Laboratori 4 CDI-FIB

Curs 2022/2023

29 de març de 2023

Mètodes de diccionari. Definiu funcions per codificar/descodificar amb mètodes de diccionari: es codifica un text `txt` que es converteix en una llista de "tokens" `tok` amb format que depèn del mètode. Per a la descodificació en la majoria de casos n'hi ha prou amb la llista de tokens però en alguns mètodes s'ha de saber també l'alfabet `alf` en què està escrit el text.

LZ77: diccionaris de finestra lliscant.

- `encode_LZ77(txt,s,t)` dona com a resultat una llista de tokens que són tripletes $\texttt{tok} = [(\theta_\texttt{i}, \lambda_\texttt{i}, \texttt{a}_\texttt{i})]_{1 \leqslant \texttt{i} \leqslant \texttt{n}}$. Cada tripleta conté dos enters positius θ_i i λ_i i una lletra $\texttt{a}_i$ de l'alfabet `alf`.

  Els paràmetres `t` i `s` indiquen la mida màxima de la paraula del diccionari i la mida de banda esquerra de la finestra on es busca aquesta paraula, respectivament, de manera que els enters θ_i i λ_i dels tokens pertanyen als intervals $1 \leqslant \theta_i \leqslant \texttt{s}$ i $0 \leqslant \lambda_i \leqslant \texttt{t}$.

  Quan la primera lletra a codificar no es troba a la finestra esquerra es codifica amb el token $(1, 0, \texttt{a})$, on `a` és la lletra; aquí el paràmetre $\lambda = 0$ indica que no hi ha hagut match i el paràmetre $\theta = 1$ no serveix per a res i s'hi podria posar qualsevol nombre; nosaltres convindrem de posar-hi un 1.
- `decode_LZ77(tok)` recupera el text `txt` a partir de la llista de tokens `tok`.
- `encode_LZSS(txt,mm,s,t)` dona com a resultat una llista de tokens que poden ser de dos tipus:
  - o bé un parell de nombres enters (θ_i, λ_i) amb $\texttt{mm} \leqslant \lambda_i \leqslant \texttt{t}$ i $1 \leqslant \theta_i \leqslant \texttt{s}$;
  - o bé una lletra $\texttt{a}_i$ de l'alfabet `alf`.

  En aquest cas `mm` és la mida mínima de les parules del diccionari (el match mínim acceptable per usar un token l·llarg"), de manera que $\texttt{mm} \leqslant \lambda_i \leqslant \texttt{t}$.
- `decode_LZSS(tok)` recupera el text `txt` a partir de la llista de tokens `tok`.

LZ78: diccionaris creats a partir de paraules del text.

- `encode_LZ78(txt)` dona com a resultat una llista de tokens que són parells $\texttt{tok} = [(\imath_{\texttt{i}}, \texttt{a}_{\texttt{i}})]_{1 \leqslant \texttt{i} \leqslant \texttt{n}}$. Cada parell conté un enter $\imath_i$ i una lletra $\texttt{a}_i$ de l'alfabet `alf`.

  L'enter $\imath_i$ és un punter cap a una paraula del diccionari.

- `decode_LZ78(tok)` recupera el text `txt` a partir de la llista de tokens `tok`.

- `encode_LZW(txt)` dona com a resultat una llista nombres enters $[\imath_{\texttt{i}}]_{1 \leqslant \texttt{i} \leqslant \texttt{n}}$ que representen punters cap a paraules del diccionari.

- `decode_LZW(tok,alf)` recupera el text `txt` a partir de la llista de tokens `tok`. En aquest cas s'ha de passar l'alfabet de les lletres del text (ordenades alfabèticament) com a paràmetre al descodificador per tal que pugui inicialitzar el diccionari.

Altres: reordenació de les lletres. Escriviu funcions que codifiquin/descodifiquin un text segons els mètodes de Burrows-Wheeler i move-to-front.

- `encode_burrows_wheeler(txt)` retorna una seqüència $\texttt{cod} \in \mathbb{A}^*$ de lletres de l'alfabet de la mateixa longitud n que la seqüència `txt` i un enter $\texttt{i} \in [0, n-1]$ que indica la posició de `txt` dins de la llista de les seves permutacions cícliques ordenades alfabèticament.

- `decode_burrows_wheeler(cod,i)` recupera el text `txt`.

- `encode_move_to_front(txt)` retorna una seqüència $\texttt{cod} \in \mathbb{N}^*$ de nombres enters entre 0 i $q-1$, on q és el nombre de lletres de l'alfabet amb què s'ha escrit el text.

- `decode_move_to_front(cod,alf)` recupera el text `txt`. Se li ha de passar l'alfabet (ordenat).