Algorithmique

TP 1 : Programmation dynamique

Calcul de la distance d'édition

Renaud Marlet
Laboratoire LIGM-IMAGINE

http://imagine.enpc.fr/~marletr

Distance de Levenshtein (1965) = distance d'édition

- Nombre minimum de modifications pour passer d'une chaîne à une autre :
  - suppression d'un caractère
  - insertion d'un caractère
  - remplacement d'un caractère par un autre
    - ex. d_L(ponts,hotes) = 3
- Mesure de la similarité de deux chaînes de caractères
  - applications : vérificateur orthographique, OCR...
  - généralisation du pb de plus longue séquence commune
- Complexité : $O(mn)$ pour deux mots de taille m et n

Distance de Damerau-Levenshtein

- Nombre minimum de modifications pour passer d'une chaîne à une autre :
  - suppression d'un caractère
  - insertion d'un caractère
  - remplacement d'un caractère par un autre
  - transposition de deux caractères successifs
    - ex. d_{DL}(écoles,éclose) = 2
      d_{L}(écoles,éclose) = 3

écoles, o ↔ l =
écloes, s ↔ e =
éclose

écoles, +l =
écloles, l → s =
écloses, -s =
éclose

- Couvrirait ≈80% des fautes d'orthographes (anglais)
- Complexité : $O(mn)$ pour deux mots de taille m et n

TP 1 : distance d'édition

1. **Fermez votre ordinateur !**
2. Étudiez le calcul par programmation dynamique de la distance de Levenshtein $d_L(s,s')$ entre deux chaînes s et s'
   - **suivez la méthodologie du cours !**
   - **indice :** étudiez la distance des préfixes
   - **rédigez** la preuve de la sous-structure optimale (qq lignes)

   chaîne $s = c_1 \ldots c_n$
   préfixe $s_i = c_1 \ldots c_i \; 1 \leq i \leq n$
3. Proposez un algorithme itératif de calcul de la distance
4. Indiquez sa complexité (polynomiale) en espace et en temps
5. Implémentez-le (≈ **une dizaine de lignes de code !**)
6. Affichez la suite de modifs élémentaires pour aller de s à s'
7. Étendez ce travail à la distance de Damerau-Levenshtein
8. Proposez et discutez une version linéaire en espace