MATH0500 - Introduction à l'algorithmique numérique

Projet de programmation

Dans ce projet, nous allons mettre en pratique les méthodes vues au cours pour traiter des matrices creuses.

En 2019, la fédération francophone de tennis (AFT) a mis au point un nouveau système de classement pour les joueurs amateurs. Ce nouveau système a dû être révisé plusieurs fois pour corriger des effets que la fédération n'avait pas prévus. Dans ce projet, nous allons mettre en place une modélisation un peu inhabituelle des classements en utilisant un concept similaire au page rank de google et le comparer avec la version obtenue par la fédération.

Nous allons travailler avec un fichier extrait du site de l'AFT correspondant à tous les matches amateurs joués en une saison. Ce fichier est un fichier texte dont chaque ligne est un match amateur joué en Belgique francophone en 2020. Sur chaque ligne, il y a trois valeurs. Les deux premières valeurs correspondent aux deux numéros d'affiliation des deux joueurs s'étant affrontés. Le troisième nombre indique si c'est le premier ou le deuxième joueur qui a remporté la partie. Notre première tâche sera de transformer le fichier en une matrice dont les lignes et les colonnes sont les joueurs affiliés à la fédération. Ensuite, afin que la suite du calcul puisse avoir du sens, nous devons nous assurer que notre matrice est une matrice stochastique, c'est-à-dire que la somme de chacune des lignes soit égale à 1. Pour ce faire, nous allons remplir dans un premier temps la matrice A de telle façon que $A(i, j)$ est égale à 1 si le joueur i a battu au moins une fois le joueur j, divisé par le nombre de matches gagnés par le joueur i. Un problème pourrait se poser pour les joueurs n'ayant pas gagné un seul match car, dans ce cas, la somme des éléments de la ligne vaut 0 et nous ne pourrons pas diviser par le nombre de matches gagnés et obtenir une matrice stochastique. Pour pallier ce problème, nous allons utiliser un truc introduit dans le calcul du page rank de google. Nous allons utiliser la matrice $B = A + E$ où E est une matrice carrée remplie de 1 divisés par le nombre de joueurs. Il est à noter que la matrice E n'est pas creuse et il est donc indispensable de ne pas l'additionner à A et de ne pas calculer explicitement la matrice B. Remarquez que, malgré que E ne soit pas creuse, le calcul du produit Ev où v est un vecteur peut se faire en temps linéaire puisque c'est simplement la somme des éléments de v divisé par le nombre de joueurs. Finalement, pour le calcul du vecteur propre, nous utiliserons la matrice $\tilde{B}$ qui est obtenue en divisant chaque ligne de B par sa somme, c'est-à-dire 2 pour les lignes correspondant à des joueurs ayant gagné au moins un match et 1 pour les lignes correspondant à des joueurs n'ayant gagné aucun match. A nouveau, tâchez de représenter vos matrices de façon à ne pas les remplir.

0. Une matrice A creuse sera représentée par une structure `struct creuse` avec les champs `startCol` (le vecteur `p` dans le cours), `rows` (le vecteur `i` dans le cours), `values` (le vecteur `x` dans le cours) et l'entier `nz`.
1. Ecrire une fonction C `lecture.c` qui lit le fichier aft.txt dans le répertoire courant et qui donne en sortie deux matrices creuses. La première est la

matrice A vue plus haut où chaque ligne et chaque colonne correspondent à un joueur, et chaque élément (i, j) de la matrice vaut 1 divisé par le nombre de joueurs battus par i si le joueur i a battu au moins une fois le joueur j et 0 si le joueur i n'a jamais battu le joueur j. La matrice contient une ligne de 0 si le joueur n'a gagné aucun match. La deuxième matrice à créer est une matrice M contenant 1 en $M(i, j)$ si le joueur i a battu au moins une fois le joueur j.

2. Ecrire une fonction C `produitMatriceVecteurDense.c` qui prend en entrée une matrice creuse A et un vecteur dense x et qui renvoie le produit Ax sous forme dense.
3. Utiliser la fonction précédente avec la matrice M créée en 1. et un vecteur de 1 afin de calculer un vecteur contenant le nombre de matches gagnés par chaque joueur.
4. Ecrire une fonction C `produitMatriceVecteurCreux.c` qui prend en entrée une matrice creuse A et un vecteur creux x et qui renvoie le produit Ax sous forme creuse.
5. Ecrire une fonction C `puissance.c` qui prend en entrée une matrice creuse et donne en sortie le vecteur propre correspondant à la valeur propre de plus grand module (en dense). Utiliser la méthode de la puissance sur la transposée de $\tilde{B}$ afin de trouver la valeur propre de plus grand module et le vecteur propre correspondant. Normaliser votre vecteur propre pour que sa norme 2 soit égale à 1.
6. Utiliser les composantes du vecteur propre par ordre décroissant afin de déterminer un classement des joueurs.
7. Déterminer une similarité entre votre classement et le classement calculé par l'AFT. Pour ce faire, allez chercher sur le site de l'AFT (`www.aftnet.be`) le classement de quelques joueurs au hasard pour déterminer si le classement calculé par le vecteur propre semble approprié. Dans cette sous-question, il vous est demandé d'être créatif afin de valider (ou de ne pas valider) le classement.

Consignes

Le projet se réalise par groupes de 2. Vous ne devez pas écrire de rapport . Le code sera par contre testé sur une machine unique. Le critère le plus important concerne l'efficacité de votre code et le fait que le code a été écrit par vous.

Une présentation orale de 10 minutes où votre code sera testé est prévue le vendredi 18 décembre. Pour ce faire, votre code doit être soumis sur la plateforme de soumission `submit.montefiore.ulg.ac.be` sous forme d'archive `zip` contenant l'ensemble de vos fichiers ainsi qu'un `makefile` pour le 17 décembre à 19h. Un ordre de passage sera déterminé dans le courant du mois de décembre.

Le plagiat de codes trouvés sur internet ou de codes d'autres groupes d'étudiants sera systématiquement testé et, le cas échéant, résultera en une note de 0/20 pour le projet.