Méthodes Empiriques et Langages de Script
TP 4

G. A. Musillo
musillo4@etu.unige.ch

November 30, 2005

1 Exercice 1: un index

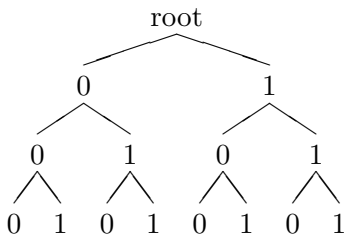
On vous demande de réaliser un index qui prend en entrée un fichier de données textuelles et qui écrit sur la sortie standard la liste des mots de ce fichier, triée alphabétiquement, ainsi que les lignes auxquelles ils apparaissent.

2 Exercice 2: le jeu de Shannon

Voici une façon simpliste de générer du texte automatiquement: associer chaque caractère de l'alphabet à un nombre entier distinct compris entre 1 et 27 (un tel alphabet comprend les 26 lettres et l'espace), lancer un dé à 27 faces et écrire le caractère correspondant à la face tirée. Selon quelques statisticiens, un tel modèle de génération pourrait très bien être réalisé par un chimpanzé tapant au hasard sur les touches d'un clavier alphabétique !
On vous demande de réaliser un modèle de génération plus subtil. Ce modèle, qu'on dit *uni-gramme*, génère des caractères proportionellement à leur fréquence relative d'occurence.
Pour réaliser ce programme, donnez-vous un fichier de textes français et calculez la fréquence relative des caractères minuscules de l'alphabet latin et de l'espace qui y apparaissent.
Au moyen de la fonction prédéfinie `rand()` qui retourne un réel compris entre `0` et `1`, générez une séquence de nombres aléatoires et écrivez sur la sortie standard la séquence aléatoire de caractères qui lui correspond.
On vous demandera prochainement de réaliser un modéle de génération plus complexe qui exploite les probabilités conditionnelles des caractères, plutôt que les probabilitées simples.

3 Exercice 3: un générateur de mots

Etant donné un alphabet représenté par un tableau et un entier n (> 1), codez un programme qui génére toutes les séquences et rien que les séquences de longueur n de caractères appartenant à l'alphabet. Autrement dit, votre programme doit pouvoir générer tous les mots qui matchent l'expression rationnelle $\Sigma\{n\}$ (où Σ est l'alphabet donnée en entrée).
Par exemple, si on lui donnait en entrée l'alphabet `01` et le nombre `3`, alors le programme retournerait en sortie les mots `000`, `001`, `010`, `011`, `100`, `101`, `110` et `111`. Ceci revient à parcourir l'arbre à lettres (ou arbre lexicographique, ou *trie* en anglais) suivant:



Les travaux pratiques suivantes montreront l'intérêt de tels arbres pour la compression de données et l'implémentation de dictionnaires.