

## Détermination d'un plus long sous mot commun à deux mots sur un réseau linéaire

Thierry Lecroq et Jean-Frédéric Myoupo <sup>a</sup>

<sup>a</sup>LIR, Université de Rouen, Place Emile Blondel, 76821 Mt-St-Aignan Cedex, France.

### Résumé

Le problème du plus long sous mot commun consiste à trouver un sous mot de longueur maximale appartenant à un mot  $x$  et à un mot  $y$ . En séquentiel la méthode classique utilise la programmation dynamique, le problème peut alors être résolu en temps et en espace  $O(|x|.|y|)$ . En ajoutant une technique “diviser pour régner” il est possible d'obtenir une complexité spatiale linéaire [1]. En parallèle Robert et Tchunte [2] ont donné un algorithme systolique modulaire linéaire pour trouver la longueur d'un plus long sous mot commun mais pour exhiber un plus long sous mot commun ils ont recours à un réseau orthogonal.

Nous présentons un réseau linéaire pouvant exhiber un plus long sous mot commun. La technique employée donne un nouvel algorithme séquentiel simple fonctionnant en temps  $O(|x|.|y|)$  et en espace  $O(\min(|x|, |y|).l)$  avec  $l$  la longueur du plus long sous mot commun.

On note  $LLCS[i, j]$  la longueur d'un plus long sous mot commun entre  $x[1, i]$  et  $y[1, j]$  et  $LCS[i, j]$  un plus long sous mot commun entre  $x[1, i]$  et  $y[1, j]$ .

Pour  $1 \leq i \leq |x|$  et  $1 \leq j \leq |y|$ :  $LLCS(i, 0) = LLCS(0, j) = 0$  et  $LCS(i, 0) = LCS(0, j) = \varepsilon$ .

Pour  $1 \leq i \leq |x|$  et  $1 \leq j \leq |y|$ :

$$LLCS(i, j) = \begin{cases} \text{si } x[i] = y[j] & \text{alors } 1 + LLCS(i-1, j-1) \\ \text{sinon} & \max(LLCS(i-1, j), LLCS(i, j-1)) \end{cases}$$

Pour  $1 \leq i \leq |x|$  et  $1 \leq j \leq |y|$ :

$$LCS(i, j) = \begin{cases} \text{si } x[i] = y[j] & \text{alors } LCS(i-1, j-1)x[i] \\ \text{sinon si } LLCS(i-1, j) \geq LLCS(i, j-1) & \text{alors } LCS(i-1, j) \\ \text{sinon} & LCS(i, j-1) \end{cases}$$

Notre algorithme systolique linéaire utilise donc  $|x|$  processeurs. Chaque processeur  $j$  reçoit  $x[i]$ ,  $LLCS(i-1, j-1)$ ,  $LLCS(i, j-1)$  et  $LCS(i-1, j-1)$  et stocke  $y[j]$ ,  $LLCS[i, j]$  et  $LCS[i, j]$  après le traitement de  $x[i]$ .

Notre algorithme fonctionne en temps  $O(|x| + |y|.l)$ .

### REFERENCES

- [1] D.S. HIRSCHBERG, A linear space algorithm for computing maximal common subsequences, *Comm. ACM* **18**(6) (1975) 664-675.
- [2] Y. ROBERT, M. TCHUNTE, A systolic array for the longest common subsequence problem, *Inform. Proc. Lett.* **21** (1985) 191-198.