Mais programação dinâmica

CLRS 15.5

= "recursão–com–tabela"
= transformação inteligente de recursão em iteração

Buscas em um conjunto conhecido

Suponha que temos um vetor $v[1\,..\,n]$ de inteiros, com n grande, e que vamos fazer um número gigantesco $Q \gg n$ de consultas.

Cada consulta tem a forma: o inteiro k está no vetor v?

Buscas em um conjunto conhecido

Suponha que temos um vetor $v[1\,..\,n]$ de inteiros, com n grande, e que vamos fazer um número gigantesco $Q \gg n$ de consultas.

Cada consulta tem a forma: o inteiro k está no vetor v?

Exemplos com "strings" no lugar de "inteiros":
spell checker, dicionário, tradutor.

Buscas em um conjunto conhecido

Suponha que temos um vetor $v[1\,..\,n]$ de inteiros, com n grande, e que vamos fazer um número gigantesco $Q \gg n$ de consultas.

Cada consulta tem a forma: o inteiro k está no vetor v?

Exemplos com "strings" no lugar de "inteiros":
spell checker, dicionário, tradutor.

Problema: Como armazenar o conteúdo de $v[1\,..\,n]$?

Buscas em um conjunto conhecido

Suponha que temos um vetor $v[1\,..\,n]$ de inteiros, com n grande, e que vamos fazer um número gigantesco $Q \gg n$ de consultas.

Cada consulta tem a forma: o inteiro k está no vetor v?

Exemplos com "strings" no lugar de "inteiros":
spell checker, dicionário, tradutor.

Problema: Como armazenar o conteúdo de $v[1\,..\,n]$?

Podemos ordenar v e aplicar busca binária.

Buscas em um conjunto conhecido

Suponha que temos um vetor $v[1\,..\,n]$ de inteiros, com n grande, e que vamos fazer um número gigantesco $Q \gg n$ de consultas.

Cada consulta tem a forma: o inteiro k está no vetor v?

Exemplos com "strings" no lugar de "inteiros":
spell checker, dicionário, tradutor.

Problema: Como armazenar o conteúdo de $v[1\,..\,n]$?

Podemos ordenar v e aplicar busca binária.

Se sabemos a frequência de buscas por certos itens, podemos fazer algo melhor?

Buscas em conjunto conhecido

Dadas estimativas $e[1\,..\,n]$ do número de acessos a cada elemento de $v[1\,..\,n]$, qual é a melhor estrutura de dados para v?

Buscas em conjunto conhecido

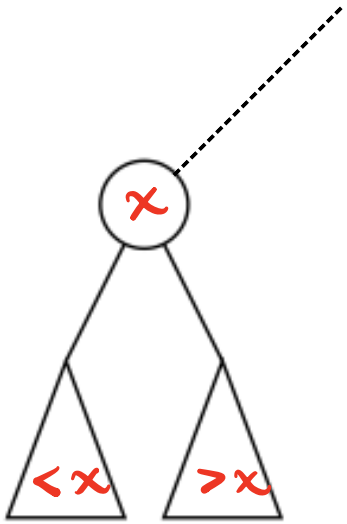
Dadas estimativas $e[1\,..\,n]$ do número de acessos a cada elemento de $v[1\,..\,n]$, qual é a melhor estrutura de dados para v?

Árvore de busca binária (ABB)?

Buscas em conjunto conhecido

Dadas estimativas $e[1\,..\,n]$ do número de acessos a cada elemento de $v[1\,..\,n]$, qual é a melhor estrutura de dados para v?

Árvore de busca binária (ABB)?



Buscas em conjunto conhecido

Dadas estimativas $e[1\,..\,n]$ do número de acessos a cada elemento de $v[1\,..\,n]$, qual é a melhor estrutura de dados para v?

Árvore de busca binária (ABB)?

Exemplo: $n = 3$ e $e_1 = 10$, $e_2 = 20$, $e_3 = 40$.

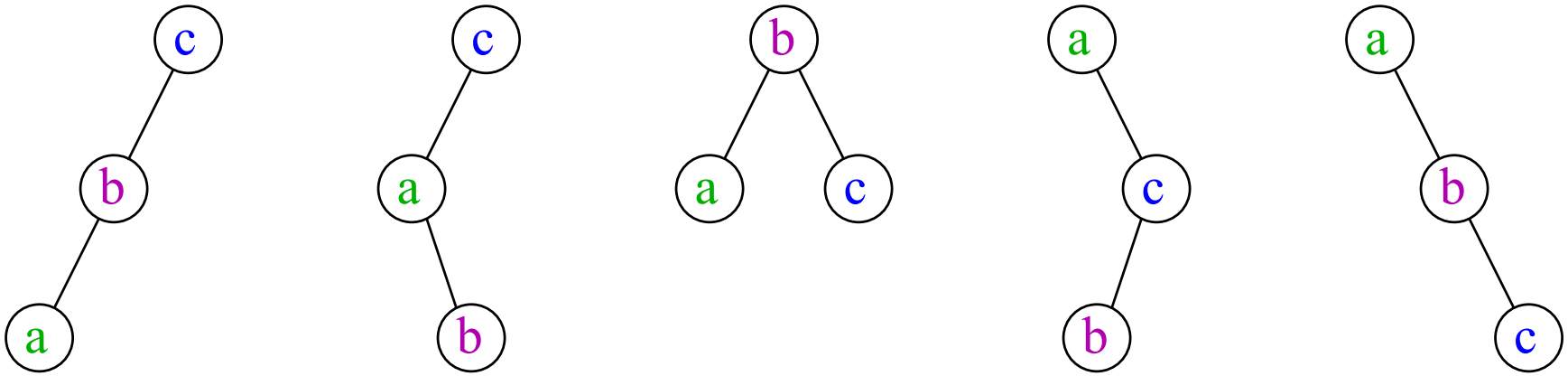
Buscas em conjunto conhecido

Dadas estimativas $e[1\,..\,n]$ do número de acessos a cada elemento de $v[1\,..\,n]$, qual é a melhor estrutura de dados para v?

Árvore de busca binária (ABB)?

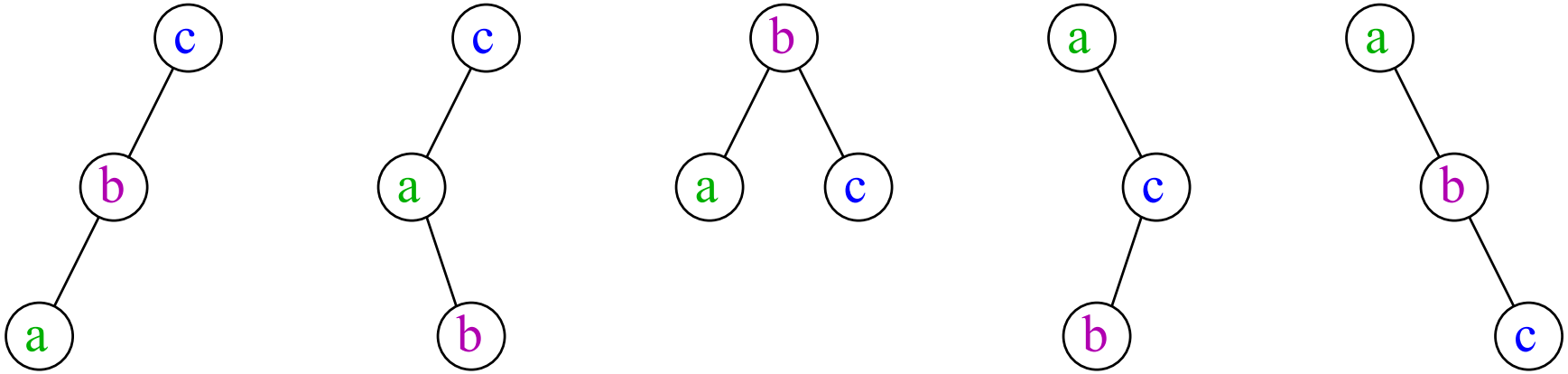
Exemplo: $n = 3$ e $e_1 = 10$, $e_2 = 20$, $e_3 = 40$.

Qual a melhor das ABBs?



Exemplo

Exemplo: $n = 3$ e $e_1 = 10$, $e_2 = 20$, $e_3 = 40$.

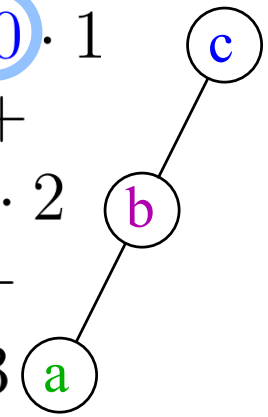


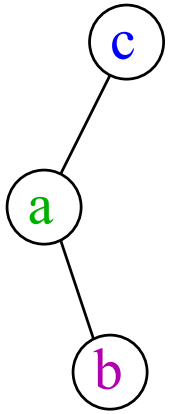
Qual a melhor das ABBs?

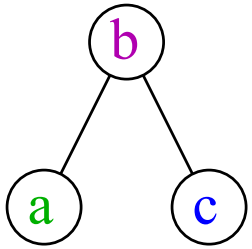
Exemplo

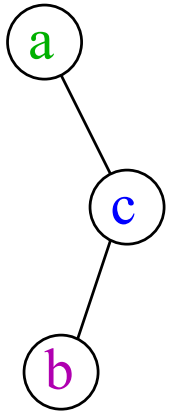
Exemplo: $n = 3$ e $e_1 = 10$, $e_2 = 20$, $e_3 = 40$.

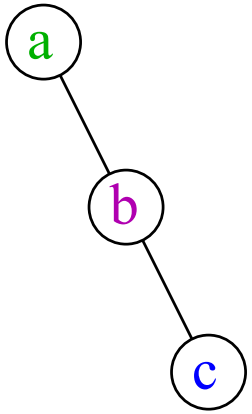
$40 \cdot 1 + 20 \cdot 2 + 10 \cdot 3 = 110$







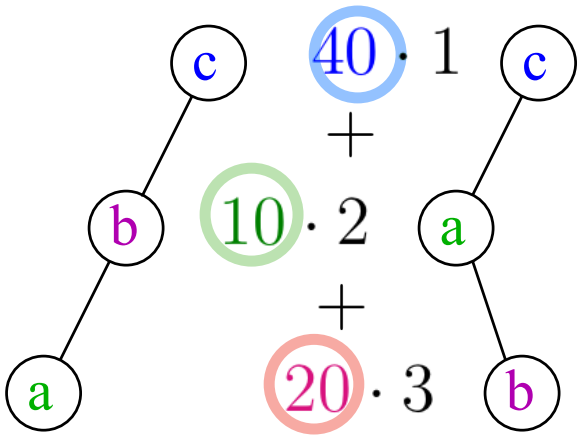


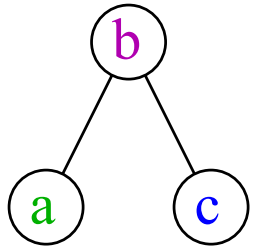


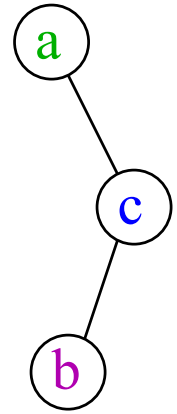
Qual a melhor das ABBs?

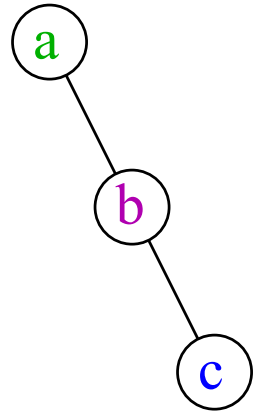
Exemplo

Exemplo: $n = 3$ e $e_1 = 10$, $e_2 = 20$, $e_3 = 40$.





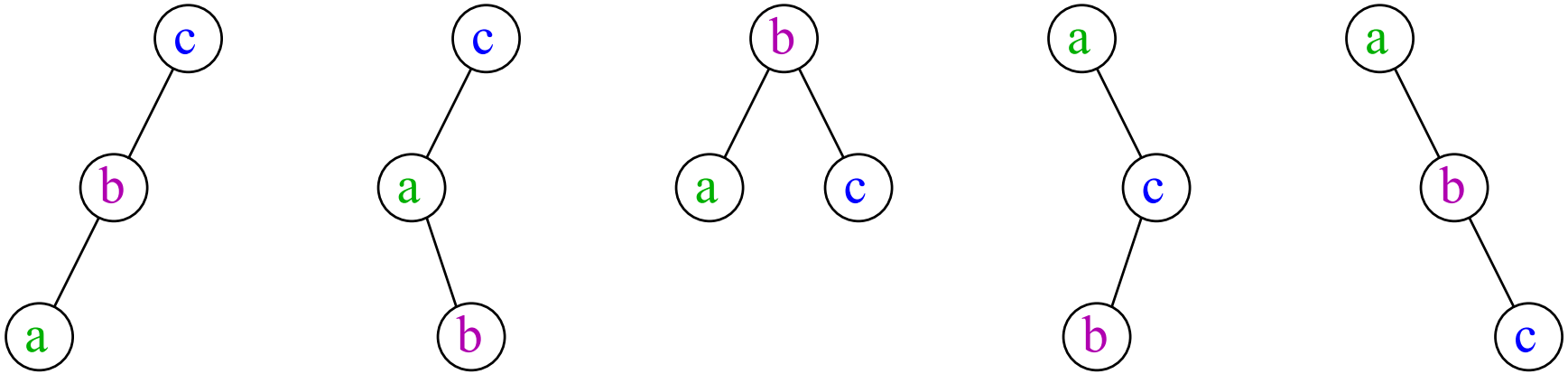




Qual a melhor das ABBs?

Exemplo

Exemplo: $n = 3$ e $e_1 = 10$, $e_2 = 20$, $e_3 = 40$.

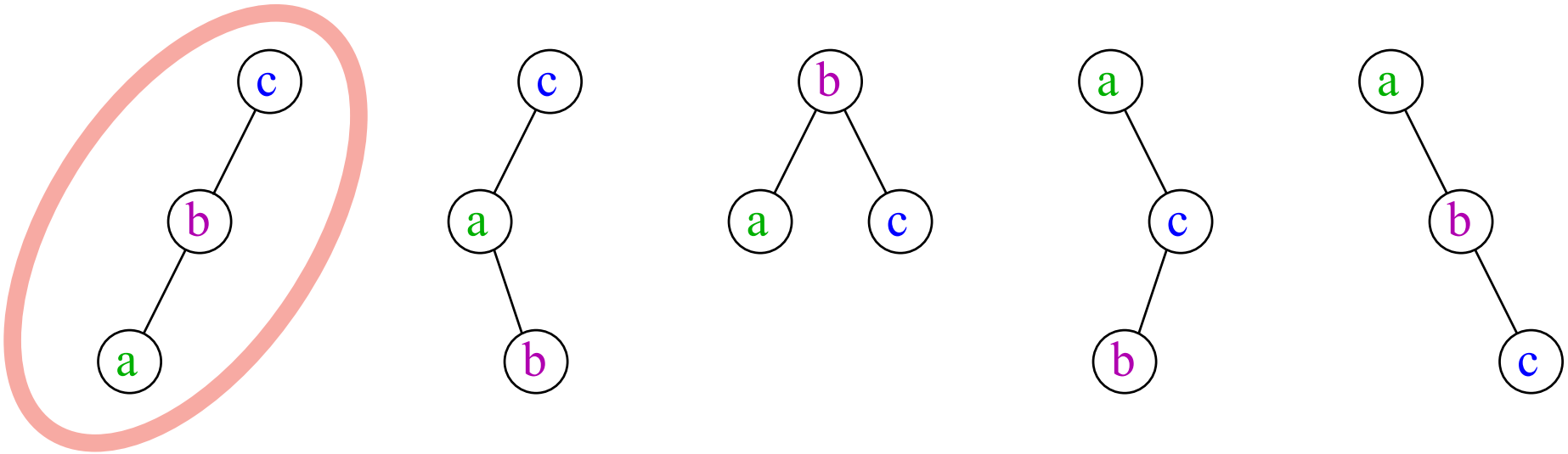


Número esperado de comparações:

- $10 \cdot 3 + 20 \cdot 2 + 40 \cdot 1 = 110$
- $10 \cdot 2 + 20 \cdot 3 + 40 \cdot 1 = 120$
- $10 \cdot 2 + 20 \cdot 1 + 40 \cdot 2 = 120$
- $10 \cdot 1 + 20 \cdot 3 + 40 \cdot 2 = 150$
- $10 \cdot 1 + 20 \cdot 2 + 40 \cdot 3 = 170$

Exemplo

Exemplo: $n = 3$ e $e_1 = 10$, $e_2 = 20$, $e_3 = 40$.



Número esperado de comparações:

- $10 \cdot 3 + 20 \cdot 2 + 40 \cdot 1 = 110$ $\longleftarrow$ ABB ótima
- $10 \cdot 2 + 20 \cdot 3 + 40 \cdot 1 = 120$
- $10 \cdot 2 + 20 \cdot 1 + 40 \cdot 2 = 120$
- $10 \cdot 1 + 20 \cdot 3 + 40 \cdot 2 = 150$
- $10 \cdot 1 + 20 \cdot 2 + 40 \cdot 3 = 170$

Árvore de busca ótima

Considere um vetor $e[1\,..\,n]$ de inteiros com uma estimativa do número de acessos a cada elemento de $\{1, \ldots, n\}$.

Árvore de busca ótima

Considere um vetor $e[1\,..\,n]$ de inteiros com uma estimativa do número de acessos a cada elemento de $\{1, \ldots, n\}$.

Uma ABB ótima com respeito ao vetor e é uma ABB para o conjunto $\{1, \ldots, n\}$ que minimiza o número

$$\sum_{i=1}^{n} h_i \, e_i,$$

onde h_i é o número de nós no caminho de i até a raiz da árvore (que é o número de comparações feitas na busca por i).

Árvore de busca ótima

Considere um vetor $e[1\,..\,n]$ de inteiros com uma estimativa do número de acessos a cada elemento de $\{1, \ldots, n\}$.

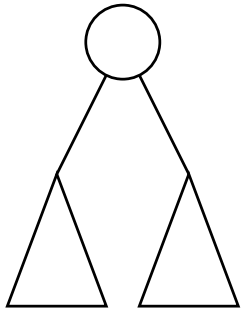
Uma ABB ótima com respeito ao vetor e é uma ABB para o conjunto $\{1, \ldots, n\}$ que minimiza o número

$$\sum_{i=1}^{n} h_i \, e_i,$$

onde h_i é o número de nós no caminho de i até a raiz da árvore (que é o número de comparações feitas na busca por i).

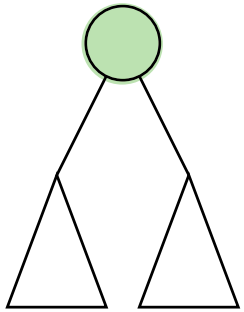
Problema (ABB Ótima): Dado $e[1\,..\,n]$, encontrar uma árvore de busca binária ótima com respeito a e.

Subestrutura ótima



Subárvores esquerda e direita de uma ABB ótima são ABBs ótimas.

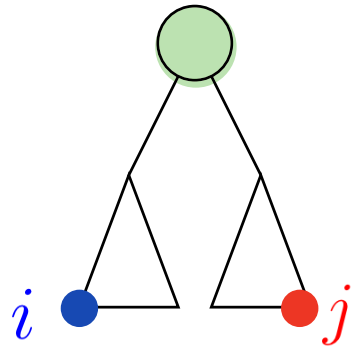
Subestrutura ótima



Subárvores esquerda e direita de uma ABB ótima são ABBs ótimas.

Resta determinar a raiz da ABB ótima.

Subestrutura ótima

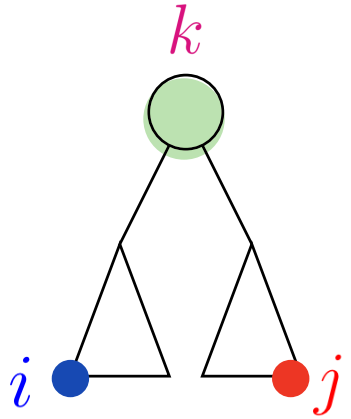


Subárvores esquerda e direita de uma ABB ótima são ABBs ótimas.

Resta determinar a raiz da ABB ótima.

$c[i, j]$: custo min de uma ABB para $e[i \,..\, j]$
$s[i, j]$: soma dos acessos em $e[i \,..\, j]$

Subestrutura ótima



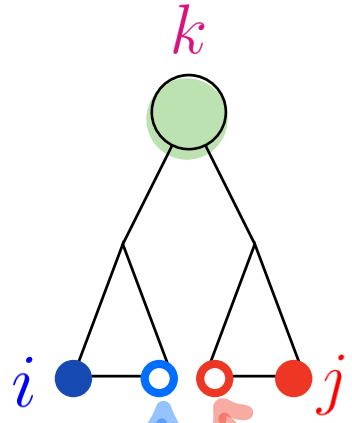
Subárvores esquerda e direita de uma ABB ótima são ABBs ótimas.

Resta determinar a raiz da ABB ótima.

$c[i,j]$: custo min de uma ABB para $e[i\,..\,j]$
$s[i,j]$: soma dos acessos em $e[i\,..\,j]$

$$c[i,j] = \begin{cases} 0 & \text{se } i > j \\ \min_{i \le k \le j}\{c[i,k-1] + c[k+1,j] + s[i,j]\} & \text{se } i \le j \end{cases}$$

Subestrutura ótima



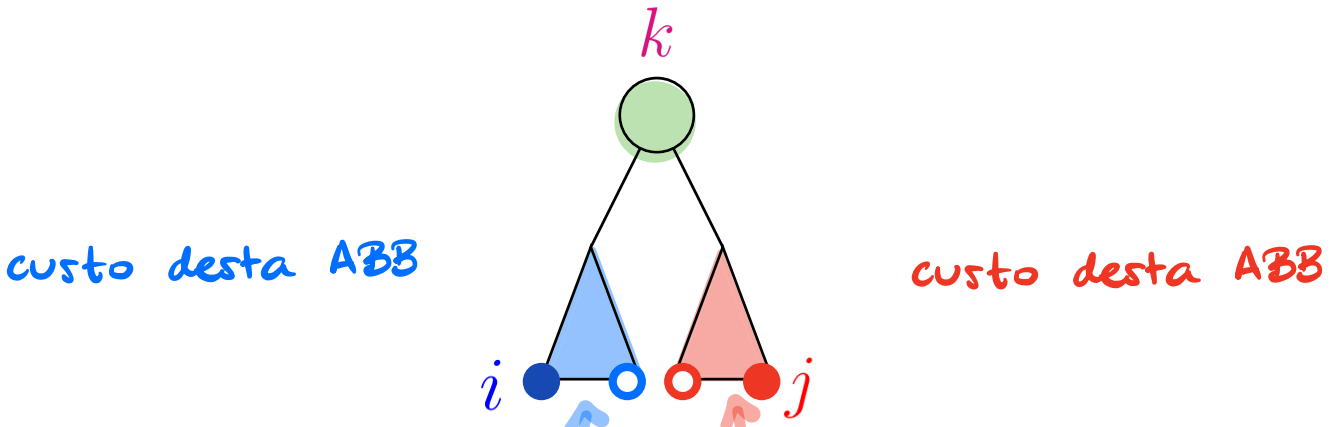
Subárvores esquerda e direita de uma ABB ótima são ABBs ótimas.

Resta determinar a raiz da ABB ótima.

$c[i,j]$: custo min de uma ABB para $e[i\,..\,j]$
$s[i,j]$: soma dos acessos em $e[i\,..\,j]$

$$c[i,j] = \begin{cases} 0 & \text{se } i > j \\ \min_{i \le k \le j}\{c[i,k-1] + c[k+1,j] + s[i,j]\} & \text{se } i \le j \end{cases}$$

Subestrutura ótima



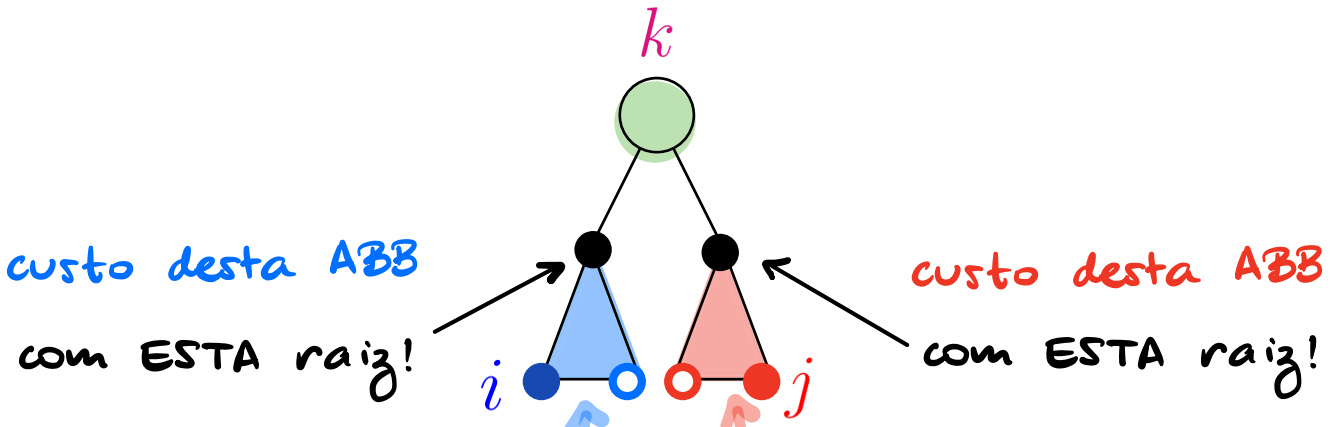
Subárvores esquerda e direita de uma ABB ótima são ABBs ótimas.

Resta determinar a raiz da ABB ótima.

$c[i,j]$: custo min de uma ABB para $e[i\,..\,j]$
$s[i,j]$: soma dos acessos em $e[i\,..\,j]$

$$c[i,j] = \begin{cases} 0 & \text{se } i > j \\ \min_{i \le k \le j}\{c[i,k-1] + c[k+1,j] + s[i,j]\} & \text{se } i \le j \end{cases}$$

Subestrutura ótima



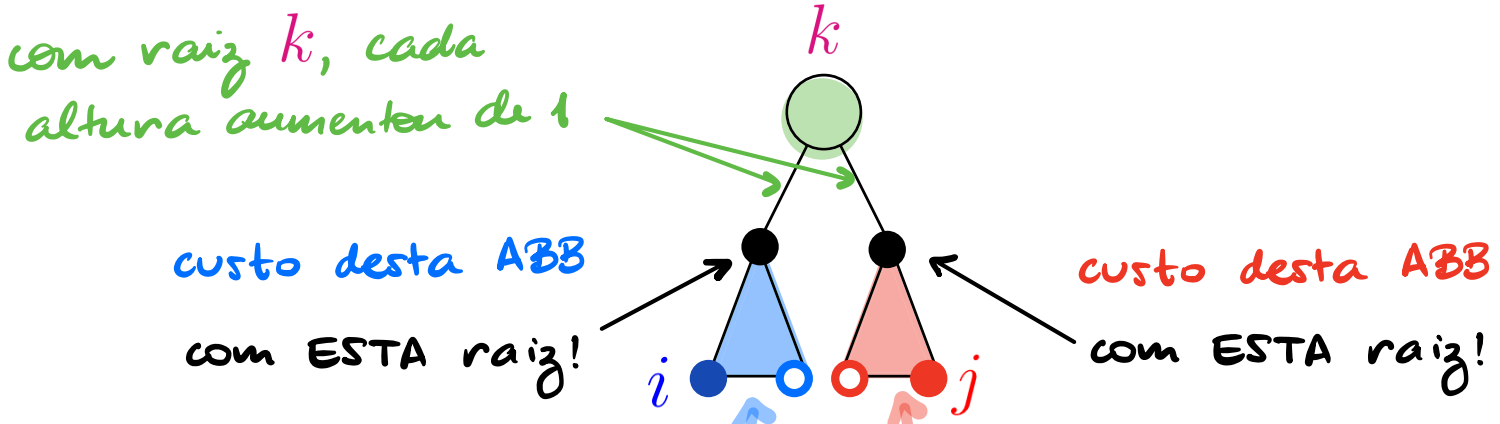
Subárvores esquerda e direita de uma ABB ótima são ABBs ótimas.

Resta determinar a raiz da ABB ótima.

$c[i,j]$: custo min de uma ABB para $e[i\,..\,j]$
$s[i,j]$: soma dos acessos em $e[i\,..\,j]$

$$c[i,j] = \begin{cases} 0 & \text{se } i > j \\ \min_{i \le k \le j}\{c[i,k-1] + c[k+1,j] + s[i,j]\} & \text{se } i \le j \end{cases}$$

Subestrutura ótima



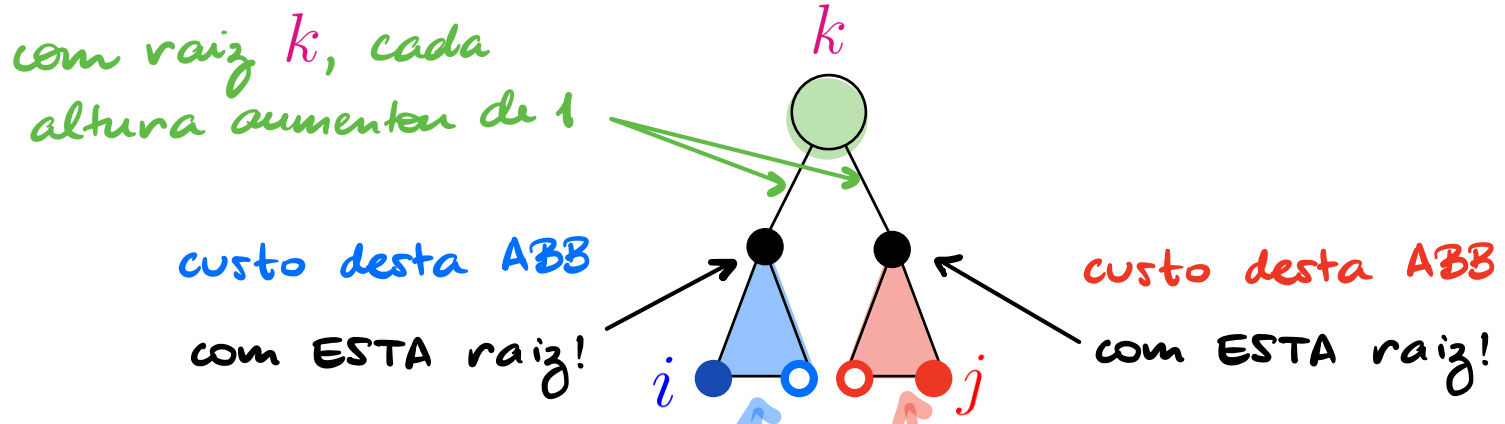
Subárvores esquerda e direita de uma ABB ótima são ABBs ótimas.

Resta determinar a raiz da ABB ótima.

$c[i,j]$: custo min de uma ABB para $e[i\,..\,j]$
$s[i,j]$: soma dos acessos em $e[i\,..\,j]$

$$c[i,j] = \begin{cases} 0 & \text{se } i > j \\ \min_{i \le k \le j}\{c[i,k-1] + c[k+1,j] + s[i,j]\} & \text{se } i \le j \end{cases}$$

Subestrutura ótima



Subárvores esquerda e direita de uma ABB ótima são ABBs ótimas.

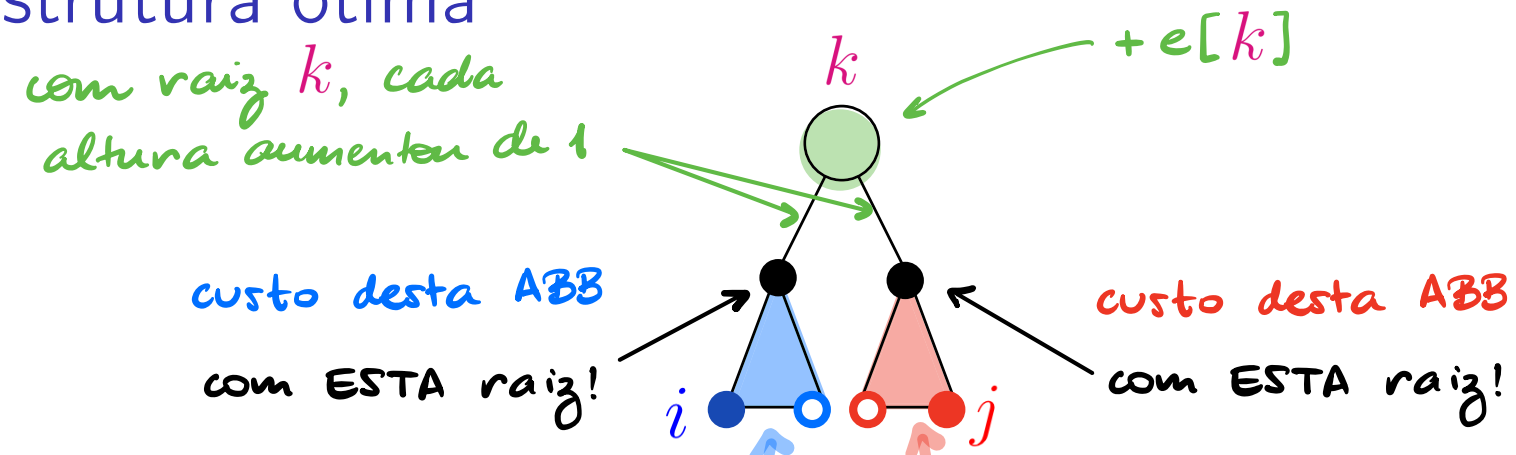
Resta determinar a raiz da ABB ótima.

$c[i,j]$: custo min de uma ABB para $e[i\,..\,j]$
$s[i,j]$: soma dos acessos em $e[i\,..\,j]$

$$c[i,j] = \begin{cases} 0 & \text{se } i > j \\ \min_{i \le k \le j}\{c[i,k-1] + c[k+1,j] + s[i,j]\} & \text{se } i \le j \end{cases}$$

$+\,s[i,k-1]$ $\quad$ $+\,s[k+1,j]$

Subestrutura ótima



Subárvores esquerda e direita de uma ABB ótima são ABBs ótimas.

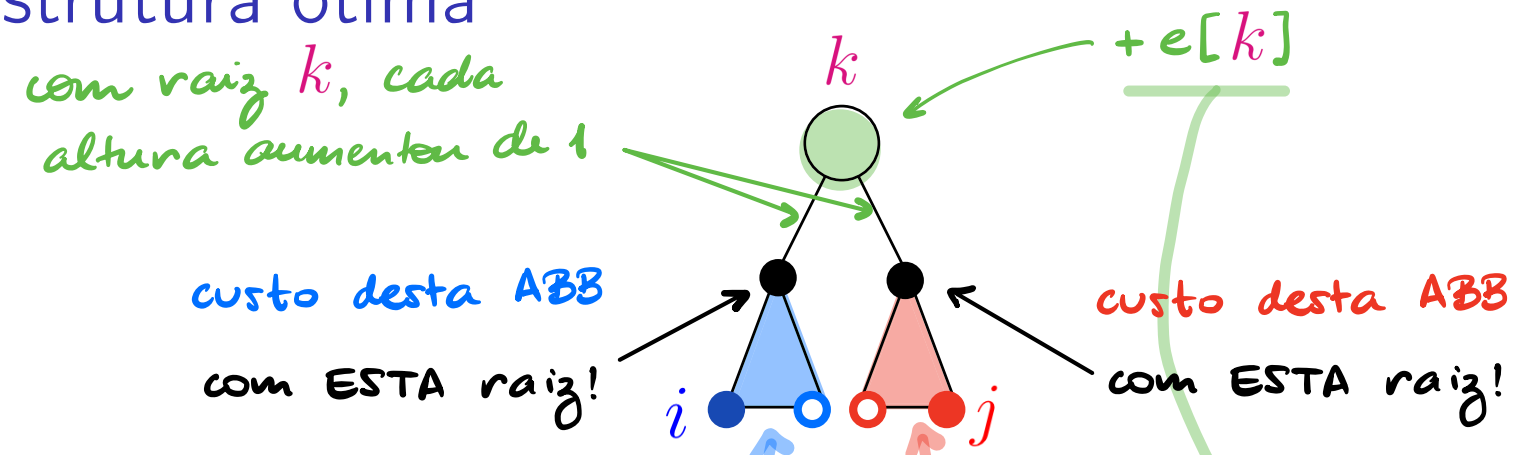
Resta determinar a raiz da ABB ótima.

$c[i,j]$: custo min de uma ABB para $e[i\,..\,j]$
$s[i,j]$: soma dos acessos em $e[i\,..\,j]$

$$c[i,j] = \begin{cases} 0 & \text{se } i > j \\ \min_{i \le k \le j} \{ c[i,k-1] + c[k+1,j] + s[i,j] \} & \text{se } i \le j \end{cases}$$

$+\,s[i,k-1]$ $\quad$ $+\,s[k+1,j]$

Subestrutura ótima



Subárvores esquerda e direita de uma ABB ótima são ABBs ótimas.

Resta determinar a raiz da ABB ótima.

$c[i,j]$: custo min de uma ABB para $e[i\,..\,j]$
$s[i,j]$: soma dos acessos em $e[i\,..\,j]$

$$c[i,j] = \begin{cases} 0 & \text{se } i > j \\ \min_{i \le k \le j} \{c[i,k-1] + c[k+1,j] + s[i,j]\} & \text{se } i \le j \end{cases}$$

$+s[i,k-1]$ $+s[k+1,j]$ $+$ $=$

Custo de uma ABB ótima

$c[i,j]$: custo min de uma ABB para $e[i\,..\,j]$

$s[j]$: soma dos acessos em $e[1\,..\,j]$

$s[j] - s[i-1]$: soma dos acessos em $e[i\,..\,j]$

$$c[i,j] = \begin{cases} 0 & \text{se } i > j \\ \min_{i \le k \le j}\{c[i,k-1] + c[k+1,j]\} + s[j] - s[i-1] & \text{se } i \le j \end{cases}$$

Custo de uma ABB ótima

$c[i,j]$: custo min de uma ABB para $e[i\,..\,j]$

$s[j]$: soma dos acessos em $e[1\,..\,j]$

$s[j] - s[i-1]$: soma dos acessos em $e[i\,..\,j]$

$$c[i,j] = \begin{cases} 0 & \text{se } i > j \\ \min_{i \le k \le j}\{c[i,k-1] + c[k+1,j]\} + s[j] - s[i-1] & \text{se } i \le j \end{cases}$$

Para calcular s:

```
1  s[0] ← 0
2  para i ← 1 até n faça
3      s[i] ← s[i−1] + e[i]
```

Custo de uma ABB ótima

$c[i,j]$: custo min de uma ABB para $e[i\,..\,j]$

$s[j]$: soma dos acessos em $e[1\,..\,j]$

$s[j] - s[i-1]$: soma dos acessos em $e[i\,..\,j]$

$$c[i,j] = \begin{cases} 0 & \text{se } i > j \\ \min_{i \le k \le j}\{c[i,k-1] + c[k+1,j]\} + s[j] - s[i-1] & \text{se } i \le j \end{cases}$$

Como preencher a matriz c?

Em que ordem?

Custo de uma ABB ótima

$c[i,j]$: custo min de uma ABB para $e[i\,..\,j]$

$s[j]$: soma dos acessos em $e[1\,..\,j]$

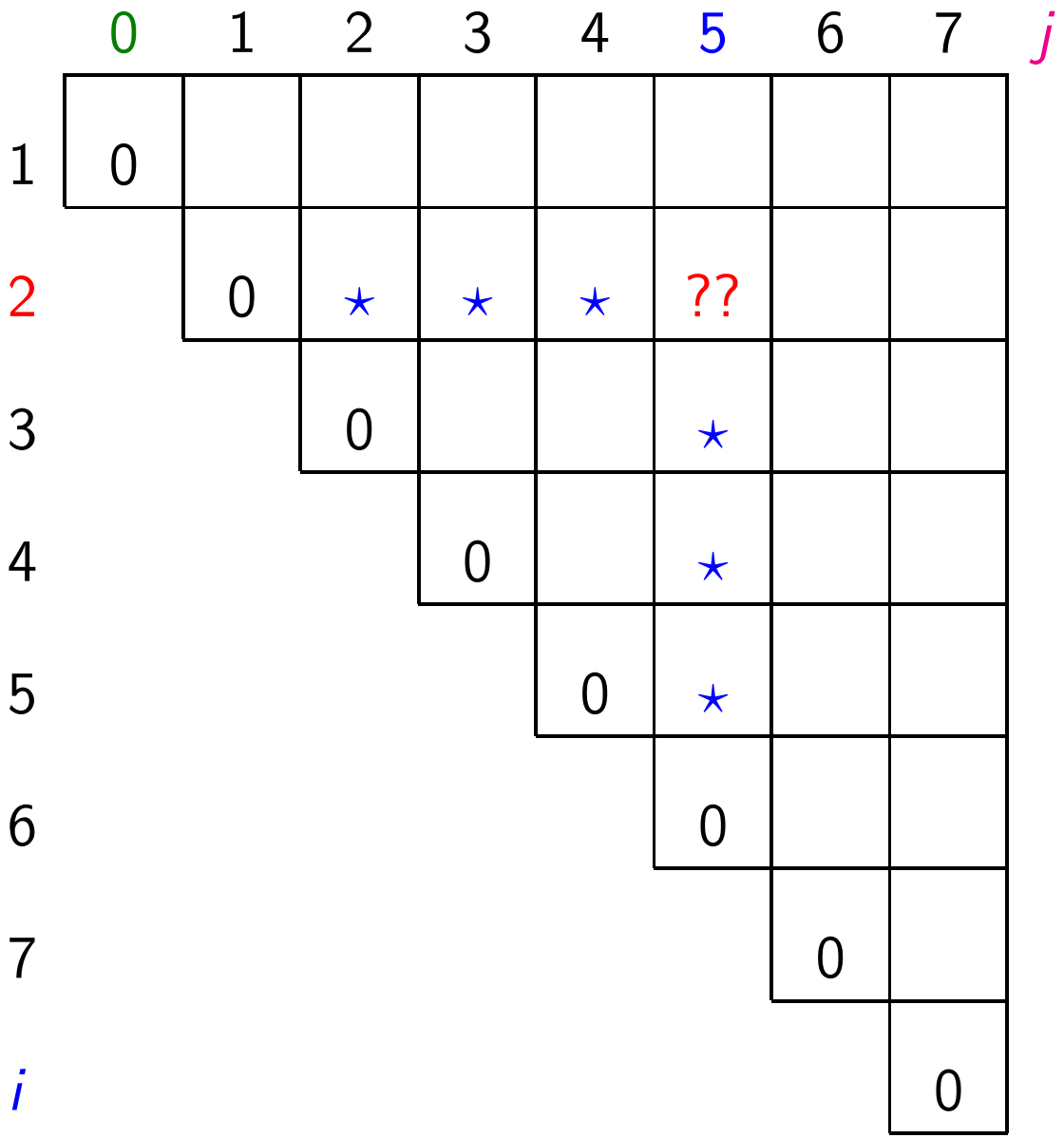
$s[j] - s[i-1]$: soma dos acessos em $e[i\,..\,j]$

$$c[i,j] = \begin{cases} 0 & \text{se } i > j \\ \min_{i \le k \le j}\{c[i,k-1] + c[k+1,j]\} + s[j] - s[i-1] & \text{se } i \le j \end{cases}$$

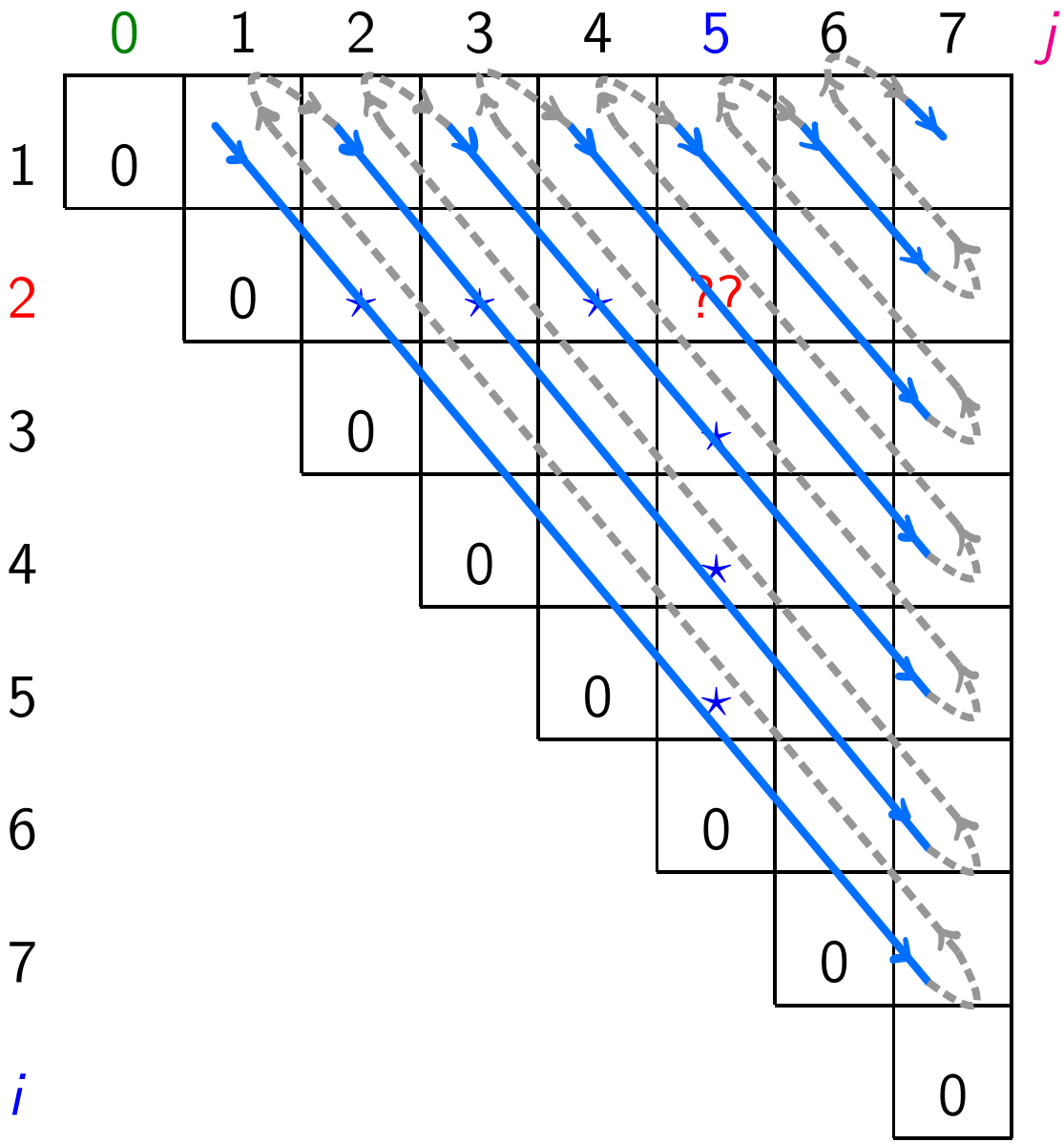
Como preencher a matriz c?
Em que ordem?

Como no problema da parentização! Pelas diagonais!

Programação dinâmica



Programação dinâmica

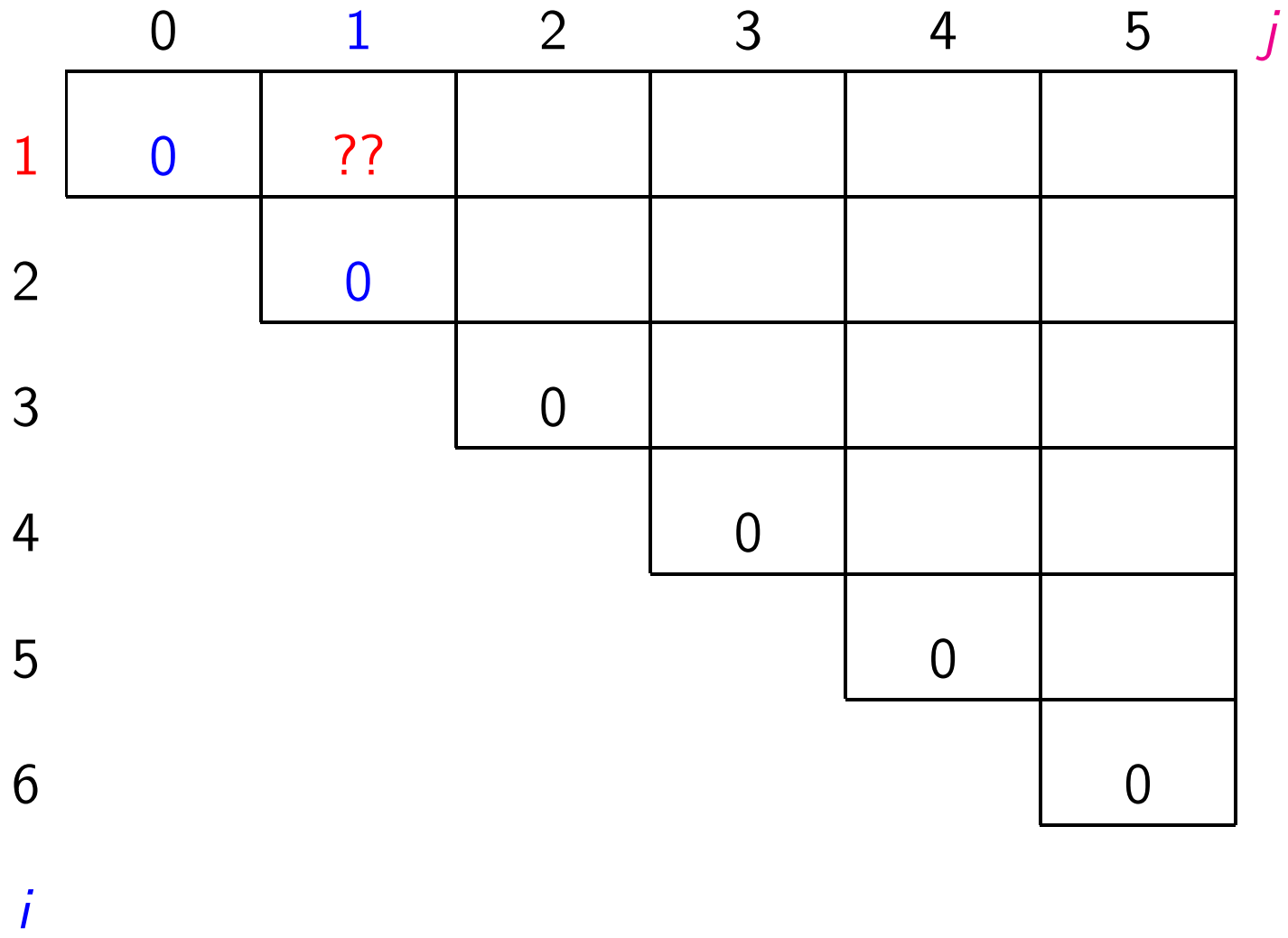


Simulação

$e[1]=10 \quad e[2]=20 \quad e[3]=30 \quad e[4]=15 \quad e[5]=30$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$



Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10				
2		0				
3			0			
4				0		
5					0	
6						0

$c[1,1-1] + e[1] + c[1+1,1] = 0+10+0 = 10$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10				
2		0	??			
3			0			
4				0		
5					0	
6						0

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10				
2		0	20			
3			0			
4				0		
5					0	
6						0

$$c[2,2-1] + e[2] + c[2+1,2] = 0+20+0 = 20$$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10				
2		0	20			
3			0	??		
4				0		
5					0	
6						0

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10				
2		0	20			
3			0	30		
4				0		
5					0	
6						0

$$c[3,3-1] + e[3] + c[3+1,3] = 0+30+0 = 30$$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10				
2		0	20			
3			0	30		
4				0	??	
5					0	
6						0

Simulação

$e[1]{=}10 \quad e[2]{=}20 \quad e[3]{=}30 \quad e[4]{=}15 \quad e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10				
2		0	20			
3			0	30		
4				0	15	
5					0	
6						0

$$c[4,4{+}1] + e[4] + c[4{+}1,4] = 0{+}15{+}0 = 15$$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10				
2		0	20			
3			0	30		
4				0	15	
5					0	??
6						0

Simulação

$e[1]{=}10 \quad e[2]{=}20 \quad e[3]{=}30 \quad e[4]{=}15 \quad e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10				
2		0	20			
3			0	30		
4				0	15	
5					0	30
6						0

$c[5,5{+}1] + e[5] + c[5{+}1,5] = 0{+}30{+}0 = 30$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10	??			
2		0	20			
3			0	30		
4				0	15	
5					0	30
6						0

Simulação

$e[1]{=}10 \quad e[2]{=}20 \quad e[3]{=}30 \quad e[4]{=}15 \quad e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	50			
2		0	20			
3			0	30		
4				0	15	
5					0	30
6						0

$$c[1,1{-}1] + (e[1] + e[2]) + c[1{+}1,2] = 0{+}30{+}20 = 50$$

Simulação

$e[1]{=}10 \quad e[2]{=}20 \quad e[3]{=}30 \quad e[4]{=}15 \quad e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	40			
2		0	20			
3			0	30		
4				0	15	
5					0	30
6						0

$$c[1, 2{-}1] + (e[1] + e[2]) + c[2{+}1, 2] = 10{+}30{+}0 = 40$$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10	40			
2		0	20	??		
3			0	30		
4				0	15	
5					0	30
6						0

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10	40			
2		0	20	80		
3			0	30		
4				0	15	
5					0	30
6						0

$$c[2,2-1] + (e[2] + e[3]) + c[2+1,3] = 0+50+30 = 80$$

Simulação

$e[1]{=}10$ $e[2]{=}20$ $e[3]{=}30$ $e[4]{=}15$ $e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	40			
2		0	20	70		
3			0	30		
4				0	15	
5					0	30
6						0

$$c[2, 3{-}1] + (e[2] + e[3]) + c[3{+}1, 3] = 20{+}50{+}0 = 70$$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10	40			
2		0	20	70		
3			0	30	??	
4				0	15	
5					0	30
6						0

Simulação

$e[1]{=}10 \quad e[2]{=}20 \quad e[3]{=}30 \quad e[4]{=}15 \quad e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	40			
2		0	20	70		
3			0	30	60	
4				0	15	
5					0	30
6						0

$$c[3, 3{-}1] + (e[3] + e[4]) + c[3{+}1, 4] = 0{+}45{+}15 = 60$$

Simulação

$e[1]{=}10 \quad e[2]{=}20 \quad e[3]{=}30 \quad e[4]{=}15 \quad e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	40			
2		0	20	70		
3			0	30	60	
4				0	15	
5					0	30
6						0

$$c[3, 4{-}1] + (e[3] + e[4]) + c[4{+}1, 4] = 30{+}45{+}0 = 75$$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10	40			
2		0	20	70		
3			0	30	60	
4				0	15	??
5					0	30
6						0

Simulação

$e[1]=10 \quad e[2]=20 \quad e[3]=30 \quad e[4]=15 \quad e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10	40			
2		0	20	70		
3			0	30	60	
4				0	15	75
5					0	30
6						0

$c[4,4-1] + (e[4] + e[5]) + c[4+1,5] = 0+45+30=75$

Simulação

$e[1]{=}10 \quad e[2]{=}20 \quad e[3]{=}30 \quad e[4]{=}15 \quad e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	40			
2		0	20	70		
3			0	30	60	
4				0	15	60
5					0	30
6						0

$$c[4,5{-}1] + (e[4] + e[5]) + c[5{+}1,5] = 15{+}45{+}0 = 60$$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10	40	??		
2		0	20	70		
3			0	30	60	
4				0	15	60
5					0	30
6						0

Simulação

$e[1]{=}10 \quad e[2]{=}20 \quad e[3]{=}30 \quad e[4]{=}15 \quad e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	40	130		
2		0	20	70		
3			0	30	60	
4				0	15	60
5					0	30
6						0

$$c[1,1{-}1] + (e[1] + e[2] + e[3]) + c[1{+}1,3] = 0{+}60{+}70 = 130$$

Simulação

$e[1]{=}10 \quad e[2]{=}20 \quad e[3]{=}30 \quad e[4]{=}15 \quad e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	40	100		
2		0	20	70		
3			0	30	60	
4				0	15	60
5					0	30
6						0

$$c[1, 2{-}1] + (e[1]) + e[2] + e[3]) + c[2{+}1, 3] = 10{+}60{+}30 = 100$$

Simulação

$e[1]{=}10 \quad e[2]{=}20 \quad e[3]{=}30 \quad e[4]{=}15 \quad e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	40	100		
2		0	20	70		
3			0	30	60	
4				0	15	60
5					0	30
6						0

$$c[1, 3{-}1] + (e[1] + e[2] + e[3]) + c[3{+}1, 3] = 40{+}60{+}0 = 100$$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10	40	100		
2		0	20	70	??	
3			0	30	60	
4				0	15	60
5					0	30
6						0

Simulação

$e[1]{=}10$ $e[2]{=}20$ $e[3]{=}30$ $e[4]{=}15$ $e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	40	100		
2		0	20	70	125	
3			0	30	60	
4				0	15	60
5					0	30
6						0

$$c[2, 2{-}1] + (e[2] + e[3] + e[4]) + c[2{+}1, 4] = 0{+}65{+}60 = 125$$

Simulação

$e[1]{=}10$ $e[2]{=}20$ $e[3]{=}30$ $e[4]{=}15$ $e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	40	100		
2		0	20	70	100	
3			0	30	60	
4				0	15	60
5					0	30
6						0

$c[2, 3{-}1] + (e[2] + e[3] + e[4]) + c[3{+}1, 4] = 20{+}65{+}15 = 100$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10	40	100		
2		0	20	70	100	
3			0	30	60	
4				0	15	60
5					0	30
6						0

$c[2,4-1] + (e[2]+e[3]+e[4]) + c[4+1,4] = 70+65+0 = 135$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10	40	100		
2		0	20	70	100	
3			0	30	60	??
4				0	15	60
5					0	30
6						0

Simulação

$e[1]{=}10 \quad e[2]{=}20 \quad e[3]{=}30 \quad e[4]{=}15 \quad e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	40	100		
2		0	20	70	100	
3			0	30	60	135
4				0	15	60
5					0	30
6						0

$$c[3,3{-}1] + (e[3] + e[4] + e[5]) + c[3{+}1,5] = 0{+}75{+}60 = 135$$

Simulação

$e[1]{=}10 \quad e[2]{=}20 \quad e[3]{=}30 \quad e[4]{=}15 \quad e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	40	100		
2		0	20	70	100	
3			0	30	60	135
4				0	15	60
5					0	30
6						0

$$c[3, 4{-}1] + (e[3] + e[4] + e[5]) + c[4{+}1, 5] = 30{+}75{+}30 = 135$$

Simulação

$e[1]{=}10 \quad e[2]{=}20 \quad e[3]{=}30 \quad e[4]{=}15 \quad e[5]{=}30$

i \ j	0	1	2	3	4	5
1	0	10	40	100		
2		0	20	70	100	
3			0	30	60	135
4				0	15	60
5					0	30
6						0

$$c[3,5{-}1] + (e[3] + e[4] + e[5]) + c[5{+}1,5] = 60{+}75{+}0 = 135$$

Simulação

$e[1]=10$ $e[2]=20$ $e[3]=30$ $e[4]=15$ $e[5]=30$

i \ j	0	1	2	3	4	5
1	0	10	40	100		
2		0	20	70	100	
3			0	30	60	135
4				0	15	60
5					0	30
6						0

Exercício: Preencha o que falta!

Árvore de busca ótima

```
ABB-Ótima (e, n)
 1  s[0] = 0
 2  para i ← 1 até n faça
 3      s[i] ← s[i−1] + e[i]
 4  para i ← 1 até n+1 faça
 5      c[i, i−1] ← 0
 6  para ℓ ← 1 até n faça
 7      para i ← 1 até n−ℓ+1 faça
 8          j ← i+ℓ−1
 9          c[i, j] ← c[i+1, j]
10          para k ← i+1 até j faça
11              se c[i, k−1] + c[k+1, j] < c[i, j]
12                  então c[i, j] ← c[i, k−1] + c[k+1, j]
13          c[i, j] ← c[i, j] + s[j] − s[i−1]
14  devolva c[1, n]
```

Exercício

Como fazer para obter uma ABB ótima
e não apenas o seu custo?

Complete o serviço!