# Edit distance

- An **alignment**, or matched up, of two strings is simply a way of writing the strings one above the other.

# Edit distance

- An **alignment**, or matched up, of two strings is simply a way of writing the strings one above the other.

  *example: alignments of "*`SNOWY`*" and "*`SUNNY`*":*

  ```
  s - n o w y        - s n o w - y
  s u n n - y        s u n - - n y
  ```

  *"–" indicates a "gap"*

# Edit distance

- An **alignment**, or matched up, of two strings is simply a way of writing the strings one above the other.

  *example: alignments of "SNOWY" and "SUNNY":*

  ```
  s - n o w y          - s n o w - y
  s u n n - y          s u n - - n y
  ```

  *"–" indicates a "gap"*

- The **cost** of an alignment is the number of columns in which the letters differ.

# Edit distance

- An **alignment**, or matched up, of two strings is simply a way of writing the strings one above the other.

  *example: alignments of "*`SNOWY`*" and "*`SUNNY`*":*

  ```
  s - n o w y          - s n o w - y
  s u n n - y          s u n - - n y
  ```

  *"–" indicates a "gap"*

- The **cost** of an alignment is the number of columns in which the letters differ.

  *example: alignments of "*`SNOWY`*" and "*`SUNNY`*":*

  ```
  s - n o w y          - s n o w - y
  s u n n - y          s u n - - n y
   cost = 3              cost = 5
  ```

# Edit distance

- An **alignment**, or matched up, of two strings is simply a way of writing the strings one above the other.

  *example: alignments of "*`SNOWY`*" and "*`SUNNY`*":*

```
s - n o w y          - s n o w - y
s u n n - y          s u n - - n y
```

  *"–" indicates a "gap"*

- The **cost** of an alignment is the number of columns in which the letters differ.

  *example: alignments of "*`SNOWY`*" and "*`SUNNY`*":*

```
s - n o w y          - s n o w - y
s u n n - y          s u n - - n y
 cost = 3              cost = 5
```

- **Edit distance** between two strings is the **minimum cost** of their alignment, i.e., *the best possible alignment*

# Edit distance

- An **alignment**, or matched up, of two strings is simply a way of writing the strings one above the other.

  *example: alignments of "*`SNOWY`*" and "*`SUNNY`*":*

  ```
  s - n o w y          - s n o w - y
  s u n n - y          s u n - - n y
  ```

  *"–" indicates a "gap"*

- The **cost** of an alignment is the number of columns in which the letters differ.

  *example: alignments of "*`SNOWY`*" and "*`SUNNY`*":*

  ```
  s - n o w y          - s n o w - y
  s u n n - y          s u n - - n y
   cost = 3              cost = 5
  ```

- **Edit distance** between two strings is the **minimum cost** of their alignment, i.e., *the best possible alignment*
- Edit distance is the minimum number of *edits* – insertions, deletions and substitutions of characters – need to transform the first string into the second.

# Edit distance

- An **alignment**, or matched up, of two strings is simply a way of writing the strings one above the other.

  *example: alignments of "*`SNOWY`*" and "*`SUNNY`*":*

  ```
  s - n o w y        - s n o w - y
  s u n n - y        s u n - - n y
  ```

  *"–" indicates a "gap"*

- The **cost** of an alignment is the number of columns in which the letters differ.

  *example: alignments of "*`SNOWY`*" and "*`SUNNY`*":*

  ```
  s - n o w y        - s n o w - y
  s u n n - y        s u n - - n y
   cost = 3            cost = 5
  ```

- **Edit distance** between two strings is the **minimum cost** of their alignment, i.e., *the best possible alignment*
- Edit distance is the minimum number of *edits* – insertions, deletions and substitutions of characters – need to transform the first string into the second. *e.g. a spell checker.*

# Edit distance

- Given strings $x[1 \cdots m]$ and $y[1 \cdots n]$.

# Edit distance

- Given strings $x[1 \cdots m]$ and $y[1 \cdots n]$. Define

$$e(m, n) = \text{the edit distance between } x \text{ and } y$$

# Edit distance

- Given strings $x[1 \cdots m]$ and $y[1 \cdots n]$. Define

$$e(m, n) = \text{the edit distance between } x \text{ and } y$$

Our objective is to compute $e(m, n)$ efficiently

# Edit distance

- Given strings $x[1 \cdots m]$ and $y[1 \cdots n]$. Define

$$e(m, n) = \text{the edit distance between } x \text{ and } y$$

  Our objective is to compute $e(m, n)$ efficiently
- *Subproblem*:

# Edit distance

- Given strings $x[1 \cdots m]$ and $y[1 \cdots n]$. Define

$$e(m, n) = \text{the edit distance between } x \text{ and } y$$

  Our objective is to compute $e(m, n)$ efficiently
- *Subproblem*:

$$\text{edit distance } e(i, j) \text{ between } x[1 \cdots i] \text{ and } y[1 \cdots j]$$

# Edit distance

- Given strings $x[1 \cdots m]$ and $y[1 \cdots n]$. Define

$$e(m, n) = \text{the edit distance between } x \text{ and } y$$

  Our objective is to compute $e(m, n)$ efficiently
- *Subproblem*:

$$\text{edit distance } e(i, j) \text{ between } x[1 \cdots i] \text{ and } y[1 \cdots j]$$

- How to express $e(i, j)$ in terms of its subproblems, *recursively*?

# Edit distance

- Given strings $x[1 \cdots m]$ and $y[1 \cdots n]$. Define

$$e(m, n) = \text{the edit distance between } x \text{ and } y$$

Our objective is to compute $e(m, n)$ efficiently

- *Subproblem*:

edit distance $e(i, j)$ between $x[1 \cdots i]$ and $y[1 \cdots j]$

- How to express $e(i, j)$ in terms of its subproblems, *recursively*?
- **key observation:** the rightmost column of an alignment of $x[1 \cdots i]$ and $y[1 \cdots j]$ can only be one of the following three cases:

| Case 1 | | Case 2 | | Case 3 |
|---|---|---|---|---|
| $x[i]$ | or | $-$ | or | $x[i]$ |
| $-$ | | $y[j]$ | | $y[j]$ |

# Edit distance

- By the above key observation, then

$$e(i,j) = \min\{\underbrace{1+e(i-1,j)}_{\text{case 1}},\ \underbrace{1+e(i,j-1)}_{\text{case 2}},\ \underbrace{\text{diff}(i,j)+e(i-1,j-1)}_{\text{case 3}}\}$$

where

$$\text{diff}(i,j) = \begin{cases} 0 & \text{if } x[i] = y[j] \\ 1 & \text{if } x[i] \neq y[j] \end{cases}$$

- Question: how to find the corresponding optimal alignment?

# Edit distance

- The answers to all the subproblems $e(i, j)$ form a two-dimensional table, and the final answer (our objective) is at $e(m, n)$.

# Edit distance

- The answers to all the subproblems $e(i, j)$ form a two-dimensional table, and the final answer (our objective) is at $e(m, n)$.
- Initialization:

$$\begin{aligned} e(0,0) &= 0; \\ e(i,0) &= i \text{ for } i = 1, \ldots, m \\ e(0,j) &= j \text{ for } j = 1, \ldots, n \end{aligned}$$

# Edit distance

- The answers to all the subproblems $e(i, j)$ form a two-dimensional table, and the final answer (our objective) is at $e(m, n)$.
- Initialization:

$$e(0,0) = 0;$$
$$e(i,0) = i \text{ for } i = 1, \dots, m$$
$$e(0,j) = j \text{ for } j = 1, \dots, n$$

- Pseudocode

# Edit distance

- The answers to all the subproblems $e(i, j)$ form a two-dimensional table, and the final answer (our objective) is at $e(m, n)$.
- Initialization:

$$\begin{aligned} e(0,0) &= 0; \\ e(i,0) &= i \text{ for } i = 1, \dots, m \\ e(0,j) &= j \text{ for } j = 1, \dots, n \end{aligned}$$

- Pseudocode
- Example 1. $x =$ 'snowy', $y =$ 'sunny'

# Edit distance

- The answers to all the subproblems $e(i, j)$ form a two-dimensional table, and the final answer (our objective) is at $e(m, n)$.
- Initialization:

$$e(0,0) = 0;$$
$$e(i,0) = i \text{ for } i = 1, \dots, m$$
$$e(0,j) = j \text{ for } j = 1, \dots, n$$

- Pseudocode
- Example 1. $x$ = 'snowy', $y$ = 'sunny'

| | | s | u | n | n | y |
|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 |
| s | 1 | 0 | 1 | 2 | 3 | 4 |
| n | 2 | 1 | 1 | 1 | 2 | 3 |
| o | 3 | 2 | 2 | 2 | 2 | 3 |
| w | 4 | 3 | 3 | 3 | 3 | 3 |
| y | 5 | 4 | 4 | 4 | 4 | 3 |

# Edit distance

- The answers to all the subproblems $e(i, j)$ form a two-dimensional table, and the final answer (our objective) is at $e(m, n)$.
- Initialization:

$$e(0,0) = 0;$$
$$e(i,0) = i \text{ for } i = 1, \ldots, m$$
$$e(0,j) = j \text{ for } j = 1, \ldots, n$$

- Pseudocode
- Example 1. $x$ = 'snowy', $y$ = 'sunny'

| | | s | u | n | n | y |
|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 |
| s | 1 | 0 | 1 | 2 | 3 | 4 |
| n | 2 | 1 | 1 | 1 | 2 | 3 |
| o | 3 | 2 | 2 | 2 | 2 | 3 |
| w | 4 | 3 | 3 | 3 | 3 | 3 |
| y | 5 | 4 | 4 | 4 | 4 | 3 |

Therefore, the edit distance between $x$ and $y$ = $e(5, 5) = 3$.

# Edit distance

Example 2. $x =$ 'heroically', $y =$ 'scholarly'

# Edit distance

Example 2. $x =$ 'heroically', $y =$ 'scholarly'

| | | s | c | h | o | l | a | r | l | y |
|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| h | 1 | 1 | 2 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| e | 2 | 2 | 2 | 3 | 3 | 4 | 5 | 6 | 7 | 8 |
| r | 3 | 3 | 3 | 3 | 4 | 4 | 5 | 5 | 6 | 7 |
| o | 4 | 4 | 4 | 4 | 3 | 4 | 5 | 6 | 6 | 7 |
| i | 5 | 5 | 5 | 5 | 4 | 4 | 5 | 6 | 7 | 7 |
| c | 6 | 6 | 5 | 6 | 5 | 5 | 5 | 6 | 7 | 8 |
| a | 7 | 7 | 6 | 6 | 6 | 6 | 5 | 6 | 7 | 8 |
| l | 8 | 8 | 7 | 7 | 7 | 6 | 6 | 6 | 6 | 7 |
| l | 9 | 9 | 8 | 8 | 8 | 7 | 7 | 7 | 6 | 7 |
| y | 10 | 10 | 9 | 9 | 9 | 8 | 8 | 8 | 7 | 6 |

# Edit distance

Example 2. $x =$ 'heroically', $y =$ 'scholarly'

| | | s | c | h | o | l | a | r | l | y |
|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| h | 1 | 1 | 2 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| e | 2 | 2 | 2 | 3 | 3 | 4 | 5 | 6 | 7 | 8 |
| r | 3 | 3 | 3 | 3 | 4 | 4 | 5 | 5 | 6 | 7 |
| o | 4 | 4 | 4 | 4 | 3 | 4 | 5 | 6 | 6 | 7 |
| i | 5 | 5 | 5 | 5 | 4 | 4 | 5 | 6 | 7 | 7 |
| c | 6 | 6 | 5 | 6 | 5 | 5 | 5 | 6 | 7 | 8 |
| a | 7 | 7 | 6 | 6 | 6 | 6 | 5 | 6 | 7 | 8 |
| l | 8 | 8 | 7 | 7 | 7 | 6 | 6 | 6 | 6 | 7 |
| l | 9 | 9 | 8 | 8 | 8 | 7 | 7 | 7 | 6 | 7 |
| y | 10 | 10 | 9 | 9 | 9 | 8 | 8 | 8 | 7 | 6 |

Therefore, the edit distance between $x$ and $y = e(10, 9) = 6$

# Edit distance

Example 2. $x =$ 'heroically', $y =$ 'scholarly'

| | | s | c | h | o | l | a | r | l | y |
|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| h | 1 | 1 | 2 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| e | 2 | 2 | 2 | 3 | 3 | 4 | 5 | 6 | 7 | 8 |
| r | 3 | 3 | 3 | 3 | 4 | 4 | 5 | 5 | 6 | 7 |
| o | 4 | 4 | 4 | 4 | 3 | 4 | 5 | 6 | 6 | 7 |
| i | 5 | 5 | 5 | 5 | 4 | 4 | 5 | 6 | 7 | 7 |
| c | 6 | 6 | 5 | 6 | 5 | 5 | 5 | 6 | 7 | 8 |
| a | 7 | 7 | 6 | 6 | 6 | 6 | 5 | 6 | 7 | 8 |
| l | 8 | 8 | 7 | 7 | 7 | 6 | 6 | 6 | 6 | 7 |
| l | 9 | 9 | 8 | 8 | 8 | 7 | 7 | 7 | 6 | 7 |
| y | 10 | 10 | 9 | 9 | 9 | 8 | 8 | 8 | 7 | 6 |

Therefore, the edit distance between $x$ and $y = e(10, 9) = 6$

*Note: LCS*$(x, y) = 5$