Université Laval
Faculté des sciences et de génie
Département d'informatique et de génie logiciel
IFT-4003/7023

Danny Dubé
Hiver 2016
Version: 5 avril

Travail pratique #2
Compression par dictionnaire – Énumération de sous-chaînes

Questions

Donnez le développement pour vos réponses, en particulier quand le calcul est long.

1. (*40 points.*) **Compression par dictionnaire.** Dans les sous-questions qui suivent, on considère que LZ77 (et sa variante LZSS) respecte les conventions suivantes.

   Convention I: LZ77 recherche toujours une copie ayant la longueur l la plus grande possible.

   Convention II: lorsqu'il existe plusieurs copies de longueur maximale, considérez que LZ77 favorise celle qui a le décalage o le plus petit possible.

   Convention III: le dernier triplet de LZ77 pourrait ne pas respecter les conventions I et II, ceci parce que le texte d'entrée se termine. La priorité est de produire une suite de triplets qui décrivent exactement le texte d'entrée.

   (a) Donnez la suite de triplets $\langle o, l, \texttt{c}\rangle$ qui sont produits en traitant l'entrée qui suit à l'aide de la technique LZ77 de base (présentée à partir de la page 10 des acétates). Considérez que la fenêtre de recherche n'est pas limitée en taille. Lorsqu'il n'y a pas de copie trouvée, c'est-à-dire lorsque $l = 0$, fixez arbitrairement le décalage à zéro, c'est-à-dire $o = 0$.

   ```
   caabbcaabcba
   ```

   (b) Donnez la suite des messages produits en traitant l'entrée donnée en (a) à l'aide de LZSS (présentée à la page 17 des acétates). Un message est soit une paire $\langle o, l\rangle$, lorsqu'une copie est trouvée, soit un singleton $\langle c\rangle$. Nous convenons que LZSS ne produit une paire que pour une copie non-triviale, i.e. telle que $l \geq 2$. Le reste du comportement de LZSS est similaire à celui de LZ77 en (a).

   (c) Décrivez le traitement de l'entrée donnée en (a) par LZ78 (présentée à partir de la page 18 des acétates), en montrant l'état du dictionnaire à chaque étape. Considérez que le dictionnaire contient initialement une seule entrée: l'entrée 0 associée à la chaîne ϵ.

   (d) Décrivez le traitement de l'entrée donnée en (a) par LZW (présentée à partir de la page 32 des acétates), en montrant l'état du dictionnaire à chaque étape. Considérez que l'alphabet est {`a`, `b`, `c`}. Initialisez le dictionnaire en associant les entrées 0 à 2 aux chaînes `a`, `b` et `c`, respectivement.

(e) Décodez la séquence de triplets suivante à la façon de LZ77.

$\langle 0, 0, \texttt{c}\rangle,\ \langle 0, 0, \texttt{r}\rangle,\ \langle 0, 0, \texttt{a}\rangle,\ \langle 0, 0, \texttt{n}\rangle,\ \langle 2, 1, \texttt{c}\rangle,\ \langle 5, 3, \texttt{e}\rangle,\ \langle 5, 1, \texttt{a}\rangle,\ \langle 6, 1, \texttt{n}\rangle,\ \langle 0, 0, \texttt{e}\rangle$

(f) Décodez la séquence de messages suivante à la façon de LZSS.

$$\langle \texttt{r}\rangle,\ \langle \texttt{e}\rangle,\ \langle \texttt{p}\rangle,\ \langle \texttt{e}\rangle,\ \langle \texttt{t}\rangle,\ \langle \texttt{i}\rangle,\ \langle 2, 2\rangle,\ \langle \texttt{v}\rangle,\ \langle 4, 2\rangle,\ \langle \texttt{e}\rangle$$

(g) Décodez la séquence de couples suivante à la façon de LZ78.

$$\langle 0, \texttt{c}\rangle,\ \langle 0, \texttt{a}\rangle,\ \langle 0, \texttt{t}\rangle,\ \langle 2, \texttt{l}\rangle,\ \langle 2, \texttt{n}\rangle,\ \langle 0, \texttt{i}\rangle,\ \langle 2, \texttt{n}\rangle$$

(h) Décodez la séquence d'index suivante à la façon de LZW. Considérez que les entrées 0 à 5 du dictionnaire sont initialement associées aux chaînes `a`, `e`, `l`, `m`, `n` et `t`, respectivement.

3, 1, 4, 5, 0, 2, 1, 6, 8

(i) Si on refaisait un exercice comme la question 1(b) mais en cessant de respecter la Convention II, on tomberait sur des cas où plusieurs plus longues copies sont disponibles. Énumérez toutes les façons dont LZSS pourrait traiter l'entrée suivante en respectant la Convention I mais pas la Convention II. Notez qu'on continue à restreindre les copies à être d'une longueur d'au moins deux ($l \geq 2$). (*Indice:* vous devriez trouver 4 façons distinctes.)

`cabaabcaabbcbbaa`

(j) Maintenant, si on refaisait la question 1(i) mais sans respecter la Convention I non plus, on observerait une multiplication des façons différentes de traiter l'entrée. Énumérez toutes les façons dont LZSS pourrait traiter l'entrée suivante. Notez qu'il n'est même pas nécessaire de préférer une copie à un singleton. Notez qu'on continue à restreindre les copies à être d'une longueur d'au moins deux ($l \geq 2$). (*Indice:* vous devriez trouver 8 façons distinctes.)

`chouchou`

2. (*20 points.*) **Prédiction basée sur les contextes.** Ici, nous considérons la variante PPM A. Supposons que le début de texte `baccabac` a déjà été prédit et encodé. Les tables suivantes présentent les modèles de Markov d'ordres -1, 0, 1, 2 et 3 cumulés à ce point. Supposez que l'alphabet complet est $\{\texttt{a}, \texttt{b}, \texttt{c}, \texttt{d}\}$.

   Vous devez indiquer comment procéderaient les traitements indiqués dans les sous-questions. À chaque sous-question, il faut partir de l'état décrit dans les tables; i.e. c'est comme si, à chaque sous-question, on envisageait un scénario différent à la suite de `baccabac`. Il ne faut pas oublier de mettre à jour les tables après l'encodage de chaque symbole.

   Décrivez la(les) opération(s) successive(s) effectuée(s) lors du traitement, comme: "échappement vers le modèle d'ordre k", "encodage du symbole c dans le modèle d'ordre k avec probabilité p", "incrémentation du poids du symbole c dans le contexte w", etc.

Ordre -1

C	S	N
—	a	1
	b	1
	c	1
	d	1

Ordre 0

C	S	N
—	a	3
	b	2
	c	3
	⟨Esc⟩	1

Ordre 1

C	S	N
a	b	1
	c	2
	⟨Esc⟩	1
b	a	2
	⟨Esc⟩	1
c	a	1
	c	1
	⟨Esc⟩	1

Ordre 2

C	S	N
ab	a	1
	⟨Esc⟩	1
ac	c	1
	⟨Esc⟩	1
ba	c	2
	⟨Esc⟩	1
ca	b	1
	⟨Esc⟩	1
cc	a	1
	⟨Esc⟩	1

Ordre 3

C	S	N
aba	c	1
	⟨Esc⟩	1
acc	a	1
	⟨Esc⟩	1
bac	c	1
	⟨Esc⟩	1
cab	a	1
	⟨Esc⟩	1
cca	b	1
	⟨Esc⟩	1

(a) On encode le symbole `b`.

(b) On encode successivement les symboles `cd`.

(c) On encode successivement les symboles `ab`.

(d) Refaire la sous-question (a) mais en utilisant le principe d'exclusion.

3. (*20 points.*) **Transformées de Burrows-Wheeler et "Move-to-front".**

   (a) Effectuez la BWT sur la chaîne `contamination`. N'oubliez pas de calculer aussi le rang de la chaîne (en comptant à partir de zéro).

   (b) Défaites la BWT sur $\langle$`cceatuaeh`, 2$\rangle$.

   (c) Effectuez la transformée "Move-to-front" sur `acidifiee` en prenant comme liste initiale les 10 premières lettres de l'alphabet, dans l'ordre. Numérotez les lettres de 0 à 9.

   (d) Effectuez l'inverse de la transformée "Move-to-front" sur la suite de rangs 3, 4, 4, 8, 3, 3, 0. Utilisez la même liste initiale qu'en (c).

4. (*20 points.*) **Compression par énumération de sous-chaînes.** Soit $\mathbf{D} =$ `011101` la chaîne de données à compresser par énumération de sous-chaînes.

   (a) Établissez la table d'énumération pour **D**, à la façon de la page 5 des acétates sur la compression par énumération de sous-chaînes (CSE).

   (b) Calculez les bornes entourant la valeur de $C_{\texttt{010}}$ lorsque vient le temps de prédire et d'encoder ce compteur, à la façon de la page 16.

   (c) Dessinez l'arbre de sous-chaînes infini (IST) pour **D**, à la façon de la page 18. Vous pouvez représenter les branches symboliquement à l'aide de '...' à partir des noeuds de profondeur 6 dans l'IST.

   (d) Dessinez l'arbre de sous-chaînes compact (CST). Identifiez les sous-arbres isomorphes et remplacez les arcs vers des sous-arbres qui se répètent par des arcs (en pointillés) vers des noeuds situés à la même profondeur ou plus haut, à la façon de la page 19. Le véritable CST comporte 11 noeuds. Ce n'est pas nécessaire de produire le véritable CST. C'est acceptable de produire un "CST" comportant plus de noeuds, en autant qu'il soit de taille finie et isomorphe à l'IST.

Remise des travaux

Vous devez remettre le travail via **Pixel**. Les autres modalités de remise sont inscrites dans le plan de cours.