Rapport du projet de théorie des jeux

Maxence Ahlouche Maxime Arthaud Korantin Auguste
Martin Carton Thomas Forgione Thomas Wagner

11 novembre 2013

Table des matières

Listings

1 Présentation de l'équipe

Cette équipe a été menée par Thomas Forgione, assisté de son Responsable Qualité Maxence Ahlouche. Les autres membres de l'équipe sont Martin Carton, Thomas Wagner, Maxime Arthaud, et Korantin Auguste.

Tous les membres de l'équipe ont été présents à chacune des séances lors de cette UA.

2 Shifumi

3 Morpion

4 Compétition/Duopole

Le but de ce jeu est de maximiser le gain d'une entreprise en concurrence avec une autre entreprise en fonction de leur production.

4.1 Analyse

Notre gain étant égal à $g_x(x) = -x(x + y - 3)$ avec x et y les productions respectives des deux entreprises, pour le maximiser il suffirait de jouer $x = \frac{3-y}{2}$ (c'est que fait la stratégie fournie `noncooperatif`).

Cependant au moment de décider quelle quantité produire nous ne connaissons pas la production y de l'entreprise concurrente. De plus si les deux entreprises s'ignorent totalement en essayant de maximiser leur seul gain, elles gagneront au final moins que deux entreprises qui coopèrent totalement c'est à dire qui chercheraient à maximiser leur gain total $g_x(x) + g_y(y)$ avec $g_x(x) = g_y(y)$ c'est à dire en jouant $x = y = 0.75$. En effet, elles gagneront alors chacune $\frac{9}{8}$ à chaque tour au lieu de 1 si elles sont toutes les deux non-coopératives.

Nous avons donc essayé plusieurs stratégies différentes, qui essayent de maximiser le gain de l'entreprise en tenant en compte l'autre entreprise, les productions et les gains des tours précédents.

4.2 Stratégies

4.2.1 Stackelberg

Todo : Pourquoi 2/3 ? Pourquoi 1.1*2/3 c'est mieux.

Une variante de cette stratégie consiste à utiliser la production moyenne de l'autre joueur plutôt que seulement la dernière valeur. Elle permet d'obtenir des résultats légèrement meilleurs.

De plus en coopérant avec l'autre joueur (voir listing 1) si celui-ci coopère, on obtient de meilleurs résultats.

Enfin, une autre variante de cette stratégie (voir listing 2) maximise le gain si l'autre joueur a joué une constante sur les derniers tours. Cette variante donne des résultats moyens un peu moins bons, mais est la meilleure dans le pire des cas : elle est donc adaptée à une entreprise qui veut minimiser ses risques.

4.2.2 Stratégie pénalisante

Le principe de cette stratégie (voir listing 3) est d'être coopératif tant que l'adversaire l'est, et de devenir plus agressif quand il ne l'est plus : à chaque fois que l'autre joueur n'est pas coopératif, on joue comme le ferait la stratégie Stackelberg.

Une variante de cette stratégie (voir listing 4) consiste à le pénaliser de plus en plus : la première fois on le pénalise une fois, puis deux, puis trois, etc.

Ces deux stratégies sont efficaces à la fois quand l'autre joueur est coopératif (on est alors coopératif) et contre un joueur non-coopératif (on devient alors agressif).

4.2.3 Stratégie évolutive

Une autre stratégie (voir listing 5) que nous avons développée consiste à augmenter la production si la dernière augmentation a augmenté notre gain ou si la dernière diminution l'a diminué et vice-versa.

Celle-ci est plutôt efficace, mais n'est pas la meilleure que nous ayons développée : elle se met souvent à osciller inutilement.

4.2.4 Stratégie polynomiale

Enfin, une autre stratégie (voir listing 6) joue en fonction de la production moyenne de l'autre joueur telle que :

— $f(0) = 1.125$: on joue beaucoup si l'autre joue peu, sans jouer trop pour ne pas le fâcher ;
— $f(0.75) = 0.75$: elle coopère avec quelqu'un qui coopère ;
— $f(1.5) = 0.75$: elle coopère avec quelqu'un qui ne coopère pas, pour essayer de faire coopérer celui-ci (c'est dans notre intérêt et ça ne changerait pas son gain, il est donc possible qu'elle le fasse).

On choisit alors la fonction f pour être un polynôme qui interpole ces valeurs.

Cette méthode s'avère très efficace en moyenne.

4.3 Comparaison

La table 1 montre les résultats obtenus par les quelques stratégies que nous avions à notre disposition [1] pour une durée de 100 tours. Nous faisons s'opposer toutes les stratégies entre elles puis notons pour chacune d'elles son gain minimal, moyen et maximal [2].

Todo : mettre à jour avec les dernières valeurs quand on aura fini et mettre les résultats pour un autre nombre de tours.

1. Les stratégies fournies par les professeurs, les notres et des stratégies « prêtées » par d'autres groupes pour les tests.
2. Cette table peut être générée par le script matlab *comp_tests.m* fourni dans l'archive.

Stratégie	Gain minimal	Gain moyen	Gain maximum
Coopératif	110.75	114.02	125.156
Non coopératif	83.25	96.16	109.13
Stackelberg	54.42	64.67	72.75
Pénalise	0	44.98687	109.01

TABLE 1 – Résultats des différentes stratégies sur 100 tours

5 Annexes

Listing 1 – Statégie Stackelberg sur la moyenne

```
% Stackelberg en moyenne, sauf si l'adversaire est coopératif.
function x = strategie(numpart,tx,ty,gx,gy)
if (numpart == 2)
        x= 0.75;
else
    ty_near_mean = mean(ty(max(numpart-5, 2):numpart-1));

    if (ty_near_mean < 0.76)
        x = 0.75;
    else
        ty_mean = mean(ty(2:numpart-1));
        x = 2*(3-ty_mean)/3;
    end;
end;
```

Listing 2 – Statégie Stackelberg sur la moyenne (variante)

```
% Stackelberg en moyenne, sauf si l'adversaire est coopératif avec
% maximisation des gains si l'adversaire est constant.
function x = strategie(numpart,tx,ty,gx,gy)
if (numpart == 2)
        x= 0.75;
else
    near = ty(max(2:numpart-15):max(2,numpart-1));
    cst = false;
    if (near == mean(near))
      cst = true;
    end;

    if (cst && numpart > 5)
        x = (3-mean(near))/2;
    else
        ty_near_mean = mean(ty(max(numpart-5, 2):numpart-1));

        if (ty_near_mean < 0.76)
            x = 0.75;
        else
            ty_mean = mean(ty(2:numpart-1));
            x = 2*(3-ty_mean)/3;
        end;
    end;
end;
```

Listing 3 – Statégie pénalisante

```
% Stratégie qui "pénalise" l'adversaire s'il n'est pas coopératif.
function x = strategie(numpart,tx,ty,gx,gy)
nbr_penal_y = 0;
nbr_penal_x = 0;

for i = 1:numpart-1
  if (ty(i) > 0.75)
    nbr_penal_y = nbr_penal_y + 1;
  end;
  if (tx(i) > 0.75)
    nbr_penal_x = nbr_penal_x + 1;
  end;
end;

if (nbr_penal_x <= nbr_penal_y)
  ty_mean = ty(2);
  if numpart > 2
    ty_mean = mean(ty(2:numpart-1));
  end;

  x = (3-ty_mean)/2;
else
  x = 0.75;
end;
```

Listing 4 – Statégie pénalisante (variante)

```
% Stratégie qui "pénalise" de plus en plus l'adversaire s'il n'est pas
% coopératif.
function x = strategie(numpart,tx,ty,gx,gy)
nbr_penal_y = 0;
nbr_penal_x = 0;

for i = 2:numpart-1
  if (ty(i) > 0.75)
    nbr_penal_y = nbr_penal_y + 1;
  end;
end;

while numpart-nbr_penal_x-1>0 && (tx(numpart-nbr_penal_x-1) > 0.75),
  nbr_penal_x = nbr_penal_x + 1;
end;

if (nbr_penal_x < nbr_penal_y)
  x = 1.1255*2*(3-ty(numpart-1))/3;
else
  x = 0.75;
end;
```

Listing 5 – Stratégie évolutive

```
function x = strategie(numpart,tx,ty,gx,gy)
persistent step;
if (numpart <= 2)
        x = 0.75;
        step = 1.3;
elseif (numpart == 3)
        x = 0.85;
else
```

```
        if ((gx(numpart-1)>gx(numpart-2)) && (tx(numpart-1)>tx(numpart-2)))
                x = tx(numpart-1) + step;
                step = step * 1.2;
        elseif (gx(numpart-1)<gx(numpart-2) && (tx(numpart-1)>tx(numpart-2)))
                x = tx(numpart-1) - step;
                step = step / 1.2;
        elseif  (gx(numpart-1)>gx(numpart-2) && (tx(numpart-1)<tx(numpart-2)))
                x = tx(numpart-1) - step;
                step = step * 1.2;
        else
                x = tx(numpart-1) + step;
                step = step / 1.2;
        end

        x = max(0.75, min(1.5, x));
end
```

Listing 6 – Stratégie polynomiale

```
% Stratégie qui joue telle que:
%   f(0)    = 1.125   (on est pas trop brutal)
%   f(0.75) = 0.75    (on coopère avec qqn qui coopère)
%   f(1.5)  = 0.75    (on essaye de coopérer avec qqn qui ne coopère pas, des
%                      fois qu'il changerait d'avis)
% en fonction de mean(ty(2:numpart-1)).
function x = strategie(numpart,tx,ty,gx,gy)
if (numpart <= 2)
    x= 0.75;
else
    x= 1/3*mean(ty(2:numpart-1))^2-3/4*mean(ty(2:numpart-1))+1.125;
end;
```