Comprendre l'algorithme d'un moteur de recherche

Fichiers attachés : algo_google.py
Matériel :
- Des dés à 2,3,4,5 et 6 faces pour chacun des groupes
- Des dés à 100 faces pour chacun des groupes

comment un moteur de recherche classe les sites ?

Pour hiérarchiser la pertinence de site web sur un domaine, plusieurs solutions assez naturelles existent :
- On peut faire appel a une sommité (personne ou groupe de personnes) du domaine pour classer les pages. Le classement serait certes très pertinent pour ce domaine mais il faudrait solliciter un grand nombre de spécialistes de différents domaines pour obtenir un algorithme pouvant traiter des requêtes dans tous les domaines possibles et imaginables.
  Il se pose aussi la difficulté de gérer le très grand nombre de pages évoquant une requête (plusieurs millions au minimum) ainsi que l'actualisation fréquente de ce classement.
- Une deuxième solution serait de demander aux internautes eux-mêmes de voter, pour chaque domaine, et de choisir le classement sur la base de ce vote, considérant, que compte-tenu du grand nombre de votants, le classement obtenu serait pertinent.

Ces deux modèles, pourtant intéressants et logiques (ils sont mis en œuvre par wikipédia), font intervenir l'humain et trouvent leur limite dans le trop grand nombre de pages a gérer qui sont de plus évolutives. Ce qui conduit a privilégier des protocoles entièrement automatisables sous forme d'algorithme.
Dans cet optique, Google a cherché un modèle de hiérarchisation qui soit exploitable dans tous les domaines, utilisable pour tous les mots clés, adaptable a un très grand nombre de données, même évolutives, tout en étant automatisable et suffisamment efficace.
C'est en répondant a ce cahier des charges que ce nouveau venu a réussi l'exploit, en quelques mois et malgré l'émergence de Bing ou encore Qwant, a obtenir le quasi-monopole de la rechercher thématique sur le web.
L'idée a la base du modèle de Larry Page et Sergey Brin, fondateurs de Google, revient a attribuer a chaque page un nombre positif entre 0 et 1, appelé score (en anglais "PageRank") de la page, qui caractérisera la pertinence de cette page. Ils proposent alors de déterminer ce score a partir des deux règles suivantes :

R1 : Le score attribué a une page doit être d'autant plus élevé que celle-ci est référencée dans une page faisant autorité (dont le score élevé).

R2 : Le score attribué a une page doit être d'autant moins élevé que celle-ci est référencée dans une page contenant un grand nombre de références.

Leur idée :

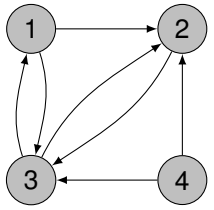
utiliser un surfeur aléatoire.

Principe du surfeur aléatoire

Après avoir fait la liste (sans classement) de tous les sites traitant la requête, le surfeur aléatoire en choisit au hasard un. Puis il s'intéresse aux liens hypertexte du site sur lequel il se trouve vers les autres sites qu'il a listé. Il en choisit alors un au hasard et répète cette opération sans s'arrêter en comptant pour chacun des sites combien de fois il l'a visité.
Les sites sont alors affichés dans l'ordre décroissant de leur nombre de visites.
Ainsi pour un certain mot clé rentré, il s'intéresse aux sites qui évoquent ce mot-clé mais également aux liens hyper-texte qui permettent de passer d'un site à l'autre.

1 *Et concrètement : un premier exemple*

Matériel : un dé à six face
Pour illustrer comment un algorithme de calcul peut être mis en place à partir de ces règles, nous allons prendre l'exemple du classement de quatre pages.
Le problème de l'attribution du score peut être représenté par un graphe orienté : les quatre pages sont représentées par les quatre sommets d'un graphe dont les arêtes orientées représentent les références (liens) pouvant exister entre ces différentes pages.

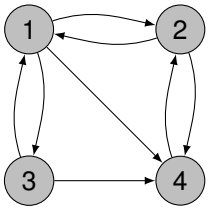


Dans ce graphe, la flèche allant de 1 vers 2 signifie que la page 1 référence la page 2 et l'absence de flèche de 2 vers 4 signifie que la page 2 ne référence pas la page 4.

1. Choisissez un site parmi les 4 qui sera votre point de départ pour tout l'exercice.
2. Comment avec un dé pouvez-vous simuler un déplacement aléatoire de notre surfeur ?
3. Simuler pendant un certain temps le surfeur aléatoire en n'oubliant pas de noter le nombre de fois où il est passé par site.
4. Proposer un classement de ces 4 pages.
5. Comparer votre classement avec les autres groupes.
6. Est ce intéressant de comparer vos effectifs avec ceux des autres groupes ? Que peut-on faire pour remédier à ce problème ? Que remarque-t-on alors ?

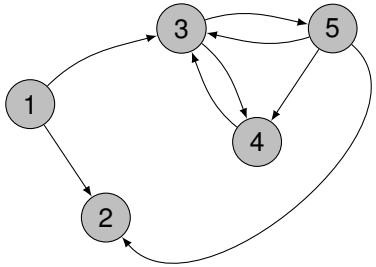
2 *Un deuxième graphe*

Estimer le PageRank des sites représentés par le graphe ci-dessous. Comparer vos scores avec les autre groupes. On effectuera 50 surfs aléatoires.



3 *Un premier problème*

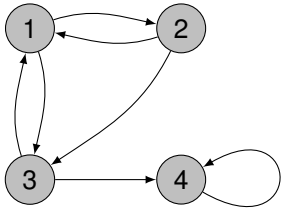
La technique du surfeur aléatoire ne marche pas pour le graphe suivant :



1. Pourquoi ?
2. Proposer une solution pour pallier à ce problème.
3. Faites alors une proposition de classement pour ce graphe après avoir calculé le PageRank.
4. Comparer vos résultats avec les autres groupes.

4 *Puits*

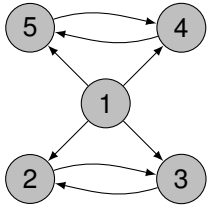
1. Calculer les PageRank du graphe suivant. Comparer le avec les autres groupes.



2. Quel problème moral est soulevé par ce graphe ?
3. Avez vous une solution à proposer pour pallier à ce problème ?

5 *Poche web*

1. Estimer les PageRank des sites suivants. Comparer avec les autres groupes.



2. Quel problème est soulevé par ce graphe ?
3. Avez vous une solution à proposer pour pallier à ce problème ?

Heureusement il y a les maths !

Un théorème complexe d'algèbre linéaire qui peut s'adapter à notre cas : le théorème de Peyron-Froebenius nous dit que si de chaque sommet part un lien vers chacun des autres sommets alors les fréquences des positions au cours de notre surf aléatoire vont toujours converger vers la même valeur et cela indépendamment de la position de départ.

Idée Brillante de L. Page et S. Brin

Préalable : Si une page ne comporte aucun lien vers l'extérieur, comme le sommet 2 du graphe 3, on créé artificiellement un lien de la page vers toutes les autres pages.
Détail de l'idée :

- À chaque étape, on continue la promenade aléatoire précédente avec une probabilité de 0.85 ;
- et avec probabilité de 0.15, on fait un saut aléatoire (vers n'importe quelle page) avec une probabilité $\frac{1}{n}$ de tomber sur une page donnée, où n est le nombre de pages.

On peut démontrer mathématiques que cette façon de procéder permet de faire systématiquement rentrer le graphe considéré dans le champ d'application du théorème de Peyron-Froebenius.
Cela garantit donc la convergence des fréquences vers des valeurs limites qui seront considérés comme étant les pages-Rank. Leur méthode résout au passage le cas des puits car elle empêche de se retrouver dans une poche du web sans pouvoir en sortir.
De plus, plutôt que de calculer les valeurs limites, calculs qui se révèlent dans la pratique très long, l'algorithme du PageRank simule comme l'on vient de le faire un surfeur aléatoire et prend les fréquences trouvés comme estimation des valeurs limites.

6 *Modélisation de leur idée*

Proposer un protocole pour modéliser leur idée avec un dé à 100 faces

7 *Et en pratique*

Matériel : Dés à 6 faces et à 100 faces

1. Grace à la méthode des fondateurs de Google, proposer un classement des pages des quatrième et cinquième graphes.
2. Leur méthode aboutie modifie-t-elle le classement de pages pour lequel il n'y avait pas de problèmes ?

plop

Mise en place de l'activité

Exercice 1

Ce premier exercice a pour but de faire manipuler le surfeur aléatoire aux élèves au moyen d'un dé. Il permet de ré-expliquer la consigne éventuellement à des groupes qui l'aurait mal comprise.
En effet la convergence des fréquences étant assez rapides, avec une trentaine de saut, les différents groupes devraient avoir le même classement. Un groupe qui aurait un classement étonnant est vraisemblablement un groupe qui a mal compris le principe.
D'autre part, sans autres consignes, notamment sur le nombre de surf aléatoire à éxécuter, la comparaison des effectifs n'a que peu de sens (même si l'ordre sera le même). L'idée est de faire émerger que la bonne quantité à comparer est la fréquence de visite de chaque site.

Exercice 2

Après avoir pris un temps de remédiation pour les groupes qui n'avaient pas tout à fait compris le principe du surf, cet exercice ou l'on invitera les différents à donner la fréquence plutôt que l'effectif, permet d'illustrer le principe de l'algorithme. Les fréquences sont très similaires, et cela indépendamment du du site de départ.
Cet exercice permet aussi une deuxième remédiation pour le groupe qui n'a toujours pas compris le principe du surf aléatoire. Il aura en effet des valeurs de fréquence divergentes des autres groupes.

Exercice 3

Cet exercice présenter le premier écueil de l'algorithme naïf : que faire quand un site n'a aucun lien vers d'autres sites.
Assez spontanément les élèves ont tendance à rajouter des flèches vers les autres sites. Cette solution : le surf équiprobable, permet de solutionner cette difficulté.

Exercice 4 et 5

Ces deux exercices présentent le principal écueil de l'algorithme : le blocage dans un puit ou une poche du web.
Même si les élèves peuvent avoir de nombreuses idées pour s'en sortir, la solution choisie par Google ne peut être trouvé à cause de la trop grand difficulté mathématique : les chaines de markov et le théorème de Peyron-Froebenius se trouvant en dehors de leur connaissance.
C'est le moment de leur présenter la solution mise en place par google.

6 et 7

Ces exercices peuvent être à traiter en classe, si il reste du temps ou à la maison.
Il est également judicieux de faire simuler le surfeur aléatoire à l'aide du fichier algo_google.py