Relazione esercizio 2

Daniele Liberatore, Sandro Massa, Matteo Palazzo

11 giugno 2019

Funzionamento edit-distance

Per l'implementazione in programmazione dinamica, teniamo nota delle operazioni già svolte onde evitare ridondanze. Tale tecnica, nota come *memoizzazione*, è efficacie in quanto i sotto-problemi del problema originario non sono indipendenti, richiedendo operazioni ridondanti, eliminate seguento questo metodo.
La matrice c che contiene tali risultati ha per dimensioni la lunghezza delle stringhe X e Y. La cella c_{ij} conterrà l'edit-distance tra i prefissi X_i e Y_j. Inizialmente vuote, le celle vengono valorizzate alla prima operazione corrispondente. A meno della *memoizzazione* l'algoritmo equivale alla versione ricorsiva (vedasi slide per la definizione).
Vengono riportate alcune comparazioni tra le versioni dell'edit-distance:

X	*Y*	*Edit-distance*	*Ricorsivo*	*Dinamico*
Calla	Palla	1	0.100s	0.100s
Bcamap	Palla	5	0.100s	0.100s
Lonpirante	Lungimirante	4	0.800s	0.100s
Antocartistuzinale	Anticostituzionale	5	$>$ 30m	0.100s

Funzionamento checker

Il *Checker* confronta ogni parola presente in *target* (correctme.txt) con le stringhe di *dict* (dictionary.txt), associando ad ogni parola della prima una lista di parole dalla seconda con edit-distance minima. Se l'edit-distance di una parola di *dict* è maggiore di quella già calcolata, questa viene ignorata. Mediamente, *Checker_Test* impiega 5 secondi per terminare, migliorando qualora si ordini in modo crescente *dict* per lunghezza delle stringhe. La complessita diminuisce comunque di solo un fattore costante.