
8INF803 Bases de données réparties AUTOMNE 2017

Devoir 1

25 % de la note finale.

Date de remise : 17 Octobre 2017 (Peut changer pour plus tard)

Remise : edmond.lachance@gmail.com OU

edmond_lachance@uqac.ca

Lien Skydrive, Lien Google Drive

Remise : **Document en PDF ou HTML** (Pas de DOCX svp, j'ai pas Word)

Screenshots de l'exécution + fichiers de code source.

(Les pièces jointes sont souvent détruites par gmail...). Gmail est devenu très stricte sur le contenu permis. Il est conseillé de me donner un lien vers un cloud (Google Drive, SkyDrive etc).

Équipes : 1 à 4 personnes

Exercice 1 : Le sort du dernier recours.

Mise en situation :

Vous êtes un cuisinier appelé Pito, ex-magicien, sorcier, et propriétaire de votre propre restaurant appelé "Les pain pitas garnis de Pito". Votre restaurant est situé sur le chemin de la forêt DragonWood, une forêt peuplée par des elfes sylvains et également domicile à un puissant dragon vert.

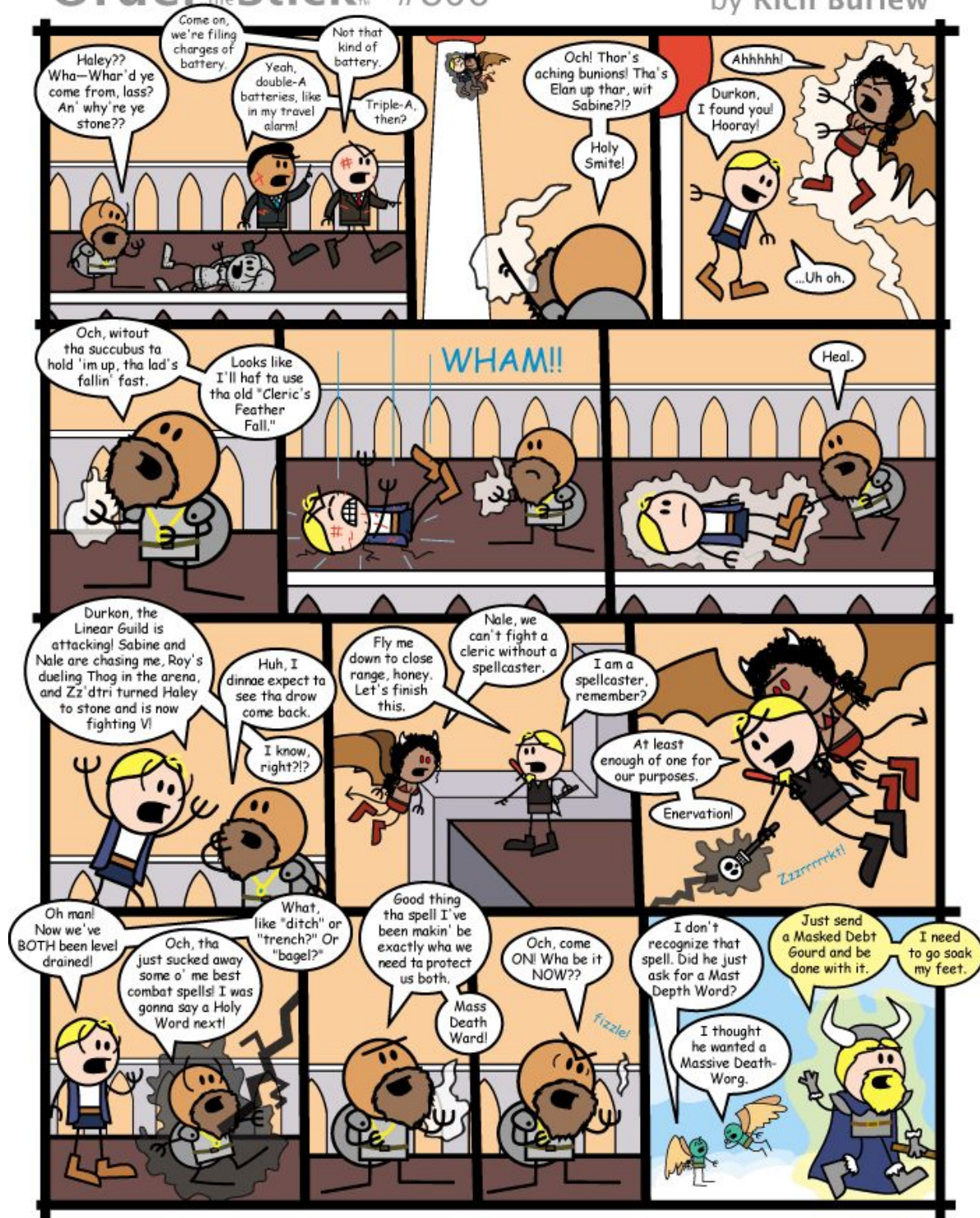
Vous avez choisi un endroit magnifique pour y établir votre restaurant. Par contre, il y a peu de passage et les elfes ne sont pas trop fan de la cuisine.

Faire de l'argent avec un commerce n'était peut-être pas la meilleure idée. J'aurais peut-être dû rester aventurier...

Un beau jour, pendant la préparation d'une nouvelle recette de pain pita, Pito se fait attaquer par une troupe de Halflings Chaotic Neutral (CN). Ces halflings étaient là pour les richesses du Dragon Vert, mais finalement n'étant pas de taille, ils s'en prennent plutôt à un restaurateur honnête!! Ne soupçonnant pas du tout les halflings, Pito est pris par surprise et capturé. Il a maintenant les deux mains liées et il est attaché à une poutre. Il doit s'échapper!

Les halflings quittent l'auberge. Ils mettent le feu. Pito est encore attaché. Normalement, quand Pito lance un sort, la plupart du temps, il doit agiter ses mains afin de satisfaire certaines composantes somatiques. Il peut également avoir besoin de certains petits-ingrédients qui servent de catalyseur pour lancer le sort. Cette fois, tout ce que Pito peut faire, c'est parler. Pito va brûler vivant s'il ne trouve pas une solution. Il n'a pas la force ou la dextérité pour se défaire de ses liens. Il plonge donc dans sa mémoire pour essayer de trouver un sort qui peut le tirer d'affaire. S'il a déjà vu le sort en action, ou alors vu le sort dans un grimoire ou parchemin, il a confiance qu'il peut le reproduire.

Votre tâche est de trouver un sort qui peut tirer Pito d'affaire (avec la programmation). Pito est capable de lancer des sorts de maximum niveau 4. Outre trouver un sort, vous devez accomplir les énoncés suivants.



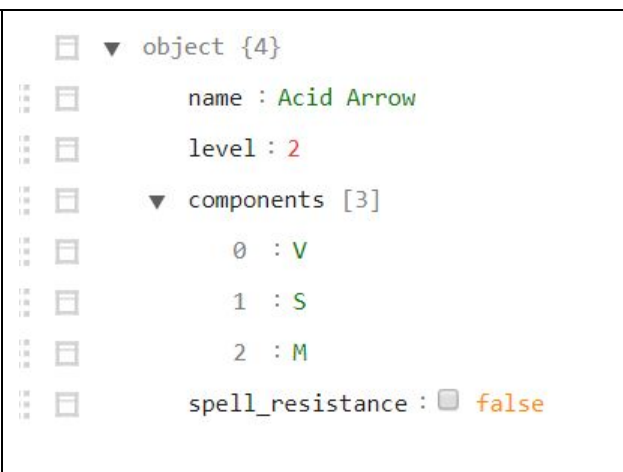
1.Crawler

Vous devez utiliser le **langage de votre choix (C++, Scala, JS, Java, Python)** pour télécharger tous les sorts du Wizard/Sorcerer. Voici quelques sites qui donnent une telle liste (voir plus bas).

Vous pouvez également décider de télécharger tous les sorts (pour toutes les classes) et faire le tri après avec votre Map/Reduce. Le choix est le votre.

Vous devez insérer les sorts dans une collection MongoDB.

Voici un exemple de JSON inséré. Vous devez utiliser le même schéma au minimum. Pour le level du sort, il peut varier. Prenez n'importe lequel. Si c'est un sort du Wizard, prenez le level du sort pour le Wizard.

<pre>{ "name": "Acid Arrow", "level": 2, "components": ["V", "S", "M"], "spell_resistance": false }</pre>	
---	---

Sites qui contiennent le data :

[Archives of Nethys](#)

[Pathfinder Main](#)

[DXContent](#)

[Pathfinder #2](#)

Note sur DXContent :

http://www.dxcontent.com/SDB_SpellBlock.asp?SDBID=1543

SDBID=1 a 1500-1600

Par contre, c'est la liste de tous les spells. Il faut faire 100 % du tri avec le map (ce qui est correct).

Voir aussi ces outils :

<http://www.d20pfsrd.com/magic/tools/advanced-spell-search/> (Un exemple d'application pour chercher dans les spells)

<http://regexr.com/> (Si vous utilisez des regex)

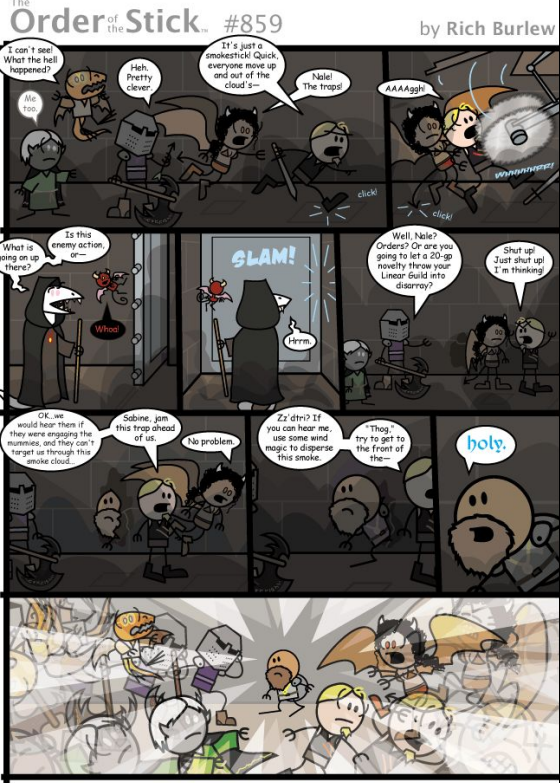
<http://www.striphtml.com/> (Pour enlever les tags HTML (le faire en code si il faut enlever les tags HTML bien entendu)

2. Trouver le/un sort qui libère Pito. Il faut que ça soit un sort de Wizard, verbal seulement, maximum niveau 4. Exemple de sort verbal seulement : **Holy Word** (voir plus bas).

Vous devez écrire un code en MapReduce qui génère la liste des spells valables.

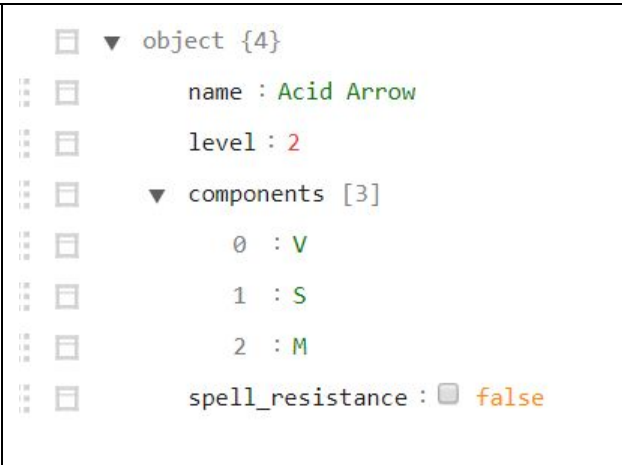
Exemple de code :

<https://gist.github.com/mitchi/18a9ad3aaf084823a97807f8eeb89a7c>

 <p>©2012 Rich Burlew www.GiantITP.com</p>	<p>Holy Word School evocation [good, sonic]; Level cleric/oracle 7, inquisitor 6 Casting Casting Time 1 standard action Components V Effect Range 40 ft. Area nongood creatures in a 40-ft.-radius spread centered on you Duration instantaneous Saving Throw Will partial; Spell Resistance yes</p>
--	--

Comic sur cette image: [The Order of the Stick](#)

3. Vous devez également envoyer les données dans une BD SQLite. Créez un schéma avec une table qui a a peu près la même structure que votre JSON (Voir image).

<pre>{ "name": "Acid Arrow", "level": 2, "components": ["V", "S", "M"], "spell_resistance": false }</pre>	
---	--

Ensuite, vous devez écrire une requête SQL qui va produire la même liste de spells qu'à l'énoncé 2.

Voici un exemple de code pour se connecter à SQLite en Node.JS :

```
var sqlite3 = require('sqlite3').verbose();
var db = new sqlite3.Database(':memory:');

db.serialize(function() {
  db.run("CREATE TABLE lorem (info TEXT)");

  var stmt = db.prepare("INSERT INTO lorem VALUES (?)");
  for (var i = 0; i < 10; i++) {
    stmt.run("Ipsum " + i);
  }
  stmt.finalize();

  db.each("SELECT rowid AS id, info FROM lorem", function(err, row) {
    console.log(row.id + ": " + row.info);
  });
});

db.close();
```

Points Bonus si votre schéma est un peu plus complexe (Plusieurs tables)

Exercice 2 : Calculer le pagerank avec plusieurs itérations MapReduce.

Cet exercice est beaucoup plus classique. Vous devez coder l'algorithme du pagerank avec MapReduce. MapReduce n'est pas le framework idéal pour faire ce genre de calcul itératif (On va voir GraphX plus tard), mais il marche bien néanmoins.

Les données des spells du Wizard ne sont pas assez interconnectées pour mériter de faire un PageRank. Nous allons donc prendre un graphe beaucoup plus simple.

Exemple de code (BFS) :

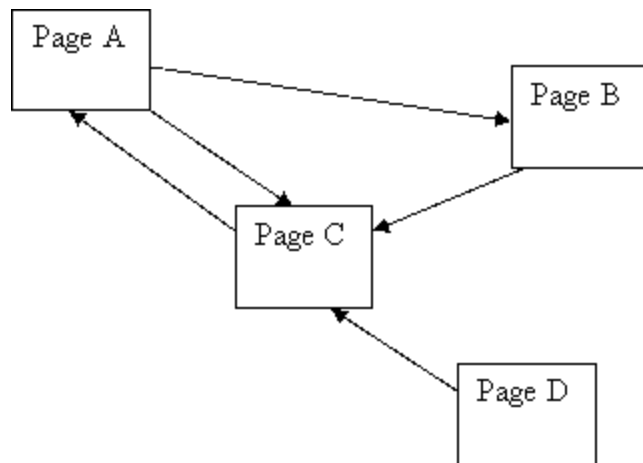
<https://gist.github.com/mitchi/18a9ad3aaf084823a97807f8eeb89a7c>

Liens pertinents :

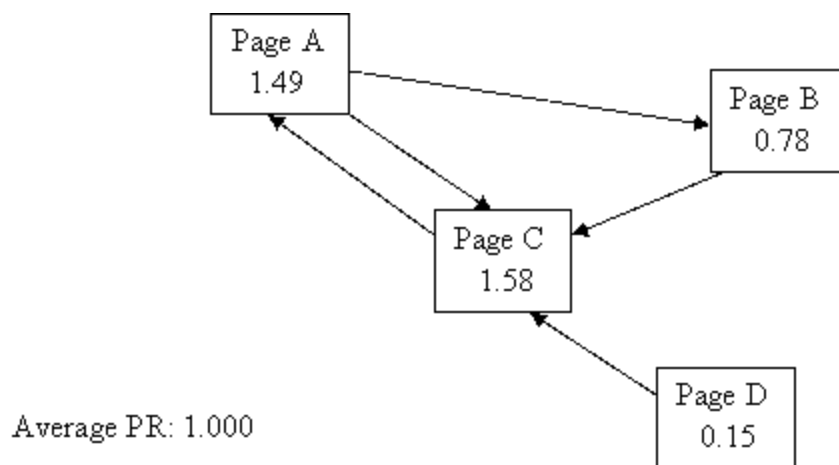
1. <http://pr.efactory.de/e-pagerank-algorithm.shtml>
2. <https://courses.cs.washington.edu/courses/cse490h/08au/lectures/algorithms.pdf>
3. <http://www.cs.princeton.edu/~chazelle/courses/BIB/pagerank.htm>
4. <http://www.sirgroane.net/google-page-rank/>
5. <http://hadooptutorial.info/mapreduce-use-case-to-calculate-pagerank/>

Faites attention, il faut qu'il y ait 2 valeurs dans le tableau des valeurs pour que le reducer d'une clé soit appelé.

Voici le graphe que vous allez utiliser pour le test :



Après 20 itérations, vous aurez les résultats suivants. Le Damping Factor utilisé est 0.85.



Voici ma sortie (pour référence). J'ai screenshot les deux dernières itérations.

```
*****
*****
*****
VALEUR DE I = 19
[ { _id: 'A',
  value: { pagerank: 1.4901259564203881, adjlist: [Object] } },
  { _id: 'B',
    value: { pagerank: 0.7832552713203321, adjlist: [Object] } },
  { _id: 'C',
    value: { pagerank: 1.57661877225928, adjlist: [Object] } },
  { _id: 'D',
    value: { pagerank: 0.15000000000000002, adjlist: [Object] } } ]
*****
*****
*****
VALEUR DE I = 20
[ { _id: 'A',
  value: { pagerank: 1.4901259564203881, adjlist: [Object] } },
  { _id: 'B',
    value: { pagerank: 0.7833035314786649, adjlist: [Object] } },
  { _id: 'C',
    value: { pagerank: 1.5765705121009472, adjlist: [Object] } },
  { _id: 'D',
    value: { pagerank: 0.15000000000000002, adjlist: [Object] } } ]
Fin du programme!!!
(node:10604) UnhandledPromiseRejectionWarning: Unhandled promise rejection (rejection id: 2): MongoError: topology was destroyed

Process finished with exit code 0
|
```