IFT 3290 H2003 **EXAMEN INTRA** **Miklós Csűrös**

Bioinformatique

Mercredi 19 février, 11 :30–13 :30, salle G-415

Directives :

– Toute documentation statique[1] est permise.
– L'examen intra vaut 100 points : il suffit de finir 4 des 5 «grands» problèmes de l'examen. Si vous travaillez sur tous les cinq problèmes, vous pouvez avoir jusqu'à 25 points de boni.

[1] écrite ou imprimée, pas d'ordinateur, pas de calculatrice, ...

0 Votre nom (1 point)

Écrivez votre nom et code permanent sur tous les cahiers remis.

1 «Ditsance» d'édition (20 points)

Dans le modèle de base de la distance d'édition, on a considéré trois sources d'erreurs : substitutions, suppressions et insertions. Cet exercice vous demande d'étendre le modèle en incluant l'opération d'inversion qui renverse l'ordre de deux caractères consécutifs. Donnez un algorithme qui calcule la distance d'édition étendue entre deux mots S et T définie comme le nombre minimal d'opérations d'éditions (substitutions, suppressions, insertions, et inversions) qui transforme S en T. Exemple : $\mathsf{distance}(\texttt{machine}, \texttt{mxcihn}) = 3$ (substitution `a` $\leftrightarrow$ `x`, inversion `hi` $\leftrightarrow$ `ih`, suppression de `e`). Analysez le temps du calcul de l'algorithme.

2 Alignement gauchiste (25 points)

Cet exercice vous demande de concevoir un algorithme pour l'alignement global de deux séquences S et T. La solution par programmation dynamique qu'on a vue au cours calcule les alignements optimaux des préfixes. Donnez une solution qui calcule les alignements optimaux des suffixes de la façon suivante. Soit $F(i, j)$ la valeur de l'alignement optimal des suffixes $S\Big[i+1..|S|\Big]$ et $T\Big[j+1..|T|\Big]$. (Dans le cas extrême quand $i = |S|$ ou $j = |T|$, le suffixe est défini comme le mot vide ε). La pondération de l'alignement est donnée par un tableau $P\colon \Sigma \cup \{\texttt{-}\} \times \Sigma \cup \{\texttt{-}\}$: $P(a, a)$ donne la valeur d'une identité, $P(a, b)$ donne la valeur de substituer b (dans T) pour a (dans S), $P(\texttt{-}, a)$ donne la valeur de supprimer le caractère a de T, et $P(a, \texttt{-})$ donne la valeur de supprimer le caractère a de S. Dérivez les récurrences pour calculer $F(i, j)$ où $i = 0, \ldots, |S|$, $j = 0, \ldots, |T|$. Donnez l'algorithme qui calcule un alignement optimal avec sa valeur, par programmation dynamique à l'aide des $F(\cdot)$. Analysez le temps du calcul de votre algorithme. Expliquez pourquoi le calcul des $F(\cdot)$ mène à trouver la solution optimale.

3 Traduction (19 points)

Traduisez la séquence d'ARN messager AGUGCAAUGUCCGGUUUUCAA à une séquence protéique qui y correspond. Combien de séquences d'ARN encodent la même séquence protéique ? Si le début de la traduction n'est pas spécifiée, quelles séquences protéiques peuvent être produites à partir de la même séquence d'ARN ? Utilisez le tableau du code génétique montré ici.

UUU Phe	UCU Ser	UAU Tyr	UGU Cys
UUC Phe	UCC Ser	UAC Tyr	UGC Cys
UUA Leu	UCA Ser	UAA TERM	UGA TERM
UUG Leu	UCG Ser	UAG TERM	UGG Trp
CUU Leu	CCU Pro	CAU His	CGU Arg
CUC Leu	CCC Pro	CAC His	CGC Arg
CUA Leu	CCA Pro	CAA Gln	CGA Arg
CUG Leu	CCG Pro	CAG Gln	CGG Arg
AUU Ile	ACU Thr	AAU Asn	AGU Ser
AUC Ile	ACC Thr	AAC Asn	AGC Ser
AUA Met	ACA Thr	AAA Lys	AGA Arg
AUG Met	ACG Thr	AAG Lys	AGG Arg
GUU Val	GCU Ala	GAU Asp	GGU Gly
GUC Val	GCC Ala	GAC Asp	GGC Gly
GUA Val	GCA Ala	GAA Glu	GGA Gly
GUG Val	GCG Ala	GAG Glu	GGG Gly

4 Circulation (25 points)

L'apropos de ce problème est le fait qu'il y a un grand nombre de génomes bactériens qui sont circulaires : l'ADN dans ces organismes forme un cercle. Même si le génome est circulaire, sa séquence est stockée comme une séquence linéaire, après qu'on l'a coupée en une position quelconque. On voudrait trouver l'occurrence d'un mot P de longueur n dans une séquence circulaire T de longueur m. Le mot P apparaît dans T en position $k \leq m - n + 1$ si $P[1] = T[k], P[2] = T[k + 1], \ldots, P[n] = T[k - 1 + n]$. Le mot P apparaît dans T en position $k > m-n+1$ si $P[1] = T[k], P[2] = T[k+1], \ldots, P[m-k+1] = T[m], P[m-k+2] = T[1], P[m-k+3] = T[2], \ldots, P[n] = T[n+k-m-1]$. Donnez un algorithme efficace pour ce problème et analysez son temps de calcul en fonction de n et m.

5 Le meilleur chemin (35 points)

Soit $\mathcal{A}$ un automate probabiliste non-déterministe avec un ensemble d'états Q, état initial $q_{\text{init}} \in Q$, probabilités de transition $\tau\colon Q \times Q \mapsto [0,1]$, et une fonction d'impression $f\colon Q \times Q \mapsto \Sigma \cup \{\varepsilon\}$ où ε est le mot vide. L'automate $\mathcal{A}$ génère une séquence S sur l'alphabet Σ en chaque série de transitions (c-à-d, chemin) $q_0 = q_{\text{init}}, q_1, q_2, \ldots, q_k$. Lors de chaque transition $q_{i-1} \to q_i$, le caractère $f(q_{i-1}, q_i)$ est imprimé. Bien sûr, si $f(q_{i-1}, q_i) = \varepsilon$, rien n'est imprimé lors de la transition. La séquence S est obtenue par la concaténation des caractères imprimés. La probabilité du chemin est calculée par $\prod_{i=1}^{k} t(p_{i-1}, p_i)$. On sait que les transitions sur le mot vide sont limitées : les états sont groupés dans deux ensembles, Q_v et Q_n. Si $q \in Q_v$ et $q' \in Q_n$, alors il est *possible* que $f(q, q') = \varepsilon$ et $\tau(q, q') > 0$. Si $q \neq Q_v$ ou $q' \neq Q_n$, alors on a $\tau(q, q') = 0$ ou $f(q, q') \neq \varepsilon$. Le problème de cet exercice est de trouver le chemin le plus probable qui génère une séquence S donnée à l'entrée. Donnez un algorithme efficace pour résoudre le problème. (Essayez de suivre la conception de l'algorithme Viterbi pour les modèles Markov cachés.) Analysez le temps du calcul de votre algorithme.

BONNE CHANCE !