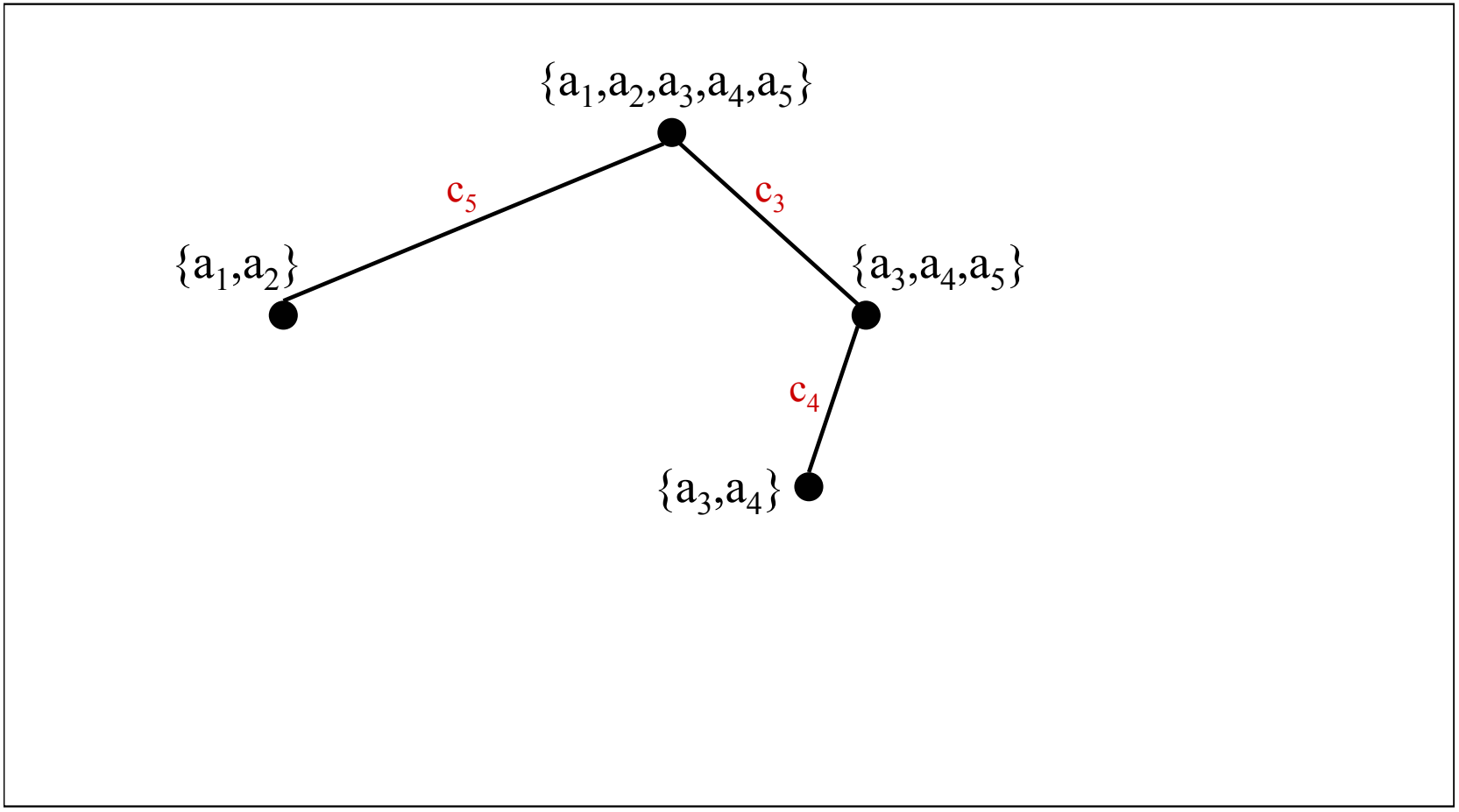
Construção de Filogenia Perfeita



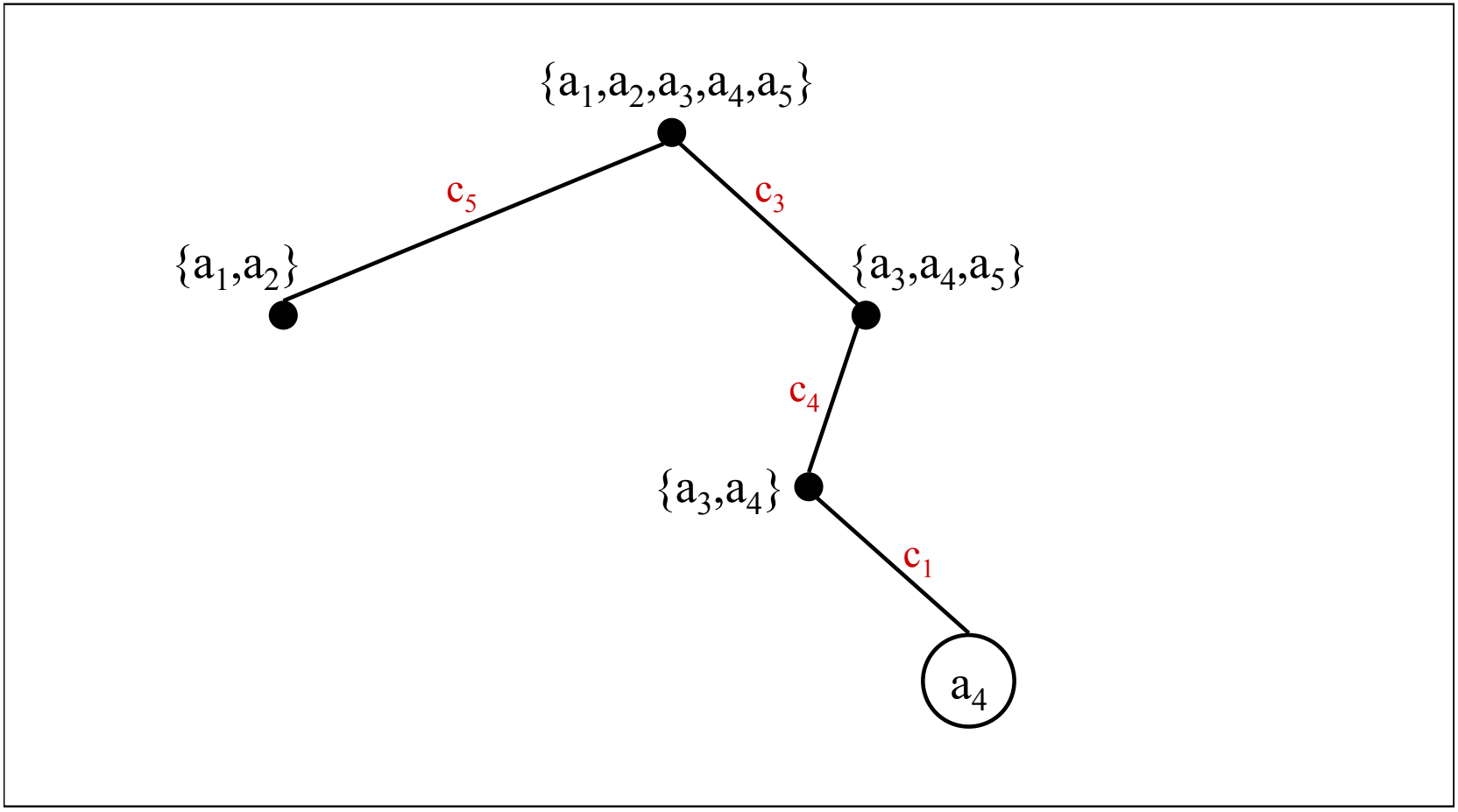
Construção de Filogenia Perfeita



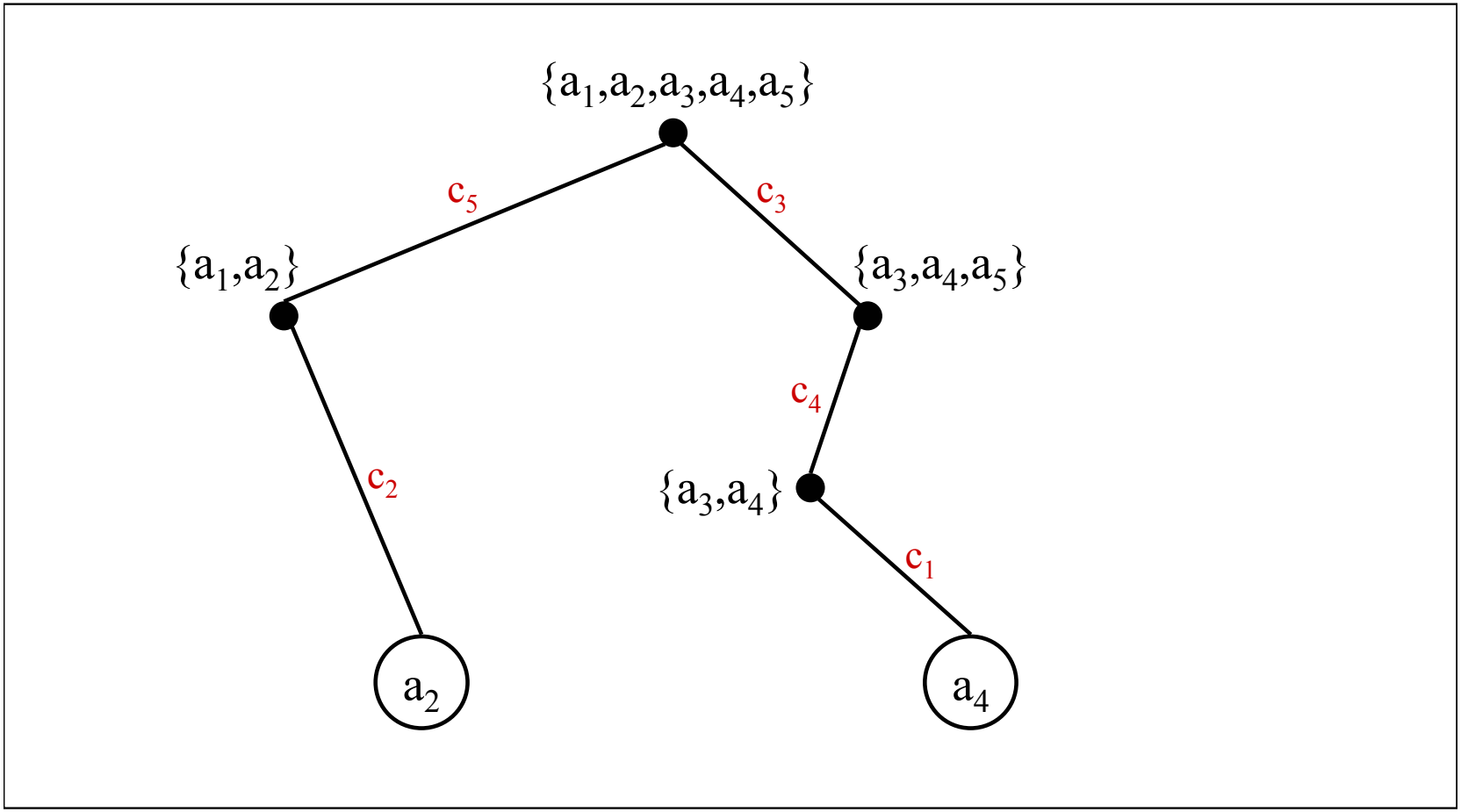
Construção de Filogenia Perfeita



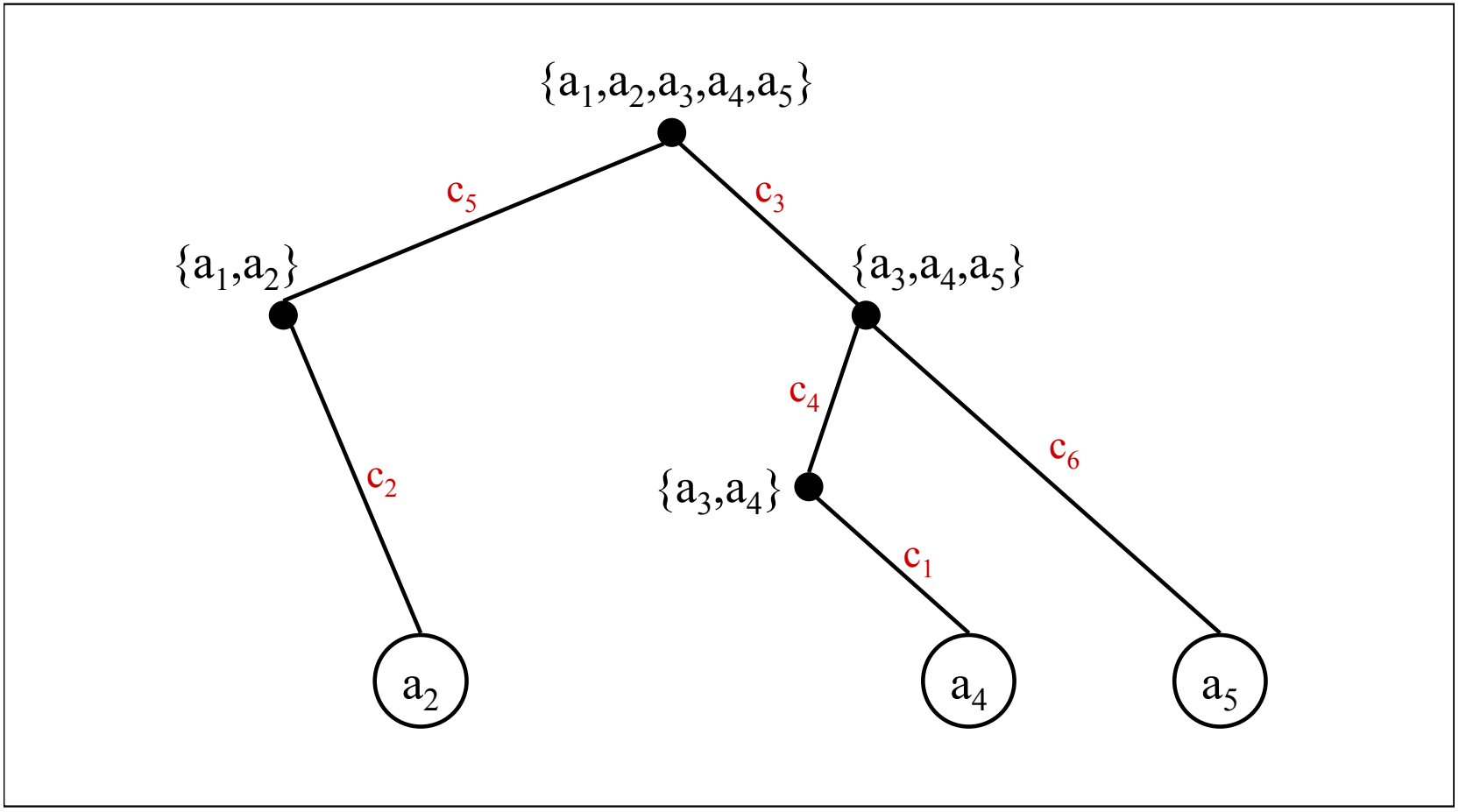
Construção de Filogenia Perfeita



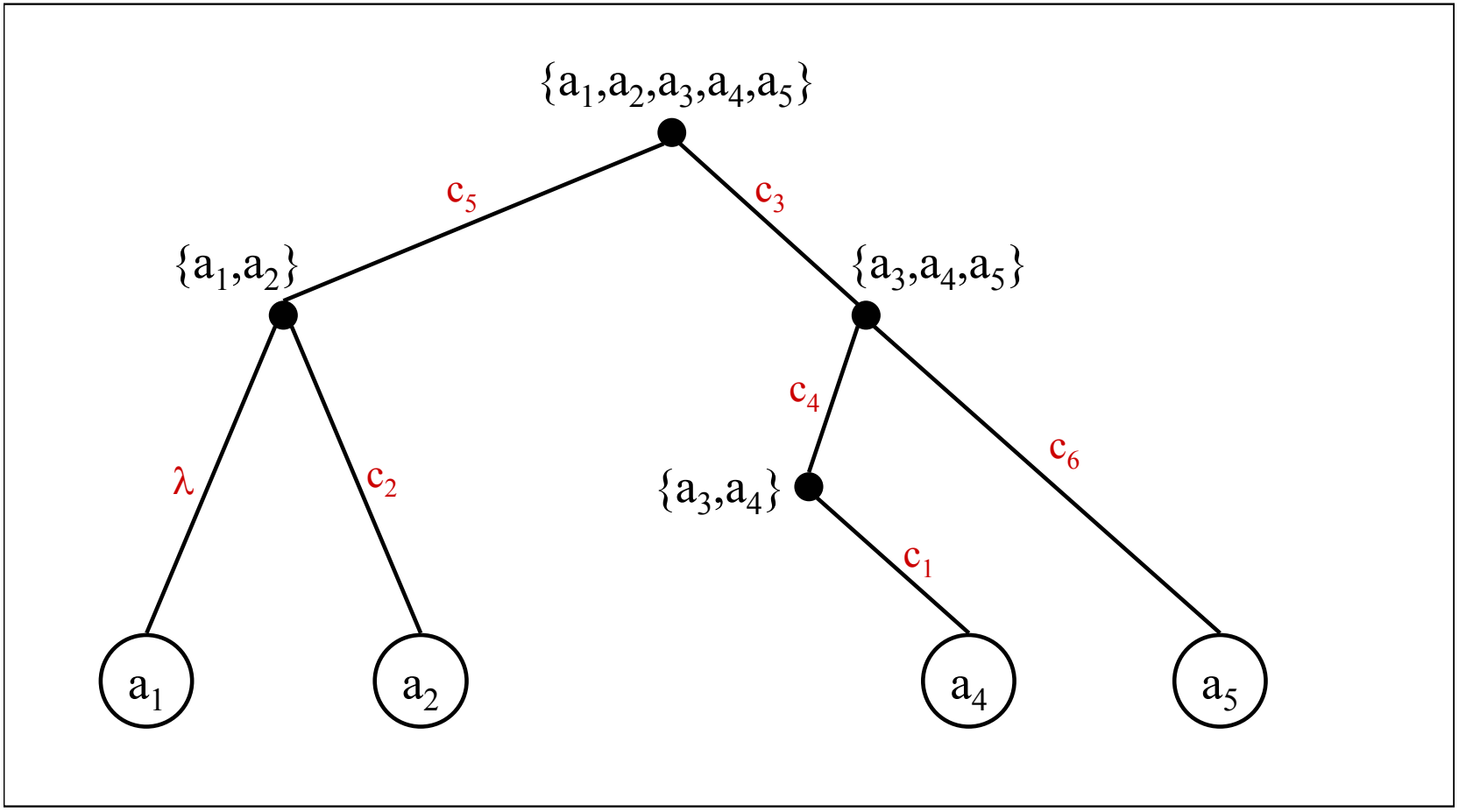
Construção de Filogenia Perfeita



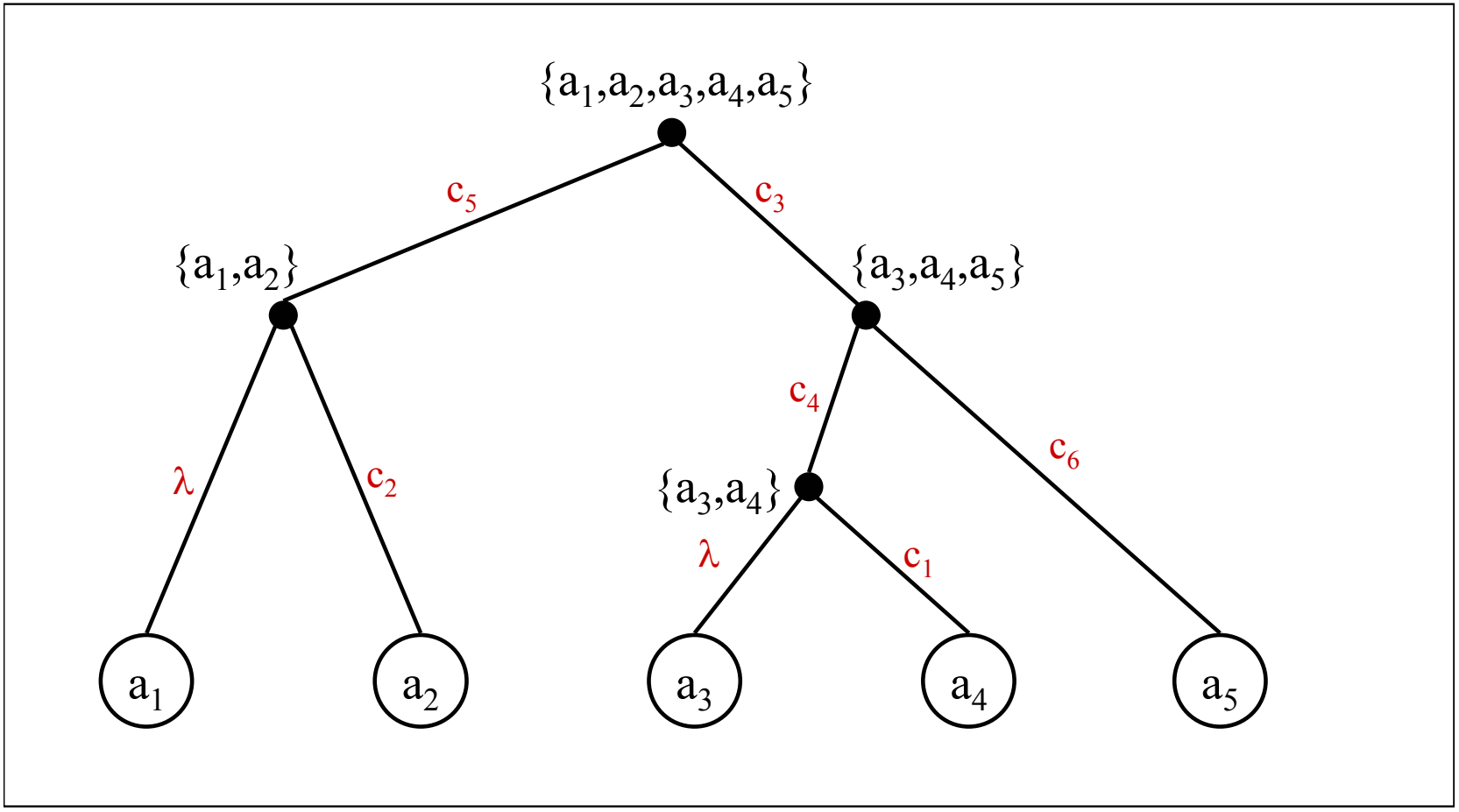
Construção de Filogenia Perfeita



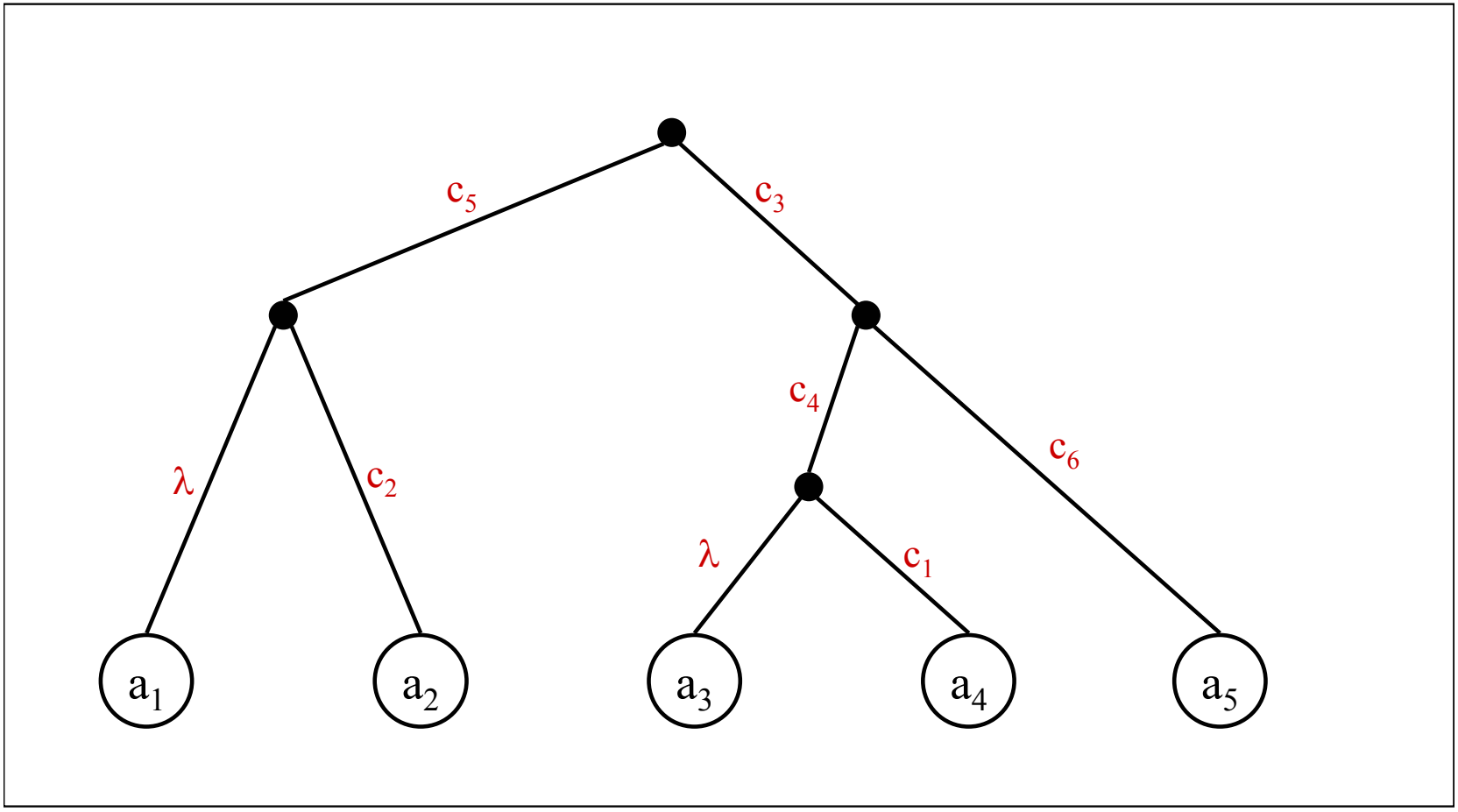
Construção de Filogenia Perfeita



Construção de Filogenia Perfeita



Construção de Filogenia Perfeita



Construção de Filogenia Perfeita

- Complexidade:
  - Ordenação da matriz de características binárias, de acordo com o número de 1's em cada coluna: $O(nm)$ (para contar o número de 1's em cada coluna) + $O(n+m)$ (para ordenar, usando Counting Sort) $= O(nm)$.
  - Busca do vértice X que representa o menor conjunto que contém $\mathcal{A}_j$: $O(m) \times O(nm) = O(nm^2)$.
  - Busca do vértice X que contém o objeto a_i (para $1 \leq i \leq n$), usando uma tabela auxiliar para armazenar o menor conjunto que contém cada objeto de $\mathcal{A}$: $O(nm)$ (para criar e atualizar a tabela a cada nova inserção de um vértice na árvore) + $n \times O(1) = O(n)$ (para acessar a tabela e criar as folhas faltantes) $= O(nm)$.
  - Total: $O(nm^2)$.

Construção de Filogenia Perfeita

- Algoritmo proposto por Dan Gusfield, em 1991.
- Hans Bodlaender, Mike Fellows e Tandy Warnow provaram, em 1992, que o problema de filogenia perfeita para estados não ordenados é $\mathcal{NP}$-Completo, independente do número de estados de cada característica.

PHYLIP

- *PHYLIP*: PHYLogeny Inference Package.
- Pacote gratuito e multiplataforma de análise filogenética desenvolvido Joseph Felsenstein em 1989, e mantida pela Universidade de Washington.
- É capaz de resolver a maioria das análises filogenéticas existentes na literatura atual.
- Aceita uma grande variedade de tipos de dados de entrada, como, por exemplo, sequências moleculares, frequência de genes, matriz de distância e características discretas.