

# 課題 3

# 「文字列処理と動的計画法」

---

井之上 直也, 吉留 崇

2016年度プログラミング演習A

# 本課題で学ぶこと

## ■ 編集距離

◆ 文字列処理



課題 1

◆ 再帰的手続き



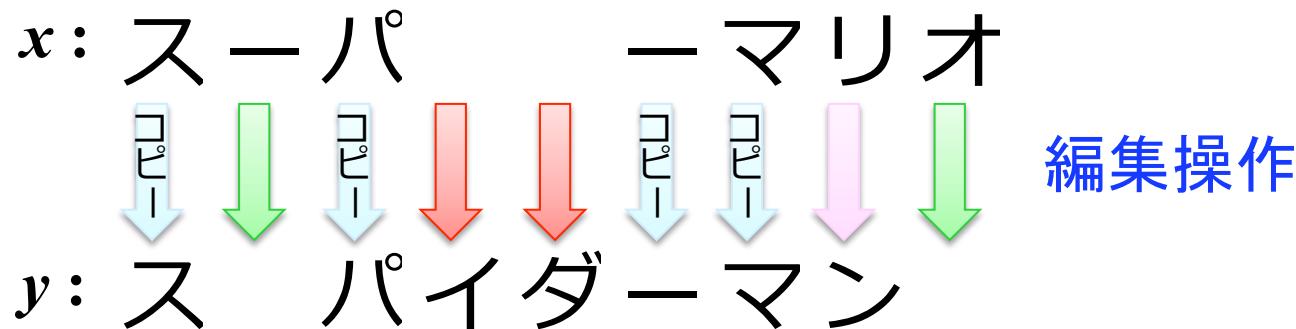
課題2

◆ 動的計画法

- ・ 問題3-0: 編集距離の解説
- ・ 問題内にも解説あり

# 編集距離 (Edit Distance)

2つの文字列がどのくらい似ているかを表す指標



文字列  $x$  を文字列  $y$  に変換する時の「削除」，「挿入」，  
「置換」の最小回数（「コピー」はカウントしない）

今回の例の編集距離 = 5

スペルチェッカーやDNAの配列の解析に応用

# 文字列に対する数学記号

	1	2	3	4	5	6	7	8	9
$x$ :	a	b	b	a	b	a	a	a	b

文字列  $x$  の長さ :  $|x|$       例)  $|x| = 9$

文字列  $x$  の  $i$  番目の文字 :  $x_i$  または  $x[i]$       例)  $x_4 = a$

文字列  $x$  の先頭から  $i$  番目までの文字列 :  $X_i$        接頭辞  
例)  $X_4 = abba$

長さ 0 の文字列 :  $\varepsilon$

# 編集距離の定義

文字列  $x = \langle x_1, x_2, \dots, x_m \rangle$ ,  $y = \langle y_1, y_2, \dots, y_n \rangle$  の接頭辞  $X_i$ ,  $Y_j$  の編集距離:  $c_{i,j}$

$$c_{i,j} = \begin{cases} \max(i, j) & (i = 0 \text{ または } j = 0 \text{ の時}) \\ \min(c_{i-1, j-1} + \delta(x_i, y_j), \\ \quad c_{i-1, j} + 1, \\ \quad c_{i, j-1} + 1) & (\text{その他}) \end{cases}$$

$$\delta(x_i, y_j) = \begin{cases} 1 & (x_i \neq y_j) \\ 0 & (x_i = y_j) \end{cases}$$

# 編集距離の計算方法

$$c_{i,j} < X_i, Y_j > = \min(c_{i-1,j-1} + \delta(x_i, y_j), \begin{cases} \text{置換} & (x_i \neq y_j) \\ c_{i-1,j} + 1, & \text{削除} \\ c_{i,j-1} + 1, & \text{挿入} \end{cases} \begin{cases} \text{コピ} & (x_i = y_j) \end{cases})$$

$$c_{i,0} < X_i, \varepsilon > = i, \quad \text{削除} \quad c_{0,j} < \varepsilon, Y_j > = j, \quad \text{挿入}$$

$$\delta(x_i, y_j) = \begin{cases} 1 & (x_i \neq y_j) \\ 0 & (x_i = y_j) \end{cases}$$

再帰関数による実装

動的計画法による実装

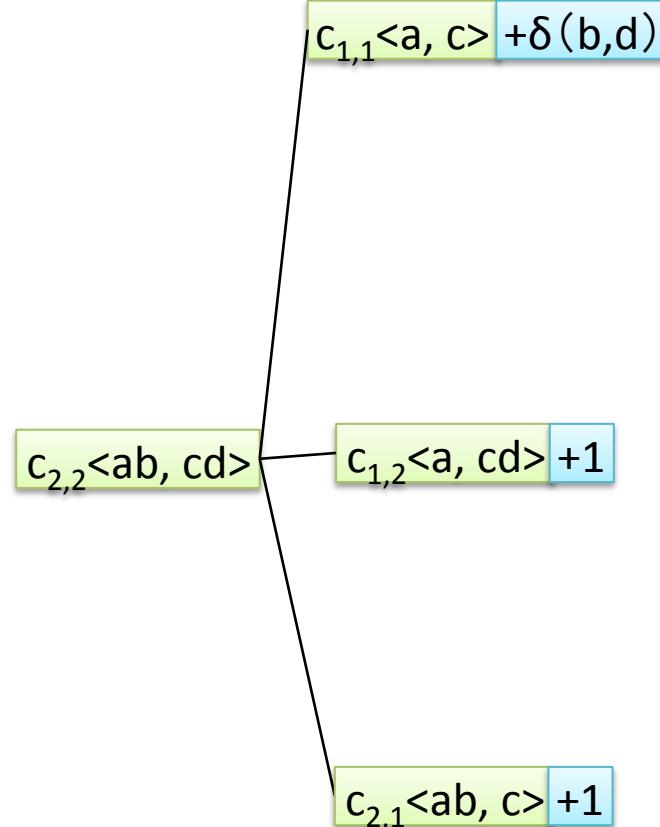
# 再帰的に計算

c<sub>2,2</sub><ab, cd>

$$c_{i,j} < X_i, Y_j > = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} < X_i, \varepsilon > = i, \quad c_{0,j} < \varepsilon, Y_j > = j$$

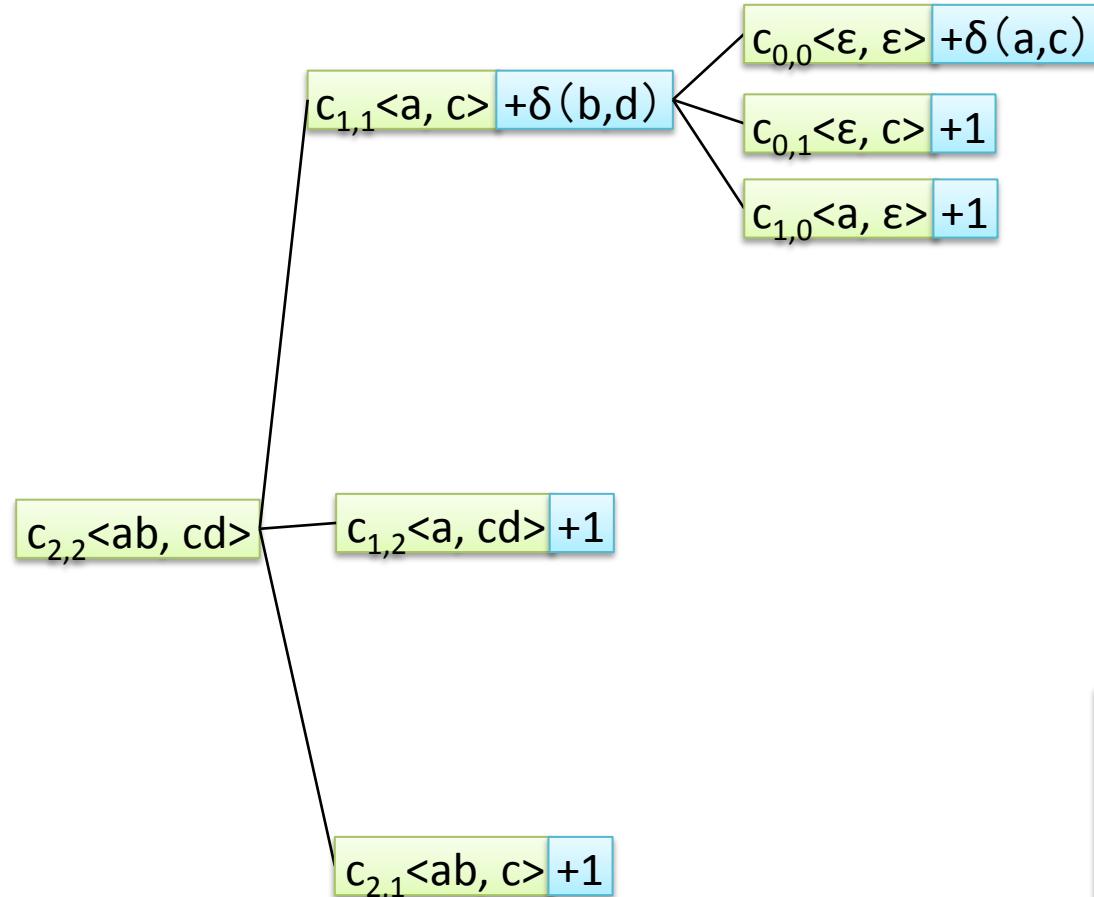
# 再帰的に計算



$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \varepsilon> = i, \quad c_{0,j} <\varepsilon, Y_j> = j$$

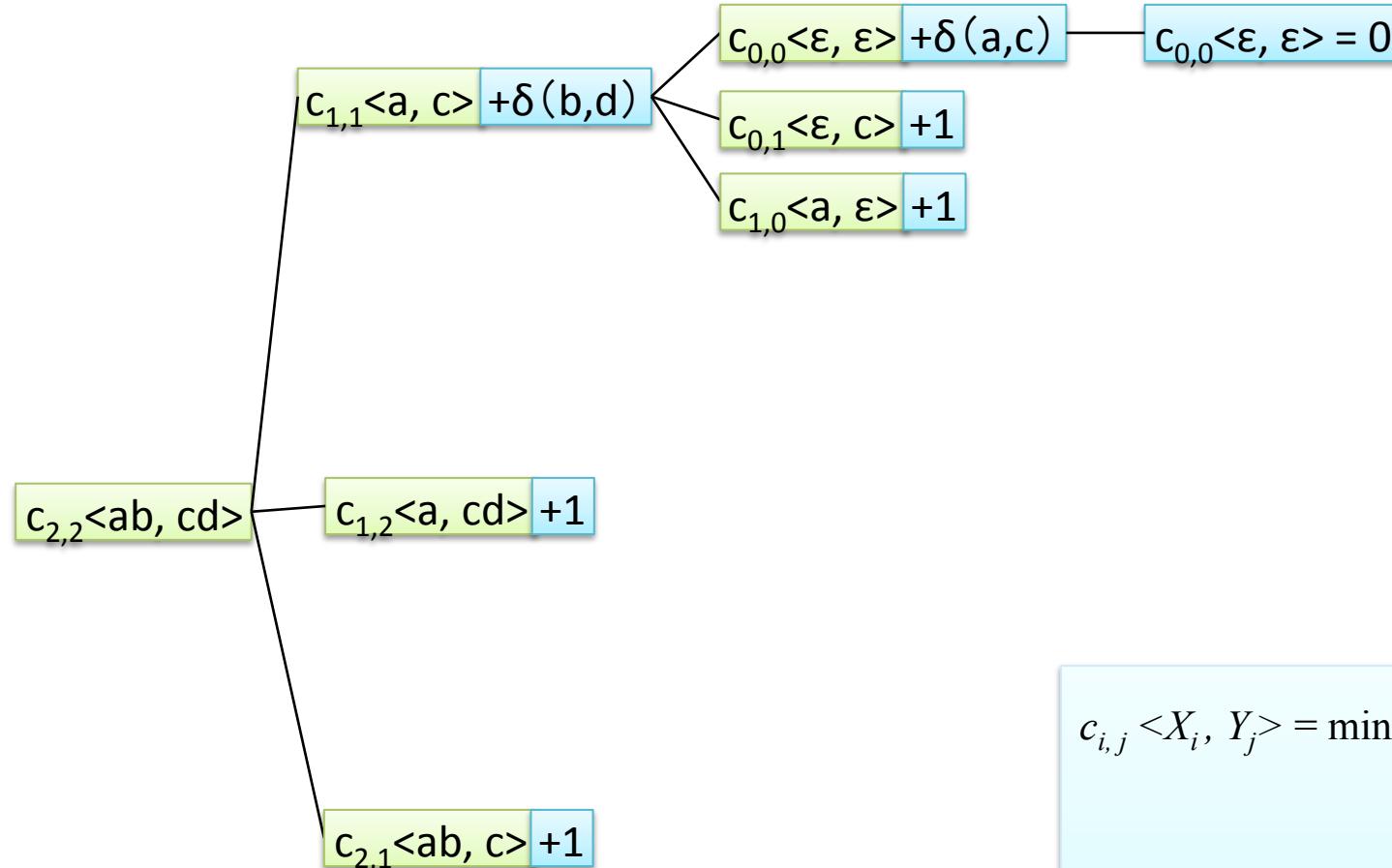
# 再帰的に計算



$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \epsilon> = i, \quad c_{0,j} <\epsilon, Y_j> = j$$

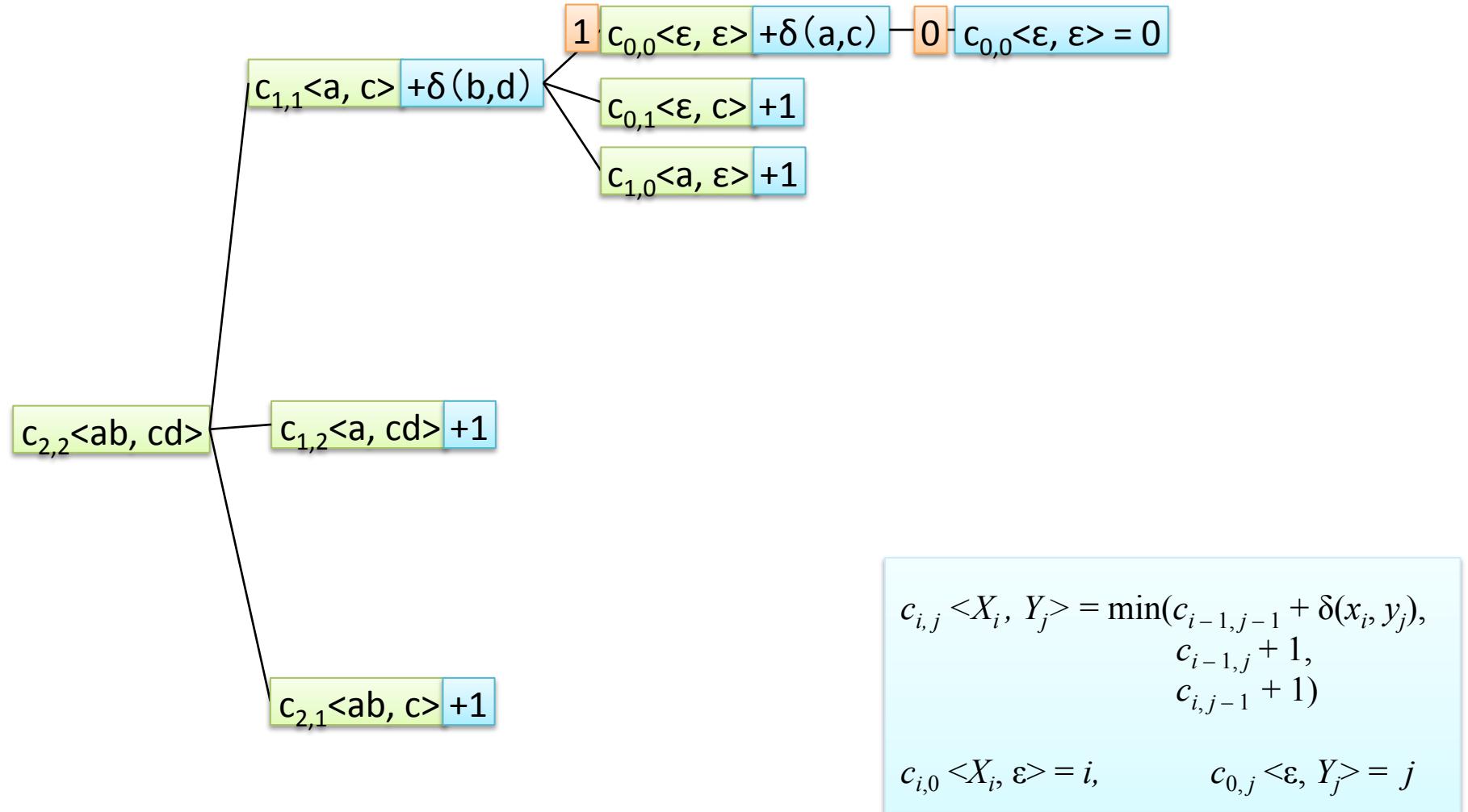
# 再帰的に計算



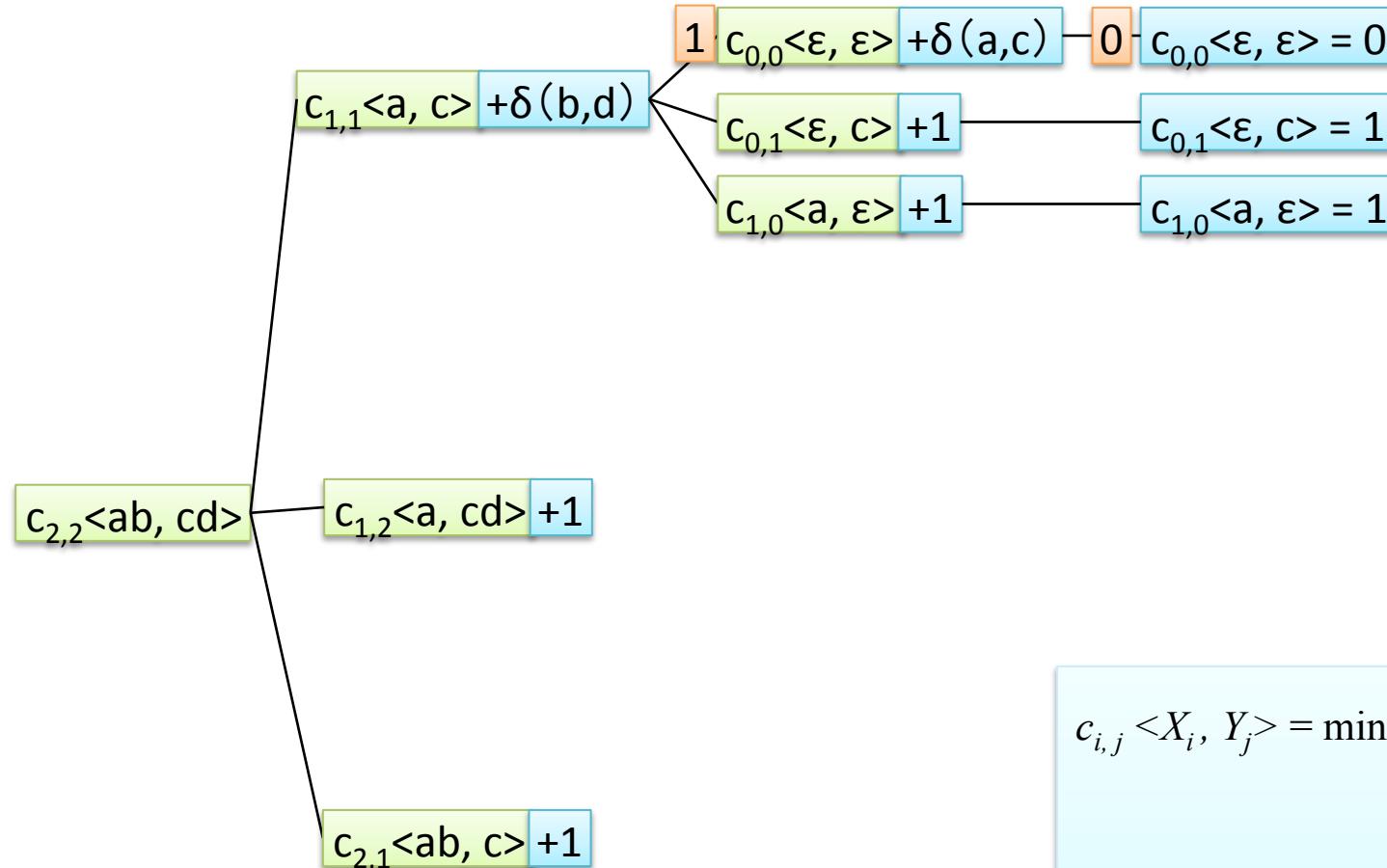
$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \epsilon> = i, \quad c_{0,j} <\epsilon, Y_j> = j$$

# 再帰的に計算



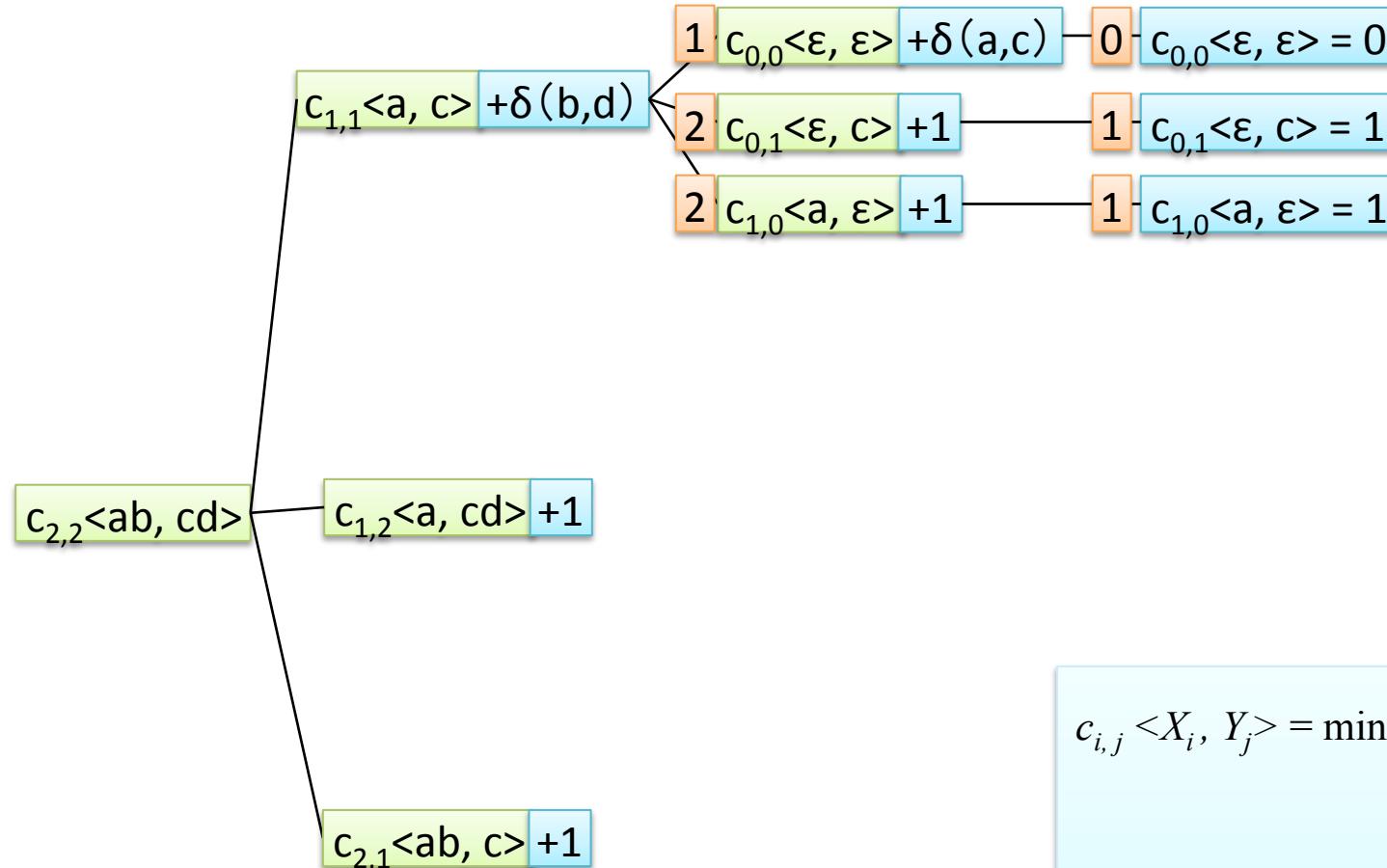
# 再帰的に計算



$$c_{i,j} < X_i, Y_j > = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} < X_i, \varepsilon > = i, \quad c_{0,j} < \varepsilon, Y_j > = j$$

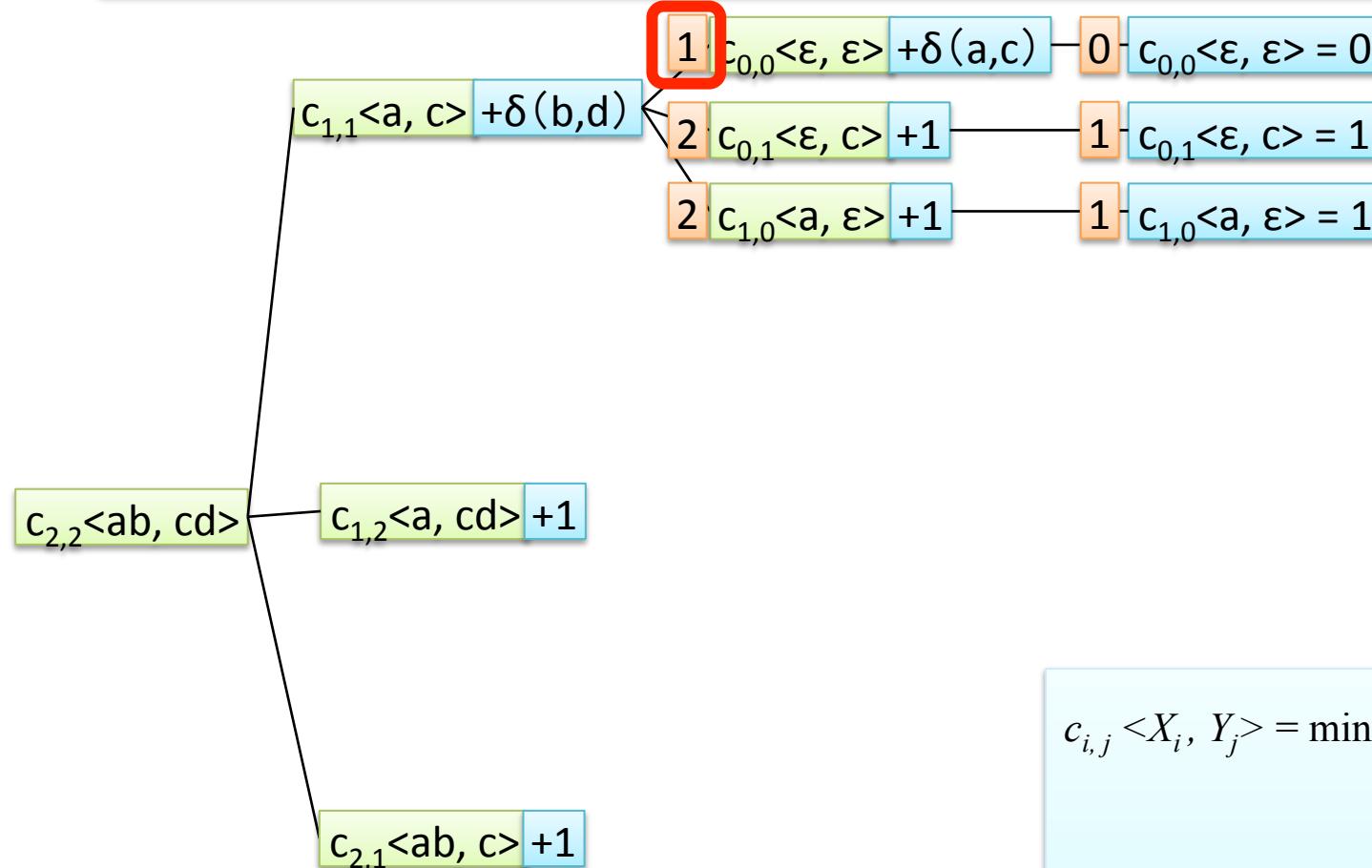
# 再帰的に計算



$$c_{i,j} < X_i, Y_j > = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} < X_i, \varepsilon > = i, \quad c_{0,j} < \varepsilon, Y_j > = j$$

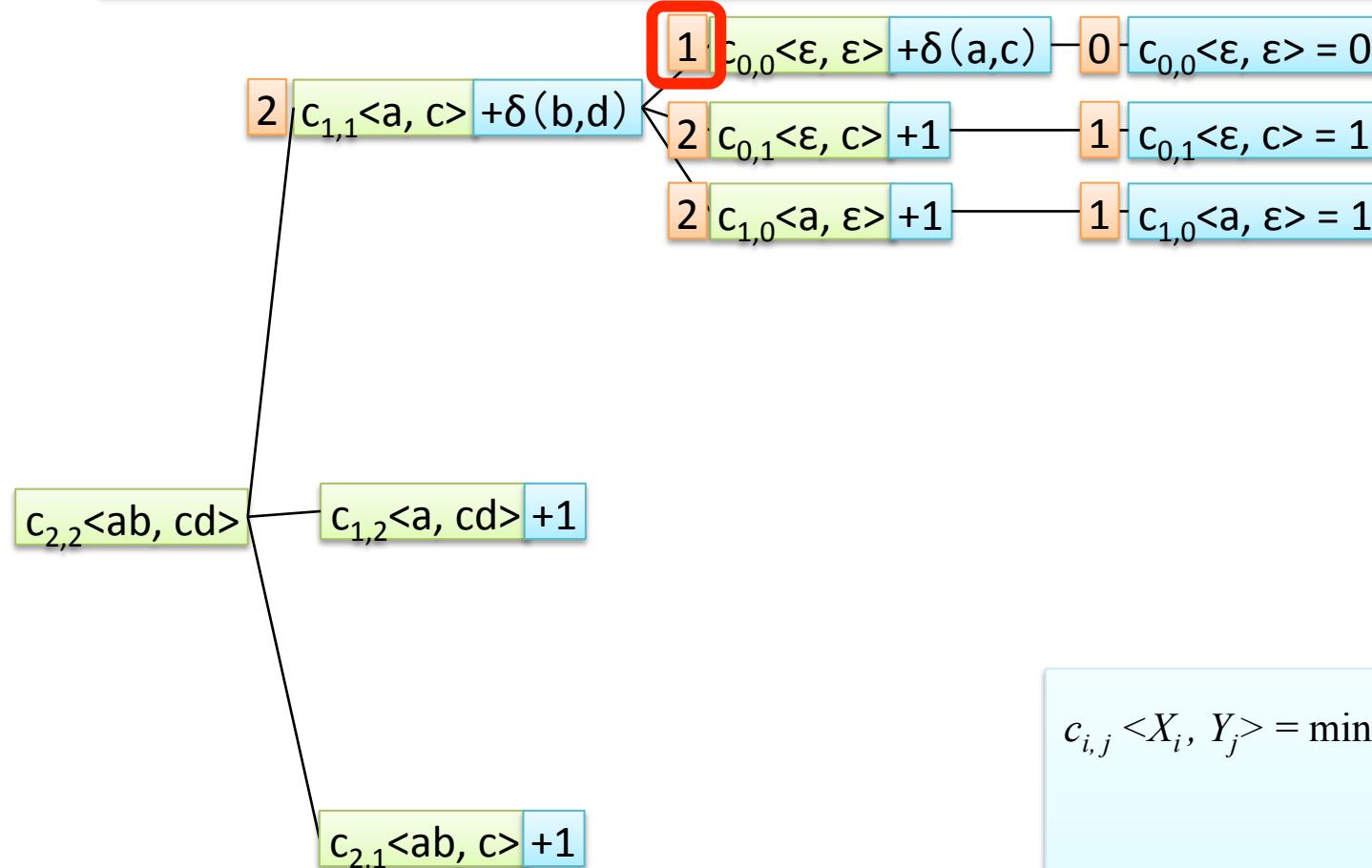
# 再帰的に計算



$$c_{i,j} < X_i, Y_j > = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} < X_i, \varepsilon > = i, \quad c_{0,j} < \varepsilon, Y_j > = j$$

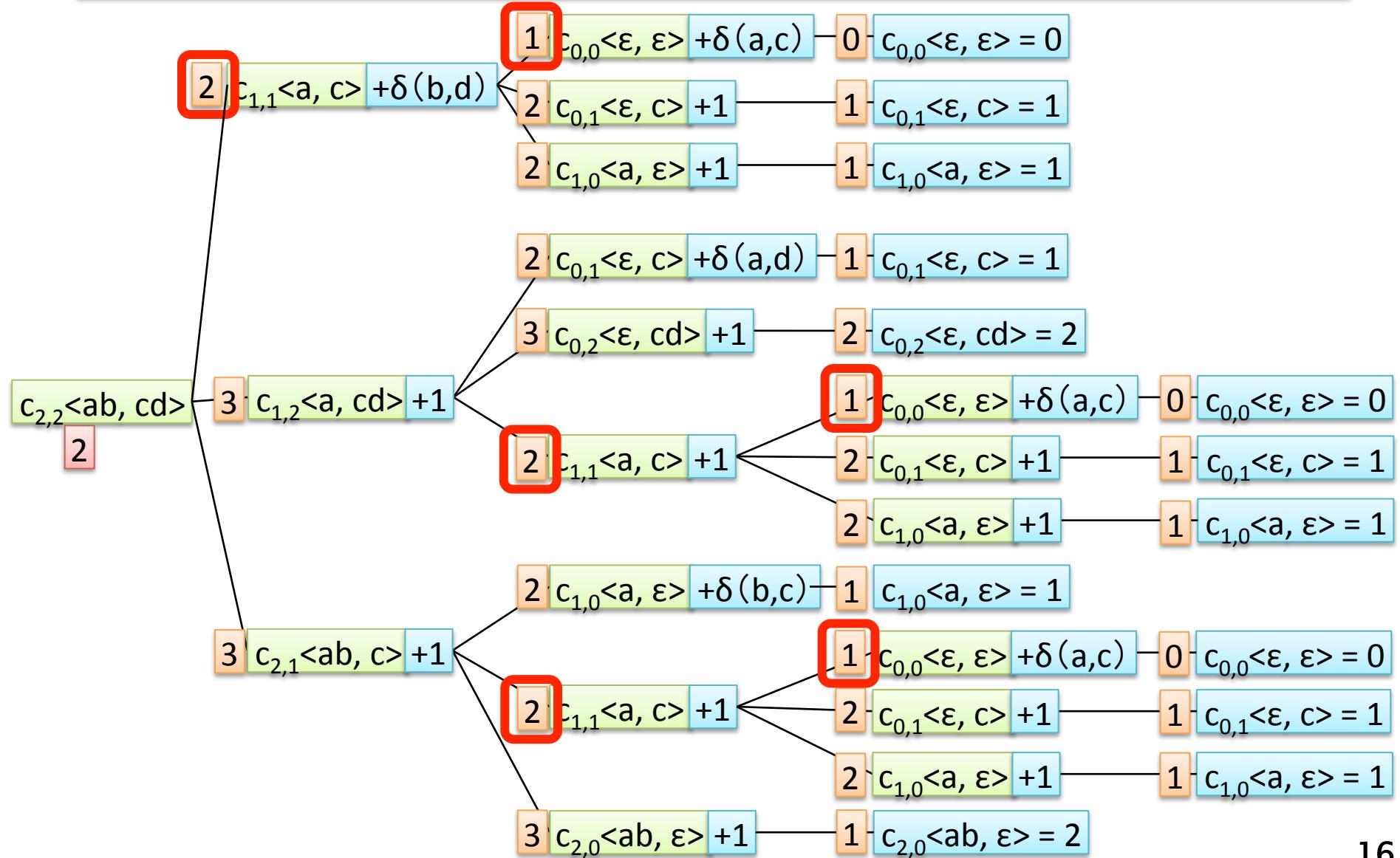
# 再帰的に計算



$$c_{i,j} < X_i, Y_j > = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} < X_i, \epsilon > = i, \quad c_{0,j} < \epsilon, Y_j > = j$$

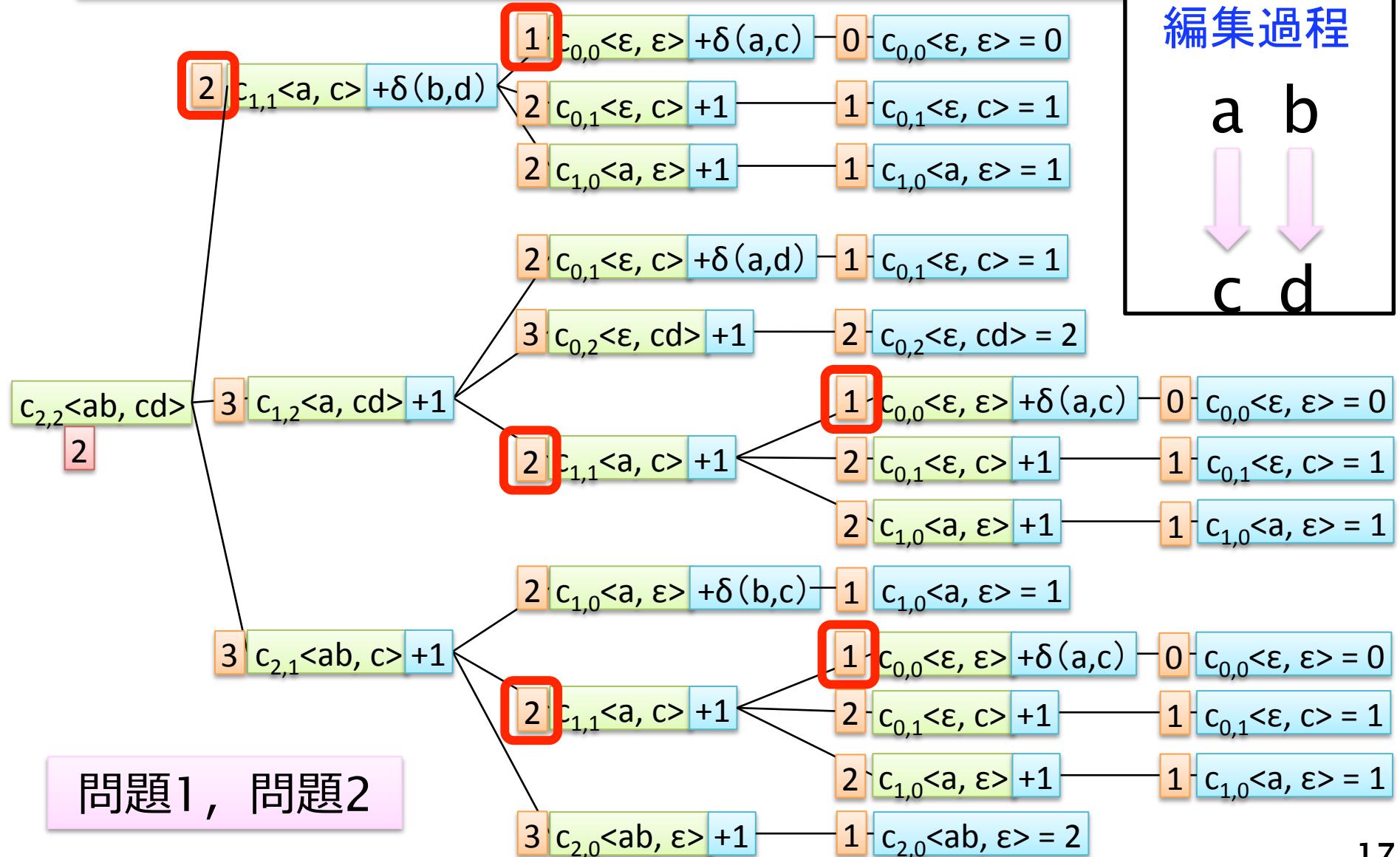
# 再帰的に計算



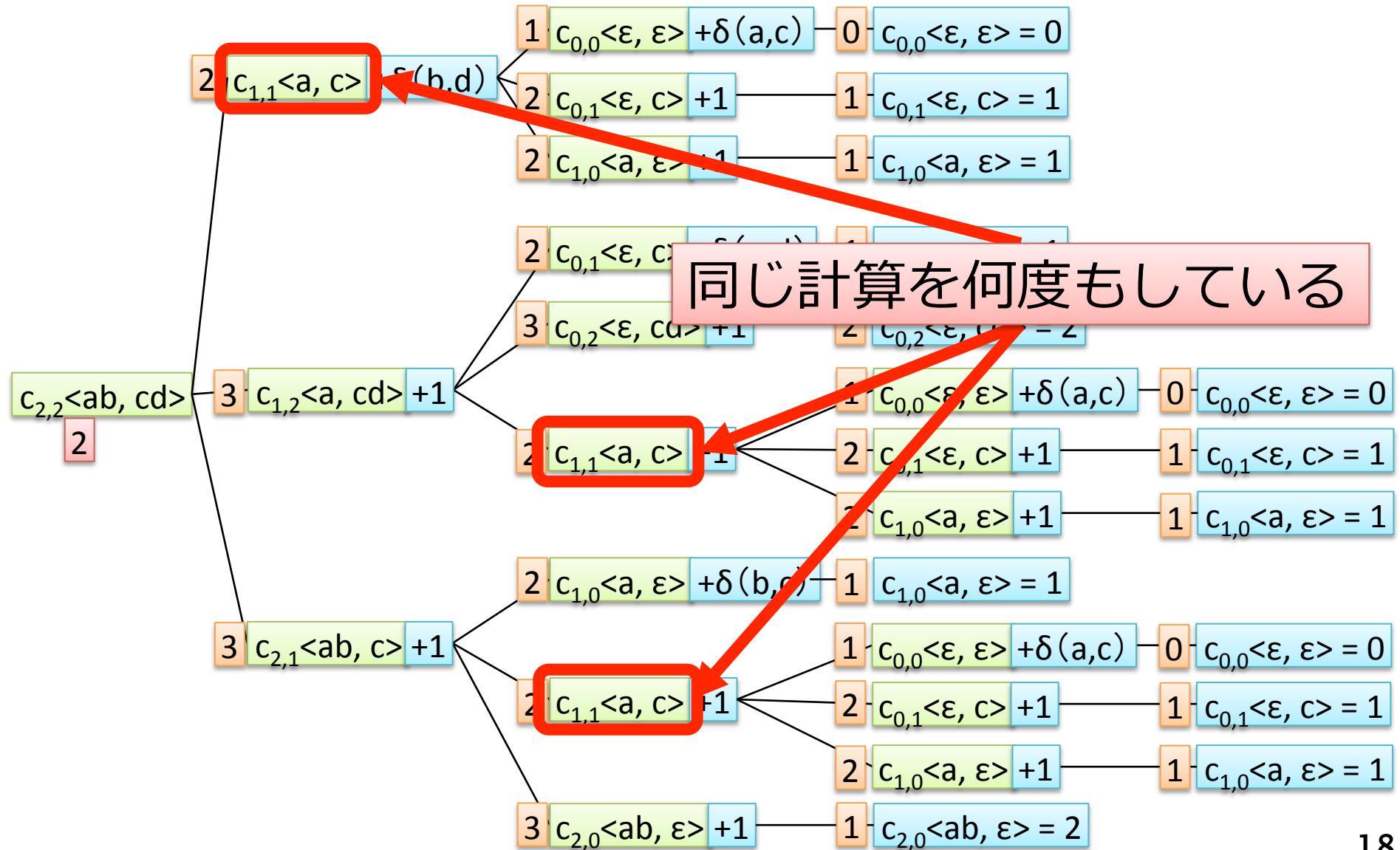
# 再帰的に計算

編集距離を  
与える  
編集過程

a b  
 ↓ ↓  
 c d



# 再帰的計算の問題点



# 再帰的計算の問題点

- 同じ計算を何度も行うため計算量大

→ メモ化 (memoization) による対策

一度計算したものをメモとして記憶、次に計算するときはそのメモを呼び出す

問題5

# 動的計画法(Dynamic Programming, DP)

$c_{i,j} <ab, cd>$

	$\varepsilon$	c	d
$\varepsilon$	$C_{0,0}$	$C_{0,0}$	$C_{0,2}$
a	$C_{1,0}$	$C_{1,1}$	$C_{1,2}$
b	$C_{2,0}$	$C_{2,1}$	$C_{2,2}$

$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \varepsilon> = i, c_{0,j} <\varepsilon, Y_j> = j$$

$C_{0,0}, C_{0,1}, C_{1,0}$ が計算済み  
ならば、 $C_{1,1}$ の値分かる

# 動的計画法(Dynamic Programming, DP)

$c_{i,j} <ab, cd>$

	$\varepsilon$	c	d
$\varepsilon$	$C_0,$ 0 1	$C_0,$ 1	$C_0,$ 2
a	$C_1,$ 0 1	$C_1,$ 1	$C_1,$ 2
b	$C_2,$ 0 1	$C_2,$ 1	$C_2,$ 2

$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j),$$
$$c_{i-1,j} + 1,$$
$$c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \varepsilon> = i, c_{0,j} <\varepsilon, Y_j> = j$$

$C_{0,1}, C_{0,2}, C_{1,1}$ が計算済み  
ならば、 $C_{1,2}$ の値分かる

# 動的計画法(Dynamic Programming, DP)

$c_{i,j} <ab, cd>$

	$\varepsilon$	c	d
$\varepsilon$	$C_0,$ 0	$C_0,$ 1	$C_0,$ 2
a	$C_1,$ 0	$C_1,$ 1	$C_1,$ 2
b	$C_2,$ 0	$C_2,$ 1	$C_2,$ 2

$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j),$$

$$c_{i-1,j} + 1,$$

$$c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \varepsilon> = i, c_{0,j} <\varepsilon, Y_j> = j$$

$C_{1,1}, C_{1,2}, C_{2,1}$ が計算済み  
ならば、 $C_{2,2}$ の値分かる

# 動的計画法(Dynamic Programming, DP)

$c_{i,j} <ab, cd>$

	$\epsilon$	c	d
$\epsilon$	$C_0, 0$	$C_0, 1$	$C_0, 2$
a	$C_1, 0$	$C_1, 1$	$C_1, 2$
b	$C_2, 0$	$C_2, 1$	$C_2, 2$

$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \epsilon> = i, c_{0,j} <\epsilon, Y_j> = j$$

ボトムアップに  $C_{i,j}$  を求める  
↓

編集距離  $C_{2,2}$  得られる

同じ計算は1度のみ  $\Rightarrow$  計算量減

# 動的計画法による計算

$c_{i,j} <ab, cd>$

	$\varepsilon$	c	d
$\varepsilon$			
a			
b			

$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \varepsilon> = i, c_{0,j} <\varepsilon, Y_j> = j$$

# 動的計画法による計算

$c_{i,j} <ab, cd>$

	$\varepsilon$	c	d
$\varepsilon$	0  1  2		
a			
b			

$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \varepsilon> = i, c_{0,j} <\varepsilon, Y_j> = j$$

$$c_{0,0} <\varepsilon, \varepsilon> = 0, c_{0,1} <\varepsilon, c> = 1, c_{0,2} <\varepsilon, cd> = 2$$

# 動的計画法による計算

$c_{i,j} <ab, cd>$

	$\varepsilon$	c	d
$\varepsilon$	0	1	2
a	1		
b	2		

$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \varepsilon> = i, c_{0,j} <\varepsilon, Y_j> = j$$

$$c_{0,0} <\varepsilon, \varepsilon> = 0, c_{1,0} <a, \varepsilon> = 1, c_{2,0} <ab, \varepsilon> = 2$$

# 動的計画法による計算

$$c_{i,j} <ab, cd>$$

		c	d
ε	0	1	$1+1=2$
a	1	1	
b	2		

Diagram illustrating the computation of  $c_{i,j} <ab, cd>$  using dynamic programming. The grid shows values for different pairs of symbols. A red circle highlights 'a' at position (0,0). An orange callout box at (0,1) shows the calculation  $0 + \delta(a, c) = 0 + 1 = 1$ . Another orange callout box at (1,1) shows  $1 + 1 = 2$ . Red arrows point from (0,0) to (0,1) and from (0,1) to (1,1).

$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \varepsilon> = i, c_{0,j} <\varepsilon, Y_j> = j$$

$$c_{1,1} <a, c> = \min(c_{0,0} <\varepsilon, \varepsilon> + \delta(a, c), \\ c_{0,1} <\varepsilon, c> + 1, \\ c_{1,0} <a, \varepsilon> + 1)$$

# 動的計画法による計算

$c_{i,j} <ab, cd>$

		$\varepsilon$	$d$
$\varepsilon$	0	1	2
$a$	1	1	2
$b$	2		

Diagram illustrating the computation of  $c_{i,j} <ab, cd>$  using dynamic programming. The grid shows values for different pairs of strings. An orange callout box at the top right of the 'd' cell contains the calculation  $1 + \delta(a, d) = 1 + 1 = 2$ . Arrows point from this box to the 'd' cell and the '1' cell in the row below it. Another orange callout box at the bottom left of the '1' cell contains the calculation  $1 + 1 = 2$ , with arrows pointing to the '1' cell and the '2' cell in the row above it.

$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j), c_{i-1,j} + 1, c_{i,j-1} + 1)$$

$$2+1=3 <X_i, \varepsilon> = i, c_{0,j} <\varepsilon, Y_j> = j$$

$$c_{1,2} <a, cd> = \min(c_{0,1} <\varepsilon, c> + \delta(a, d), c_{0,2} <\varepsilon, cd> + 1, c_{1,1} <a, c> + 1)$$

# 動的計画法による計算

$c_{i,j} <ab, cd>$

	$\varepsilon$	$c$	d
$\varepsilon$	$1 + \delta(b, c) = 1 + 1 = 2$	1	2
a	1	1	$1 + 1 = 2$
b	2	2	

$2 + 1 = 3$

$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \varepsilon> = i, c_{0,j} <\varepsilon, Y_j> = j$$

$$c_{2,1} <ab, c> = \min(c_{1,0} <a, \varepsilon> + \delta(b, c), \\ c_{1,1} <a, c> + 1, \\ c_{2,0} <ab, \varepsilon> + 1)$$

# 動的計画法による計算

$c_{i,j} <ab, cd>$

	$\varepsilon$	c	d
$\varepsilon$	0	$1+\delta(b,d)$ $= 1+1=2$	2
a	1	1	2
b	2	2	2

2+1=3

$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \varepsilon> = i, c_{0,j} <\varepsilon, Y_j> = j$$

2+1=3

$$c_{2,2} <ab, cd> = \min(c_{1,1} <a, c> + \delta(b, d), \\ c_{1,2} <a, cd> + 1, \\ c_{2,1} <ab, c> + 1)$$

# 動的計画法による計算

$c_{i,j} <ab, cd>$

	$\varepsilon$	c	d
$\varepsilon$	0	1	2
a	1	1	2
b	2	2	2

$$c_{i,j} <X_i, Y_j> = \min(c_{i-1,j-1} + \delta(x_i, y_j), \\ c_{i-1,j} + 1, \\ c_{i,j-1} + 1)$$

$$c_{i,0} <X_i, \varepsilon> = i, c_{0,j} <\varepsilon, Y_j> = j$$



ここが編集距離

問題3, 問題4

# 動的計画法による計算

$c_{i,j} <ab, cd>$

	$\epsilon$	c	d
$\epsilon$	0	1	2
a	1	1	2
b	2	2	2

編集操作の求め方

値を埋める際に使ったデータ  
の流れの矢印を表示

# 動的計画法による計算

$c_{i,j} <ab, cd>$

	$\epsilon$	c	d
$\epsilon$	0 ←↑ 1 ←↑ 2		
a	1 ↑ 1 ← 2		
b	2 2 2		

編集操作の求め方

矢印を180°回転

矢印の意味 :

← 挿入 (insert: I)

↑ 削除 (delete: D)

↖ 置換 (replace: R) ( $x_i \neq y_j$  のとき)  
↖ コピー (copy: =) ( $x_i = y_j$  のとき)

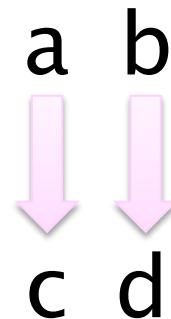
# 動的計画法による計算

$c_{i,j} <ab, cd>$

	$\epsilon$	c	d
$\epsilon$	0	1	2
a	1	1	2
b	2	2	2

編集操作の求め方

右下から0の部分までたどって  
いったときの矢印が編集操作  
(ただし、編集操作が1つに決  
まるとは限らない)



R R

問題6

# 課題3：全部で7問

- 問題1 (10点) [記述問題]再帰による編集距離の計算 (手書き可)
- 問題2 (10点) 再帰関数による実装
- 問題3 (10点) [記述問題]動的計画法による編集距離の計算 (手書き可)
- 問題4 (10点) 動的計画法による実装
- 問題5 (20点) [プログラム+記述問題]再帰呼び出しにおけるメモ化
- 問題6 (10点) 編集操作の表示
- 問題7 (10点) スペル訂正器の作成

2週目の面接  
のみ受付

# 記述問題：問題1および問題3

---

- 本日説明した編集距離を求める方法についてレポートにまとめる

## 問題1：再帰的な方法

- ◆ 途中経過を省略してはいけない（接頭辞の編集距離をすべて求める、添字 ( $C_{i,j}$  の  $i,j$  のこと) をすべて書く、編集操作が分かるようにする）
- ◆ 編集操作を説明できるようにすること
- ◆ 編集距離を与える編集過程をすべて示す

## 問題3：動的計画法

- ◆ 表形式でレポートにまとめること

# プログラム + 記述問題：問題5

---

## ■ 再帰関数の呼び出しのメモ化

### レポート

- ◆ 再帰呼び出しの回数をgnuplotを用いてグラフにすること（折れ線グラフ）
- ◆ メモ化の有無によって呼び出し回数がどのようになるか考察すること

# 面接を受けるときの注意事項

---

- C言語の用語や関数の動作をあらかじめ調べてくること  
プログラムのソースコードに説明を書いてかまわないので,面接時に説明できるようにすること
- 課題に関する問題文をきちんと読むこと  
問題文を読むことが解答への近道になることもあります
- 2週目の面接時に,途中だったとしても,取り組んでいる問題を持ってきましょう  
困っていることやわからないことがあれば, 面接の時にアドバイスをするので“, 途中のプログラムでも遠慮なく持ってきてましょう