Projet Java et Algorithmique : Projet Risk'ISEP

Jérémie Sublime

<2017/03/07>

Table des matières

1 Sujet **2**

2 Description des fonctionnalités **2**
2.1 Objectifs du jeu 2

3 Déroulement d'une partie **2**
3.1 Initialisation du jeu 2
3.2 Déroulement d'un tour 3
3.2.1 Réception de renforts 3
3.2.2 Déplacements et attaques 4
3.2.3 Batailles 4
3.3 Tableau des caractéristiques de unités 5

4 Travail attendu **6**
4.1 Fonctionnalités attendues 6
4.2 Fonctionnalités facultatives 6

5 Livrables **6**

6 Concours d'IA **7**

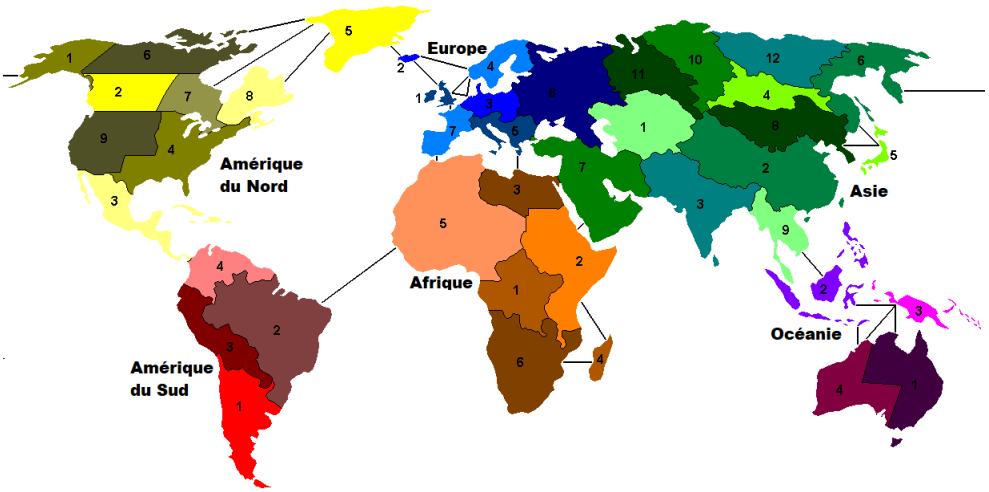
1 Sujet

L'objectif de ce projet est de concevoir, implémenter et tester un jeu vidéo écrit en Java. Il s'agira de coder un jeu de type "Risk" jouable pour 2 à 6 joueurs.

Les groupes de projet auront au maximum 3 membres !

2 Description des fonctionnalités

Risk est un jeu de stratégie se jouant sur un plateau contenant 42 territoires répartis dans 6 régions, dans lequels les 2 à 6 joueurs jouent chacun leur tour jusqu'à accomplir leurs objectifs de jeu.



2.1 Objectifs du jeu

Les objectifs du jeu sont divers et dépendent de la mission secrète que chaque joueur aura reçue au départ : détruire toutes les unités d'un autre joueur, contrôler X territoires, contrôler 2 ou 3 régions, etc. S'il ne parvient pas à accomplir sa mission initiale, chaque joueur peut décider à la place de conquérir la planète.

3 Déroulement d'une partie

Une partie de Risk se déroulent en deux phases principales décrites ci-après :

— La phase d'initialisation du jeu durant laquelle les joueurs récupèrent leur mission et installent leurs unités.

— La phase de jeu durant laquelle les joueurs jouent chacun leur tour jusqu'à ce la condition de victoire de l'un d'entre eux soit remplie.

3.1 Initialisation du jeu

1. Une mission secrète est attribuée à chaque joueur (les autres joueurs ne regardent pas la mission)
2. Les territoires sont distribués aléatoirement entre les joueurs.
3. Chaque joueur reçoit ses armées :
   — 40 armées par joueur à 2

— 35 armées par joueur à 3
— 30 armées par joueur à 4
— 25 armées par joueur à 5
— 20 armées par joueur à 6

4. Les joueurs placent chacun leur tour leurs armées sur leurs territoires. Un minimum d'une armée doit être placée sur chaque territoire.

TABLE 1 – Tableau des missions

Mission	Conditions
Détruire le joueur X	3 joueurs et plus
Conquérir tous les territoires	2-3 joueurs
Contrôler 3 régions et au moins 18 territoires	-
Contrôler 18 territoires avec au moins 2 armées	3 joueurs et plus
Contrôler 30 territoires	2-3 joueurs
Contrôler 24 territoires	4-5 joueurs
Contrôler 21 territoires	6 joueurs
Contrôler la plus grosse région + 1 autre région	-

3.2 Déroulement d'un tour

Chaque tour de jeu d'un joueur se déroule en 2 phases :

1. Réception de renforts
2. Déplacements et attaques

3.2.1 Réception de renforts

Chaque joueur reçoit un nombre d'armées égales au nombre de territoires qu'il possède divisé par 3 et arrondi à l'entier inférieur. Par exemple, un joueur contrôlant 14 territoires recevra 4 armées. De plus, si le joueur contrôle une ou plusieurs régions, il recevra des armées supplémentaires : pour chaque région, le nombre d'armées supplémentaires est égal au nombre de territoire de la région divisé par 2 et arrondi à l'entier inférieur. Par exemple pour une région de 4 territoires : 2 armées supplémentaires seront reçues.

A ces renforts peuvent venir se rajouter les renforts acquis lors de la capture de territoires lors des précédents tours (voir section suivante et Tableau 2).

Une fois la somme totale faite (territoires + régions + captures), si celle-ci est inférieure à 2 armées, le joueur concerné reçoit 2 armées.

L'ensemble des armées reçues pourront être placé sur n'importe lesquels des territoires contrôlés par le joueur.

TABLE 2 – Résumé des renforts

Renforts	Nombre
T Territoires contrôlés	$floor(\frac{T}{3})$
Régions contrôlées contenant N territoires	$floor(\frac{N}{2})$
Territoires capturés au tour précédent	1 (50% chance par territoire)

Remarque : Les joueurs ayant un nombre de renfort suffisamment élevé peuvent choisir de recruter des cavaliers (pour 3 unités) ou des canons (pour 7 unités) selon les coûts précisés dans le tableau 4. Attention, une fois le type d'unité choisi, on ne peut plus en changer.

3.2.2 Déplacements et attaques

Après avoir choisi et placé tous ses renforts sur ses territoires, le joueur entre dans la phase de déplacement.

Chaque unité dispose d'un certain nombre de points de déplacements par tours (Cf. tableau 4). Ces points peuvent être utilisés soit pour déplacer des troupes entre 2 *territoires adjacents* ou ayant une connexion maritime. Si le déplacement se fait vers un territoire allié, il ne se passe rien de particulier. Si le déplacement se fait vers un territoire occupé par un autre joueur, il s'agit d'une attaque dont les modalités sont précisés dans la section suivante.

Dans le cas particulier de l'attaque, on ne compte le déplacement qu'en cas de victoire et si les troupes sont effectivement déplacés vers le territoire nouvellement conquis. Autrement dit, en cas de nul une même unité peut tenter plusieurs fois d'attaquer des territoires adjacents pendant un même tour.

Une fois que le joueur a effectué autant de déplacements et d'attaques qu'il le souhaitait, il peut choisir de terminer son tour.

3.2.3 Batailles

Chaque joueur peut effectuer autant d'attaques qu'il veut pendant son tour. Pour chaque bataille, un joueur peut envoyer jusqu'à 3 unités attaquer un territoire voisin de celui sur lequel elles se trouvent.

Le déroulement des combats est le suivant :

- L'attaquant choisit depuis quel territoire attaquer. On ne peut attaquer qu'un territoire voisin et toutes les unités attaquantes doivent venir du même territoire.
- L'attaquant attaque avec au maximum 3 unités à la fois. *Il doit toujours rester au moins 1 unité sur le territoire de départ qui ne participe pas au combat.*
- Le défenseur défend avec au maximum 2 unités qui sont sélectionnés selon leur niveau de priorité défensive : les soldats défendent en 1er, puis les canons, puis les cavaliers.
- Un nombre aléatoire est généré pour chaque unité selon les intervalles indiqués dans la colonne puissance du Tableau 4.
- Les scores les plus élevés sont comparés pour chaque camp : plus élevé attaquant VS plus élevé défenseur, et 2ème plus élevé vs 2ème plus élevé s'il y a 2 unités en défense. **Si plusieurs unités ont le même score dans un camps** et seulement dans ce cas : c'est celle de priorité la plus basse (Cf. Tableau 4) qui est prioritaire pour la comparaison la plus haute. *On notera que les priorité en défense et en attaque ne sont pas les mêmes* !
- Pour chaque comparaison, l'unité ayant le score le plus élevé détruit celle avec le score le moins élevé. L'égalité bénéficie au défenseur.
- Si un camps avait plus d'unités que l'autre, les unités excédentaires de score faible ne sont pas détruites.

— Le combat n'est gagné que lorsque le défenseur n'a plus aucune unité
— Tant qu'il lui reste des unités, l'attaquant peut réattaquer avec une composition de son choix.
— En cas de capture d'un territoire, l'attaquant y place toutes les unités ayant participé et survécu à l'attaque.
— Il aura 50% de chance d'avoir 1 renfort supplémentaire lors de son prochain tour par territoire capturé.

TABLE 3 – Exemples de combats

Attaquant	Défenseur	Attaquant	Défenseur
Soldat : 6 Soldat : 5 Soldat : 1	Soldat : 4 Soldat : 3 -	Soldat : ok Soldat : ok Soldat : ok	Soldat : détruit Soldat : détruit -
Cavalier : 4 Soldat : 3	Soldat : 6 -	Cavalier : détruit Soldat : ok	Soldat : ok -
Soldat : 3	Soldat : 3	Soldat : détruit	Soldat : ok
Cavalier : 7 -	Canon : 4 Cavalier : 4	Cavalier : ok -	Canon : détruit Cavalier : ok
Soldat : 6 Soldat : 3	Soldat : 5 Soldat 4	Soldat : ok Soldat : détruit	Soldat : détruit Soldat : ok
Canon : 5 Cavalier : 3 Soldat : 3	Soldat : 4 Soldat : 3 -	Canon : ok Cavalier : détruit Soldat : ok	Soldat : détruit Soldat : ok -
Cavalier : 6 Soldat : 4 Canon : 4	Soldat : 5 Soldat : 4 -	Cavalier : ok Soldat : détruit Canon : ok	Soldat : détruit Soldat : ok -

Tant qu'il reste des unités à l'attaquant sur des territoires adjacents à des territoires ennemis, il peut effectuer autant de combats qu'il le souhaite pendant son tour.

3.3 Tableau des caractéristiques de unités

TABLE 4 – Puissance et coût des unités

Unité	Coût	Puissance	Priorité ATT	Priorité DEF	mvt/tour
Soldat	1	1-6	2	1	2
Cavalier	3	2-7	1	3	3
Canon	7	4-9	3	2	1

Le tableau ci-dessus résume les caractéristiques des unités existantes. Ces chiffres étant susceptibles d'évoluer pour équilibrer le jeu, il est fortement recommandé de prévoir un code adaptable permettant de facilement modifier ces valeurs et éventuellement de pouvoir rajouter des unités.

Le coût indique combien de renforts de bases doivent être échangés pour pouvoir obtenir une unité plus avancée.

La puissance correspond à l'intervalle de valeurs aléatoires qui peuvent être générées par l'unité lors d'un combat, que ce soit en attaque ou en défense.

La colonne "Priorité ATT" indique pour les attaquants quelles unités seront comparés aux unités adversaires les plus fortes en cas d'égalité de puissance générée aléatoirement, et seulement dans ce cas. Cette caractéristique n'indique en aucun cas quelles unités doivent être envoyées au combat en 1er, l'attaquant peut choisir d'envoyer les unités qu'il souhaite pour attaquer un territoire ennemi.

La colonne "Priorité DEF" sert dans 2 cas : 1) Elle détermine quelles unités défendent en 1er en cas d'attaque (le défenseur ne choisit donc pas avec quoi il défend). 2) Comme pour "Priorité ATT", cette caractéristique détermine quelles unités seront comparés aux unités adversaires les plus fortes en cas d'égalité de puissance générée aléatoirement.

La colonne "mvt/tour" indique combien de déplacement chaque unité peut faire dans un même tour. On rappelle qu'en cas de bataille, le déplacement ne compte que si le combat est gagné et l'unité effectivement déplacée sur le territoire conquis.

4 Travail attendu

Par équipes de 3 personnes maximum, vous devrez coder ce jeu et son interface graphique en Java.

4.1 Fonctionnalités attendues

- Interface graphique rendant le jeu jouable pour 2 à 6 joueurs humains chacun leur tour
- Implémentation de la carte de base (voir Section 2).
- Implémentation des fonctionnalités de base : initialisation, tours de jeu.
- Implémentation de la victoire par destruction de tous les ennemis.
- Implémentation d'une IA capable de jouer pour détruire tout le monde.
- Rédaction d'un document technique incluant les diagrammes UML de votre programme et un manuel d'utilisation.

4.2 Fonctionnalités facultatives

- Implémentation des missions (Section 2.2.1)
- Implémentation d'une IA capable de jouer les missions
- Cartes de jeux personnalisées (Westeros, Azeroth ?)
- Unités personnalisées
- Bonus (optionnel) : Concours d'IA avec combat de vos IAs de Risk sur le grand écran en amphithéâtre

5 Livrables

- Pour le 5 Juin 2018 : Rendu du document technique sur moodle.
- Pour le 8 Juin 2018 : Présentation d'une démo de votre programme lors d'une soutenance de 15min (10 min présentation, 5 min questions).
- Pour le 8 Juin 2018 (toujours) : Dépot sur moodle de l'archive avec votre code.

6 Concours d'IA

Le but sera de faire combattre votre IA contre celle de vos camarades et diffuser en direct les matchs sur le grand écran de l'amphitéâtre L012. Dans une ambiance fun et sympatique.

Le concours d'IA aura lieu 1 ou 2 semaines après la semaine de soutenance.

Pour y participer, nous vous donnerons une classe Java nommé IAConnector. Cette classe se connectera au serveur central, enverra vos mouvements et vous donnera les informations nécessaires sur le jeu.

L'utilisation sera très simple afin de pouvoir vous permettre de vous connecter facilement au serveur. Exemple d'utilisation du connecteur :

```
public class Main {
    public static void main(String[] args) {
        IAConnector connector = new IAConnector();
        connector.connectToGame("Game 2");

        // Voici comment envoyer des mouvements:
        connector.sendMove("PlaceUnit France 5 2 1"); // Positionne des
            unites sur le territoire nomme "France". Les unites sont 5
            soldats, 2 cavaliers et 1 cannon.
        connector.sendMove("Attack France Italie 0 1 2"); // Lance une
            attaque depuis la France sur l'Italie. Les unites participant
            a l'attaque sont 0 soldats, 1 cavalier et 2 cannons.
        connector.sendMove("MoveUnit France Italie 7 4 2 "); // Deplace
            des unites de la France jusqu'a l'Italie. Les unites deplaces
            sont 7 soldats, 4 cavalier, et 2 cannons.
        connector.sendMove("EndTurn"); // Pour terminer son tour.

        // Voici la boucle de jeu que vous allez avoir:
        while (true) {
            connector.waitOpponentsMove() // Le connecteur se met en
                attente des mouvements des autres joueurs et rend la main
                quand c'est a vous de jouer.
            Game game = connector.getGame(); // La methode getGame vous
                permettra de racuparer l'etat du jeu pour savoir ce qui
                s'est passe.
            String myMove = MyGame.getMyMove(game); // C'est a vous de
                coder cette partie la.
            connector.sendMove(myMove);
        }

    }
}
```

Pour récupérer la classe IAConnector, vous devrez nous montrer votre projet en version complète (sans la partie avancé de l'IA), avant la soutenance il sera interdit de partager cette classe IAConnector.

Lot : Nous recherchons des lots, il y aura au moins un petit drone à gagner.

Plus d'infos : N'hésitez pas à contacter Jérémie Sublime (jeremie.sublime@isep.fr) et Thibaut de Broca (tdebroc@gmail.com).