Dossier : jeu de dominos

Nicolas Poulain

20 mars 2012

Table des matières

1 Règles du jeu de dominos

Le jeu de dominos est un jeu de société d'origine chinoise, utilisant 28 pièces (dans le cas d'un jeu "double-six"), les dominos. On peut adopter une des règles suivantes :

Règle

- Distribuer au hasard 10 des 28 dominos à chaque joueur.
- Celui qui a le domino le plus fort (ordre lexicographique) commence et pose celui-ci sur la table
- Chaque joueur pose tour à tour à l'une des extrémites du jeu sur la table un domino de sorte que les parties voisines ont le même nombre de points, constituant ainsi une chaîne.
- Le joueur qui ne peut pas jouer passe son tour, et on continue à jouer jusqu'à ce qu'un des joueurs se soit débarrassé de tous ses dominos, ou que le jeu soit complètement bloqué.
- À la fin du jeu, celui qui totalise le moins de points (la somme des points de l'ensemble des dominos) est le gagnant. On a donc tout intérêt à se débarrasser en premier des dominos valant beaucoup de points.

Variante

- Lorsqu'un joueur n'a pas de domino qui convienne, il pioche en prenant une pièce du talon et passe son tour, c'est le suivant qui joue.
- Le vainqueur est celui qui a placé le premier tous ses dominos.

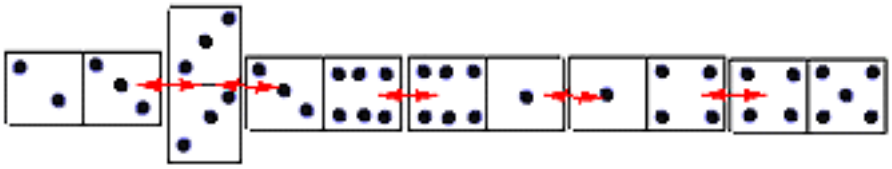

Figure 1 –

1.1 Le projet

Le but de ce projet est de comparer différentes stratégies de jeu en opposant deux joueurs virtuels et en leur faisant jouer un grand nombre de parties. Lesdonnées statistiques recueillies sur le nombre de victoires nous conduiront déterminer la meilleure méthode pour gagner à ce jeu.

2 Modélisation

Un domino peut être représenté par un vecteur colonne $\begin{pmatrix} x \\ y \end{pmatrix}$ ou par un nombre $10x + y$. Ainsi le jeu d'un joueur peut être représenté par la matrice (le tableau)

$J = \begin{pmatrix} 6 & 5 & 5 & 3 & 3 & 3 & 1 \\ 1 & 4 & 0 & 3 & 1 & 0 & 1 \end{pmatrix}$ ou $J = (61, 54, 50, 33, 31, 30, 11)$.

2.1 Premières fonctions

Ce sont des fonctions qui ne servent pas directement au déroulement de la partie de dominos mais qui permettent sa mise en place ou qui sont indispensables en tant qu'outils pour les fonctions avancées.

1. Écrire la fonction `creation_jeu` capable de donner l'ensemble des dominos de la boîte de jeu.

$$\begin{pmatrix} 6 & 6 & 6 & 6 & 6 & 6 & 6 & 5 & 5 & 5 & 5 & 5 & 5 & 4 & 4 & 4 & 4 & 4 & 3 & 3 & 3 & 3 & 2 & 2 & 2 & 1 & 1 & 0 \\ 6 & 5 & 4 & 3 & 2 & 1 & 0 & 5 & 4 & 3 & 2 & 1 & 0 & 4 & 3 & 2 & 1 & 0 & 3 & 2 & 1 & 0 & 2 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

2. Écrire une fonction `distribue` qui tire au hasard (et sans remise) une main contenant un nombre donné de dominos.
3. Écrire une fonction `est_avant` qui admet comme paramètres d'entrée deux dominos et qui renvoie `True` ou `False` selon que les deux dominos sont ou pas classés dans l'ordre décroissant. Par exemple `est_avant([6,2],[5,0])` renverrait `True`.
4. Écrire une fonction `tri_decr` qui trie un ensemble de dominos par ordre décroissant.

2.2 Les fonctions concernant le déroulement de la partie

Ces fonctions permettent d'initialiser la partie, d'enchaîner les tours de jeux, et de terminer la partie. Elles utilisent les fonctions citées précédemment à travers d'autres fonctions plus complexes. Les deux principales fonctions sont celle permettant de désigner le joueur qui posera le premier domino, et celle qui, grâce à une boucle, fait jouer les joueurs chacun leur tour.

1. Écrire une fonction `possibilites` qui à partir de la chaîne de dominos se trouvant déjà sur la table et de la main d'un joueur, renvoie les choix possibles de dominos pour ce joueur. Par exemple , pour une table constituée de la chaîne de dominos [3,4],[4,4], et pour une main constituée des dominos {[2,3],[1,5],[4,6]}, la fonction reverrait les nombres 0 et 2 qui sont les positions des deux dominos pouvant être placés sur la table.
2. Écrire une fonction `is_player1_first` qui indique si le joueur numéro 1 est ou pas celui qui commence la partie selon qu'il possède le domino le plus fort.

2.3 Les fonctions concernant les stratégies

La stratégie des deux joueurs est déterminée dans la fonction effectuant les tours de jeu. Elle peut être différente pour chacun des deux joueurs, d'où son intérêt. Il y a trois catégories de stratégie :

– une où le joueur choisit ses dominos à placer totalement au hasard ;
– celles créées pour faire gagner le joueur (jouant les dominos placés en premier dans l'ordre lexicographique ; soit ceux qui valent le plus de points ; soit en jouant ceux dont les deux valeurs sont les plus présentes dans la main du joueur, ainsi il garde le plus de possibilités pour les tours à venir) ;
– une dernière stratégie consistant à tenter de le faire perdre en jouant les dominos valant le moins de points.

2.4 Programme Python

```
#!/usr/bin/python
import random
```

```
def creation_jeu(max=6):
  """Cree la boite de jeu avec l'ensemble des dominos
  qui seront distribues
  """
  jeu=[]
  for i in range(max,-1,-1):
    for j in range(i,-1,-1):
      jeu = jeu + [[i,j]]
  return jeu

def distribue(jeu,n=10):
  """Tire dans jeu (sans remise) une main de n dominos"""
  main = []
  for i in range(n):
    r = random.randint(0,len(jeu)-1)
    main = main + [ jeu[r] ]
    jeu.pop(r)
  return main

def est_avant(d,e):
  """Verifie si deux dominos sont dans l'ordre lexicographique
  - Exemples :
  >>> est_avant( [2,3], [1,2] ); est_avant( [2,3], [2,4] )
  True
  False
  """
  if d[0]>e[0] or ( d[0]==e[0] and d[1]>e[1] ):
    return True
  return False

def tri_decr(player):
  """Trie les dominos du joueur dans l'odre
  decroissant lexocographique
  - Exemple :
  >>> tri_decr([ [1,2], [3,4], [1,3], [2,0] ])
  [[3, 4], [2, 0], [1, 3], [1, 2]]
  """
  player = sorted(player)
  player.reverse()
  return player

def is_player1_first(pl1,pl2):
  """Le joueur 1 a-t-il une meilleure main que le joueur 2 ?"""
  if est_avant(pl1[0],pl2[0]):
   return True
  return False

def possibilites(table,pl):
  """Donne, la liste des dominos de pl qui peuvent
  etre places sur la table
  >>> possibilites([[3,4],[4,4]], [[2,3],[1,5],[4,6]])
  [0, 2]
  """
  possbl = []
  x=table[0][0]
  y=table[-1][1]
  for i in range(len(pl)):
    if pl[i][0]==x or pl[i][1]==x or pl[i][0]==y or pl[i][1]==y:
      possbl = possbl + [ i ]
  return possbl

def un_tour_de_jeu(table,player,passe):
  """Tente de placer sur un des deux bouts de la table un
  domino de la main du player. Si ce n'est pas possible,
  on incremente passed_tours.
  """
  possbl = possibilites(table,player)
  if len(possbl)==0:
    return table,player,passe + 1
  table = positionne(player[ possbl[0] ], table)
  player.pop(possbl[0])
  return table,player,0

def positionne(domino,table):
  """Positionne le domino correctement sur la table"""
  if domino[0]==table[0][0]:
```

```
    table = [ [ domino[1],domino[0] ] ] + table
  elif domino[1]==table[0][0]:
    table = [ [ domino[0],domino[1] ] ] + table
  elif domino[0]==table[-1][1]:
    table = table + [ [ domino[0],domino[1] ] ]
  else:
    table = table + [ [ domino[1],domino[0] ] ]
  return table

if __name__ == "__main__":
  import doctest
  doctest.testmod()

  jeu = creation_jeu()
  player1 = tri_decr(distribue(jeu))
  player2 = tri_decr(distribue(jeu))
  print "La main du Joueur 1 :",player1
  print "La main du Joueur 2 :",player2
  print ""
  # initialisation de la partie
  if is_player1_first(player1,player2):
    table = [ player1[0] ]
    print "Joueur 1 joue :", table
    player1.pop(0)
    a_qui_le_tour = 2
  else:
    table = [ player2[0] ]
    print "Joueur 2 joue :", table
    player2.pop(0)
    a_qui_le_tour = 1

  # c'est parti
  passed_tours = 0
  while passed_tours<2 and len(player1)>0 and len(player2)>0:
    if a_qui_le_tour==1:
      table,player1,passed_tours = un_tour_de_jeu(table,player1,passed_tours)
      if passed_tours > 0:
        print "Joueur 1 passe."
      else:
        print "Joueur 1 joue :", table
      a_qui_le_tour = 2
    else:
      table,player2,passed_tours = un_tour_de_jeu(table,player2,passed_tours)
      if passed_tours > 0:
        print "Joueur 2 passe."
      else:
        print "Joueur 2 joue :", table
      a_qui_le_tour = 1

  # Fin de partie
  print "La main du Joueur 1 :",player1
  print "La main du Joueur 2 :",player2
```

voir http ://code.google.com/p/npoulain pour le code