8INF803 Bases de données réparties Hiver 2018
Devoir 2
25 % de la note finale.

Date de remise : 24 Avril 2018 (Peut changer pour plus tard)

Remise : edmond.lachance@gmail.com OU

edmond_lachance@uqac.ca

Lien vers le devoir 1

Lien Skydrive, Lien Google Drive

Remise : **Document en PDF ou HTML** (Pas de DOCX svp)

Screenshots de l'exécution dans un document

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

Bonus possibles. Les travaux avec des bonus >100%, l'excédent va sur la note finale (donc c'est pas perdu)

Bonus 1.Visualisation d'un graphe de jeu avec yedEditor, d3.js, Gephi, GraphStream ou GraphViz (pour donner quelques exemples).

Voici 2 exemples de visualisation avec Yed :

Exemple 1 (Itérations de coloriage). Généré depuis Spark GraphX (On écrit dans un XML à chaque itération)

Exemple 2 (36mb PNG)

Exemple 2 (version SVG)

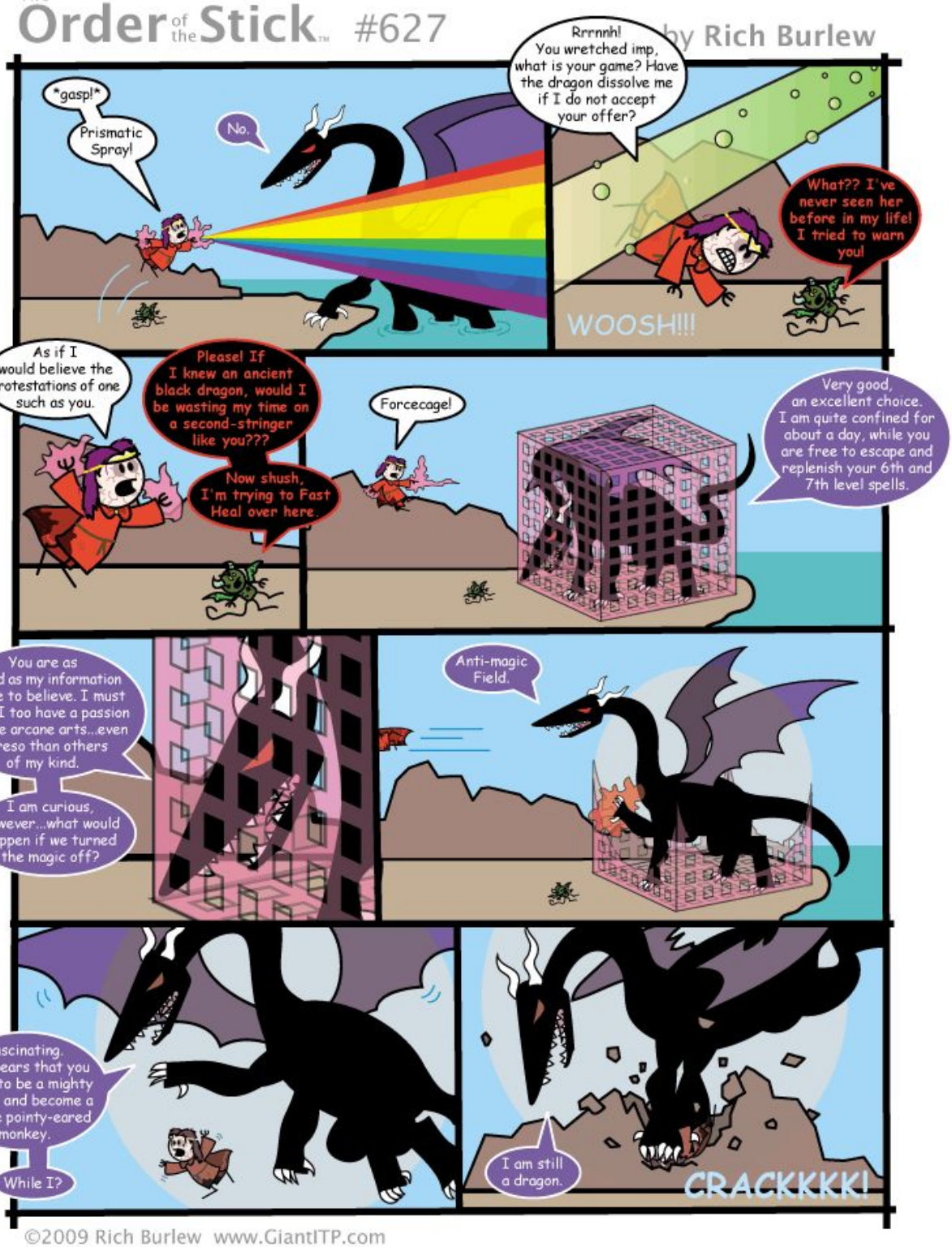
Bonus 2. Exécution sur un Cluster d'au moins 2 machines en réseau.

Tutoriel Cluster ici

Si vous faites exécuter sur un cluster, me montrer des screenshots qui montrent que tout calcul sur le cluster (ou alors une vidéo).

The Order of the Stick #627

by Rich Burlew



Exercice 1 : La guérison

Dans le dernier TP, vous avez trouvé un sort (verbal seulement) pour sortir Pito de son pétrin. Pito a utilisé le sort Dimension Door pour s'enfuir du feu. Cependant, Pito a été brûlé sévèrement par l'incendie et il va mourir de ses blessures s'il n'y a personne pour l'aider. Et Pito ne connaît aucun sort qui puisse l'aider dans la situation présente.
C'est à ce moment que le pouvoir de la bloodline de Pito se met en action. Ce pouvoir (que vous incarnez encore une fois de plus) va permettre à Pito de se concentrer et d'accéder à un grand pouvoir.
Ce grand pouvoir va lui permettre d'accéder à un savoir incroyable. Pito va pouvoir visualiser tous les sorts de soins ainsi que les noms des créatures qui possèdent ces derniers.
Une fois qu'il a vu tout ça, il conjure une créature qui peut le soigner. Cette créature, un Ange Solar (Pito n'a jamais entendu parler d'eux) soigne les brûlures de Pito et il peut finalement relaxer. Il part ensuite faire une sieste dans son hamac (qui lui n'a pas brûlé) et il contemple la suite de sa vie ainsi que ce qui vient de se passer. L'Ange, qui est toujours là, ne repars pas tout de suite.

Ce que vous devez faire :

1. Vous devez crawler le bestiaire de Pathfinder pour des créatures. Vous récupérez le texte brut des créatures à chaque fois. Ensuite, le texte brut vous pouvez déjà l'organiser un peu. Vous pouvez mettre une créature dans un schéma SQL (Spark supporte les DataFrames). Vous pouvez aussi faire un JSON (ensuite ça s'importe dans une DataFrame) ou alors juste faire un objet Scala avec. Les RDD de Spark peuvent être de n'importe quel type. Donc si vous faites un objet Créature dans Scala, vous pouvez avoir un RDD de Créatures RDD[Créature]
2. Une fois que vous avez les données des créatures vous allez mettre ces données dans un RDD Spark. Le RDD Spark est un conteneur qui est parallélisé sur le cluster. Une fois qu'on a un RDD, toutes les opérations que vous faites dessus sont faites en parallèle.
3. Maintenant que vous avez un RDD, vous pouvez faire des actions/transformations sur celui-ci. Vous allez utiliser l'API des RDD de Spark (entre autre) pour créer une batch view comme celle-ci (voir table en bas). Cette Batch View sera très utile pour Pito, car il va pouvoir visualiser rapidement les créatures qui peuvent le tirer d'affaire et faire son choix. Au niveau de l'architecture Lambda, cette Batch View irait dans une BD de Serving Layer (probablement une BD distribuée key-value).

Voici quelques fonctions de l'API RDD qui seront sûrement utiles :
Map, flatmap (voir lien), reduceByKey.
ReduceByKey et flatMap ressemblent beaucoup au mapReduce sur MongoDB. On peut utiliser

reduceByKey quand on a un PairRDD. Pour avoir un pairRDD, il faut que chaque élément du RDD soit un Tuple2.

Exemple de Ba
tch View (le RDD final après les traitements)
La dernière ligne n'est pas un sort de soins.

Spell (key)	**Créatures qui ont le sort**
Cure light wounds	Solar, Planetar , Lillend
Cure moderate wounds	Solar, Planetar
Cure serious wounds	Movanic Deva, Monadic Deva, Bralani
Cure critical wounds	Solar, Planetar
Heal	Solar, Planetar Ghaele
Mass Heal	Solar
Banishment	Solar, Planetar

Note : Bonus si vous faites ça pour tous les sorts(pas juste les sorts de healing), pour toutes les créatures! Ça fait un peu plus de travail au niveau du Crawler, mais ensuite c'est pareil.

Ensuite, une fois que votre RDD est complet. Vous pouvez le sauver sur votre disque avec **saveAsTextFile**(*path*) ou une autre façon.

Vous pouvez également collecter le RDD sur votre driver program et ensuite le sauver sur le disque, l'envoyer dans une BD etc. Le RDD devient un Array[type] lorsqu'il est collecté sur le driver program.

Vous devez sauver les infos suivantes sur chaque créature :

- Nom de la créature
- Liste des sorts préparés ou spontanés. Vous pouvez mettre ça dans le même array.

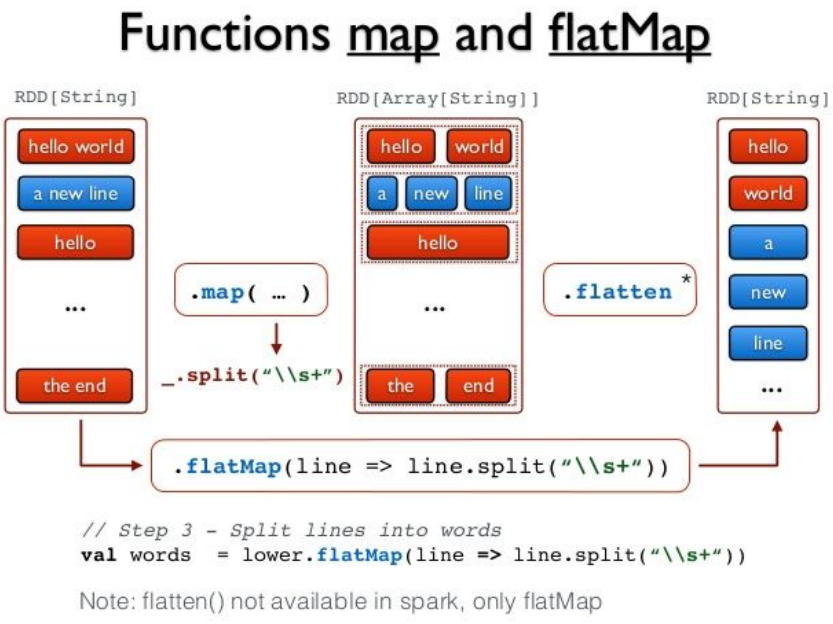
Voici un exemple en JSON.

```
{
   "name":"Solar",
   "spells":[
      "cure light wounds",
      "cure moderate wounds",
      "cure serious wounds",
      "cure critical wounds",
      "heal",
      "mass heal"
   ]
}
```

Et voici un exemple avec une classe Scala :

```
class creature(val name : String) extends Serializable {
  var spells =  ArrayBuffer[String]()
 def addspell(spell : String) : Unit = {
    spells += spell
  }
}

var solar = new creature("Solar")
solar.addspell("cure light wounds")
solar.addspell("heal")
```



Exemple de stratégie possible pour le crawling :

Chacun des livres de Beastiary (Bestiaire) a une page. Sur la page, il y a des liens vers la fiche de chaque créature.
Donc vous pouvez aller sur chacune des pages de Bestiaire (il n'y en a que quelques unes) et vous crawler tous les liens que vous trouvez.
Pour chaque créature vous prenez son nom et vous allez prendre la liste de tous ses sorts.

Ensuite avec Spark et les transformations/actions sur les RDD, vous faites la Batch View spell-> liste de créatures.

http://paizo.com/pathfinderRPG/prd/bestiary/monsterIndex.html
http://paizo.com/pathfinderRPG/prd/bestiary2/additionalMonsterIndex.html

Spell-Like Abilities (CL 20th)

Constant—*detect evil*, *detect snares and pits*, *discern lies* (DC 21), *true seeing*

At Will—*aid*, *animate objects*, *commune*, *continual flame*, *dimensional anchor*, *greater dispel magic*, *holy smite* (DC 21), *imprisonment* (DC 26), *invisibility* (self only), *lesser restoration*, *remove curse*, *remove disease*, *remove fear*, *resist energy*, *summon monster VII*, *speak with dead* (DC 20), *waves of fatigue*

3/day—*blade barrier* (DC 23), *earthquake* (DC 25), *heal*, *mass charm monster* (DC 25), *permanency*, *resurrection*, *waves of exhaustion*

1/day—*greater restoration*, *power word blind*, *power word kill*, *power word stun*, *prismatic spray* (DC 24), *wish*

Spells Prepared (CL 20th)

9th—*etherealness*, *mass heal*, *miracle*, *storm of vengeance* (DC 27)

8th—*fire storm* (DC 26), *holy aura* (2) (DC 26), *mass cure critical wounds* (2)

7th—*destruction* (DC 25), *dictum* (DC 25), *ethereal jaunt*, *holy word* (DC 25), *regenerate*

Ensuite au niveau du code source, vous avez la situation suivante :

```
href="/pathfinderRPG/prd/coreRulebook/spells/consecrate.html#consecrate" >consecrate</a></i>, <i><a
href="/pathfinderRPG/prd/coreRulebook/spells/cureModerateWounds.html#cure-moderate-wounds" >cure moderate wounds</a></i> (2), <i><a
```

Donc, chaque fois que dans un href on voit /spells/ on peut être sûr à 100 % que c'est un lien vers un spell. Et donc on récupère "Cure Moderate Wounds" comme un spell pour cette créature.

Exercice 2 : Protéger Pito et son village natal

Pito dort d'un très grand sommeil tandis que l'Ange veille sur lui avec intérêt. Cependant, la visite d'un ange aussi puissant ne passe pas inaperçue. Le méchant dragon vert de la forêt avoisinante décide d'envoyer une patrouille d'Orcs voir ce qui se passe car il a détecté une présence très puissante et il n'aime pas ça!! Ses Orcs courent pendant toute la nuit. Ils trouvent d'abord les traces des halflings et très rapidement ils sont sur le chemin de l'Auberge de Pito.

Exemple de code Spark GraphX pour colorier un graphe

Dans cet exercice, vous allez devoir simuler deux combats. Ces combats vont vous permettre d'utiliser le module GraphX de Spark. GraphX permet de faire des calculs itératifs sur un graphe distribué. GraphX offre plusieurs styles de programmation :

1. API Pregel
2. AggregateMessages suivi d'un join Messages/Sommets

La version 1 implémente l'API Pregel. Un paradigme de programmation qui existe sous plusieurs noms. La variante Pregel a été proposée par Google.

La version 2 utilise AggregateMessages. Je vous conseille plutôt cette version. l'API est plus récent, fait moins de choses, mais il fait le plus important : Il génère et fusionne les messages de notre graphe. Ensuite, une fois les différents messages envoyés, il s'agit de faire un join avec notre VertexRDD et ensuite choisir un nouvel état pour notre vertex à partir de nos messages.

Combat 1. Solar vs Éclaireurs Orcs

Le Solar détecte les orcs approcher et décider de s'occuper d'eux sans réveiller Pito. Le Solar se bat sans retenue mais si certains Orcs sont apeurés, ils les laisse fuir.

Le combat 1 va vous permettre de tester et apprendre la plateforme Spark GraphX. Le solar a la feuille de personnage suivante :
http://www.d20pfsrd.com/bestiary/monster-listings/outsiders/angel/solar
Votre travail est d'implémenter une petite intelligence artificielle qui permet à votre Solar (un sommet du graphe) d'utiliser ses capacités de défense, d'attaque et ses spells contre les Orcs. Je sais que le Solar possède énormément de fonctions. C'est pour cela que vous allez implémenter le minimum de choses et ensuite, si tout va bien, vous pouvez ajouter des choses une par une.

Les trucs que vous pouvez implémenter pour le solar (pas une liste complète) :

- Ses attaques. Le solar peut attaquer avec son **arc** les Orcs qui sont à 110 ft (pieds) de lui. Il peut attaquer plus loin bien sûr, mais contentez vous de cette distance. Après tout, il y a quand même une forêt autour.
- Le solar peut également attaquer avec son **épée quand ils sont à côté (5ft ou 10ft)**
- Régénération (15/tour)
- Son armure (44)
- Son HP

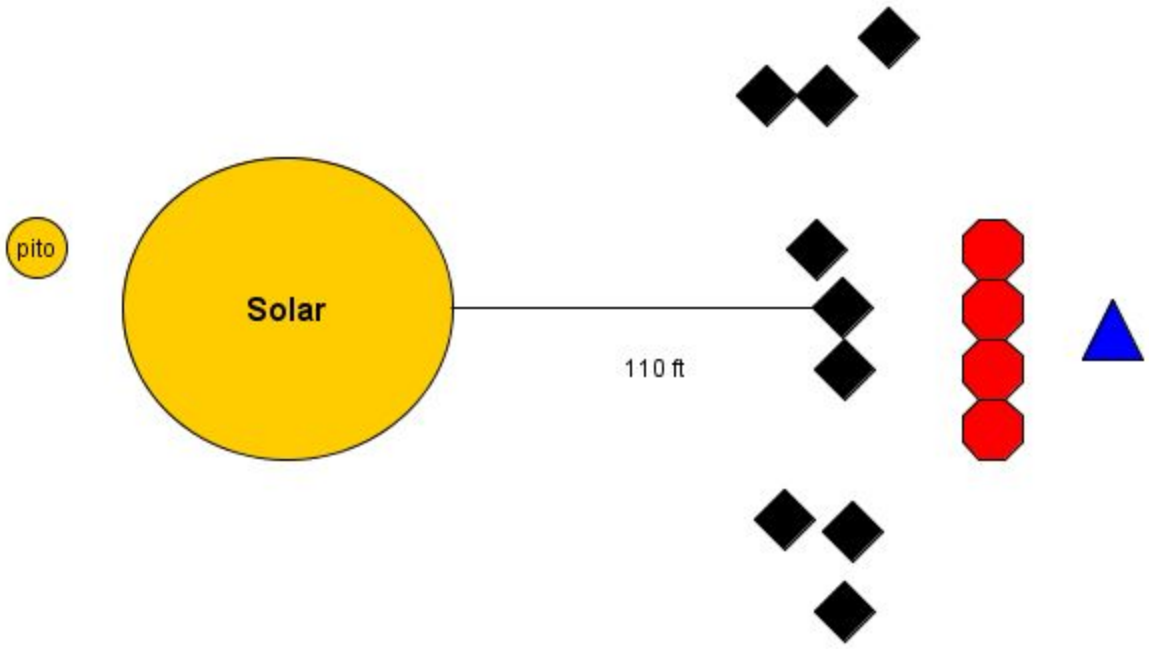
Pour les Orcs (pas la liste complète bien entendu)

- Armure
- HP
- Attaques

Le party des Orcs :
9x Worgs Rider (en noir)
1x Le Warlord (en blanc)
4x Barbares Orc (en rouge)

Si c'est trop facile, remplacez le Warlord par lui

Pour le combat 1, Imaginez vous le graphe suivant.



Question ouverte : Comment gérer efficacement un système de distance 3D avec un graphe d'agents distribué? Avoir un système de distance 3D permet d'avoir accès aux attaques de zone, le vol, les arcs et flèches etc.
On met la distance sur le sommet? sur l'arête? problèmes de collision etc.

Mécaniques du jeu.

Au niveau de la structure de donnée Graphe, on peut dire que le Solar a une arête avec tout le monde.

Attaques : Il y a deux parties pour attaquer dans ce jeu. Il faut toucher en premier, et ensuite si on touche, on peut faire les dégâts. Pour toucher une créature, il faut égaliser ou battre son armure (AC).

Comme exemple, on va prendre le Solar qui attaque le Worg Rider avec son épée.

Solar
Melee *+5 dancing greatsword* +35/+30/+25/+20 (3d6+18)

Il faut lire cette description d'attaque de la façon suivante. Le Solar a une épée magique +5 dancing. Il a 4 attaques. Chaque attaque (après la première) est un peu moins précise. Les dégâts sont toujours 3d6+18 quand l'attaque touche.

Donc pour toucher le Worg Rider il faut battre son armure (18)

Première attaque d'épée:
Toucher : 1d20 (entre 1 et 20) + 35 >= 18? Si oui, on fait les dégâts (3d6+18)
Puisque les dégâts sont > hp de l'Orc Worg Rider, ce dernier est éliminé du combat.

Condition d'arrêt. Plusieurs façons de détecter :

- Plus de messages envoyés (driver program) OU
- Tous les orcs sont éliminés (inspecter ça avec le driver program) OU
- Variable accumulateur inspectée dans le driver program. Les variables d'accumulateur fonctionnent juste quand on utilise une action Spark. Take(1) par exemple.

Combat 2. Les Orcs et le dragon vert attaquent le village de Pito

Le dragon vert perd son lieutenant préféré aux mains du Solar. À l'aide d'un spell de Scrying (espionnage) sur son lieutenant, le dragon vert a vu la bataille et est au courant de tout. En quête de vengeance contre le Solar, le dragon vert décide que le meilleur moyen de revoir le Solar est de s'en prendre à Pito. Pendant une année entière, le dragon utilise son réseau d'espions et d'hommes de main pour finalement trouver et attaquer le village natal de Pito. Le dragon vert participe lui-même à l'attaque.

Par chance, Pito visitait ses parents. Pito appelle son ami et le Solar descend de son plan pour sauver le village. Quand le Solar voit l'armée qui attaque, il utilise sa plus puissante magie (Miracle) pour faire téléporter ses alliés Anges : 2 planetar, 2 movanic deva et 5 Astral Deva.

Voici les trucs que vous devez programmer dans l'IA du Solar et de ses alliés:

1. Le Solar et ses alliés s'envoient des messages à chaque itération. Chaque sommet peut également s'envoyer un message à lui-même. Les messages servent à communiquer des informations ou des demandes aux autres anges. Par exemple, un ange blessé peut envoyer un message demandant un "heal". Ou alors, un ange peut s'envoyer un message à lui-même lorsqu'il voit, en envoyant ses messages, qu'un de ses alliés est blessé. Il y a plusieurs façons différentes de fonctionner.

Voici le workflow :

Graphe Initial (Team1, Team2)

(Boucle)
On envoie les messages. Chaque arête fait donc son "sendMsg". Lorsqu'on fait le sendMsg, on est dans la vision limitée, locale, d'une arête. On peut envoyer un message à la source, ou à la destination d'une arête seulement.

Une fois que tous les messages sont envoyés, il faut faire la fusion de ceux-ci. Est-ce qu'on détruit des messages ou alors est-ce qu'on les garde tous en compilant une liste de ceux-ci, pour le reduce par la suite?? Ça dépend du problème!

On join (joinVertices) ensuite les messages avec les données des sommets. Ensuite, en ayant accès aux données du sommet, ainsi qu'au message, on doit prendre une décision.
Est-ce qu'on change nos états à cause des messages, ou alors on garde le même état?

Team 1 :
Solar
2x Planetar
2x Movanic Deva
5x Astral Deva

Team 2 :
1x Green Great Wyrm Dragon
200x Orc Barbarians
10x Angel Slayer

Consignes

Le Dragon Vert a plusieurs stratégies possibles :

1. Il se déguise en villageois avec alter self. Une fois proche proche du Solar, il révèle sa vraie forme et essaie de le tailler en pièces avec une Full-attack de ses attaques naturelles. S'il ne réussit pas à tuer le Solar en 1 tour, il prend son envol et adopte la stratégie #2
2. Il laisse ses soldats se battre. Il vole au travers du champs de bataille et utilise sa breath weapon 24d6 d'acide. Il touche toujours 3 anges au hasard avec son attaque d'acide. Les anges peuvent faire un jet de Reflex pour éviter.
3. Si le Dragon vole, les anges doivent l'attaquer avec un arc car il vole trop vite.
4. Il peut utiliser Power Word Stun sur un Ange qui a <150 hp
5. Summon Monster 7 va toujours summoner un Dire Tiger

Également :

6. Les Orc Barbarians sont un peu low-level, mais ils sont beaucoup et ils touchent automatiquement quand ils ont un 20 sur leur dé.
7. Les Angel Slayer se positionnent pour tirer des flèches sur les anges. Si un Ange approche en corps à corps, le Angel Slayer switch pour sa hache.
8. Les Angel Slayer n'utilisent pas leur heal, à part sur le Dragon Vert (s'il le demande)

Les Angel Slayer ont +8 aux attack rolls et +8 damage rolls contre les Anges.
Ils ont 3 attaques avec leur bow et 6 attaques avec leur double hache.

La state machine du Solar peut ressembler un peu à ça :

- Soigner un allié. (Enlever le sort de soin des sorts disponibles une fois qu'il est utilisé).
- Attaquer avec son Arc (ne pas implémenter Slaying Arrow)

- Attaquer avec son Épée (Peut attaquer plusieurs adversaires pour chaque attaque lorsqu'il attaque les orcs barbarians)
- Utilise son sort : Mass Heal quand tout le monde est blessé
- Peut utiliser Summon Monster 7 pour amener un Tigre

Lorsque l'état est changé, il faut recommencer tout le processus. Générer des messages, Changer d'état.

Des trucs à ne pas oublier sur le Solar :

- Damage Reduction de 15 (Aucun adversaire de cette bataille n'est capable de la briser je crois)
- Regeneration 15
- 4 flèches ou 4 attaques à l'épée
- Résistance à la magie de 34

Rappel Bonus:

Énorme bonus sur la note finale du cours si vous implémentez un système de mouvement 3D (X,Y,Z)