Université d'Artois
Faculté des Sciences Jean Perrin

Conception Orientée Objet

Dots And Boxes - Compte Rendu

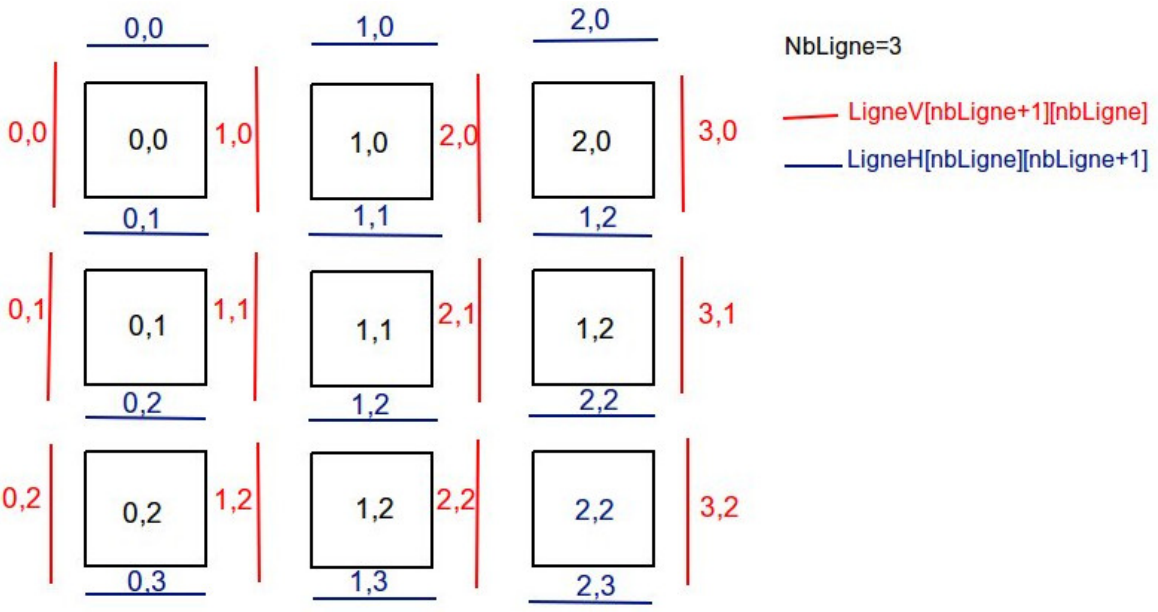
1 Ce qui a été réalisé

-Implémentation du jeu en MVC (Quelques confusions sur le menu)
-Implémentation d'une IA
-Menu de gestion du nombre de joueurs, du type de joueurs (Humain/IA), du nombre de cases.

2 Ce qui a été réalisé

Pour répondre au problème, on va modéliser la grille de la manière suivante :
-Un ensemble de traits verticaux
-Un ensemble de traits horizontaux
-Un ensemble de cases

Chaque attribut dispose d'une valeur associé à un joueur.
A l'initialisation, aucun attribut n'est assigné aux joueurs.



3 Méthode de décision de l'IA

3.1 Modélisation

Première chose à dire, chaque trait a une influence sur au plus 2 carrés. (1 seul pour les bordures).

Pour chaque trait, on va regarder les carrés auxquels il appartient.
Après avoir récupéré le(s) carré(s), on va compter les cotés appartenant à un joueur, et on va établir un couple de 2 entiers. (-1 si pas de carrés possibles)

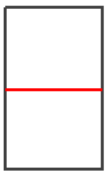
Exemple :
TraitH (3,4) si on ajoute le trait, on aura 3 traits/4 sur le carré du haut, et un carré entier sur le carré du dessous.
TraitV (-1,2) Trait sur la bordure gauche, on aura 2 traits/4 sur le carré de droite.

3.2 Décision

L'IA va alors choisir de manière aléatoire et dans l'ordre de priorité suivant le couple le plus adapté :

+++
^
|(4,4) : Optimal, On peut former deux carrés
|(4,1)/(1,4) : Prioritaire (forme un carré et aucun avantage pour l'autre joueur)
|(4,2)/(2,4) : Prioritaire (forme un carré et aucun avantage pour l'autre joueur)
|(4,3)/(3,4) : (Forme un carré mais avantage pour l'autre joueur)
|(1,1) : Aucun risque
|...
|...
|(3,1)/(1,3) : Laisse un carré pour l'autre joueur
|(3,2)/(2,3) : Laisse un carré pour l'autre joueur
|(3,3)/(3,3) : Pire cas, 2 carré pour l'autre joueur
- - -

Trait Vertical {i,j} :
Carré1 : TraitV[i][j] ; TraitV[i-1][j] ; TraitH[i-1][j] ; TraitH[i-1][j+1]
Carré2 : TraitV[i][j] ; TraitV[i+1][j] ; TraitH[i][j] ; TraitH[i][j+1]



Trait Horizontal {i,j} :
Carré1 : TraitH[i][j] ; TraitH[i][j-1] ; TraitV[i][j-1] ; Trait[i+1][j-1]
Carré2 : TraitH[i][j] ; TraitH[i][j+1] ; TraitV[i][j] ; TraitV[i+1][j]