Rapport du projet 1 MOSIMA : Modèle multi-agents de rationalité dans les organisations : émergence d'une culture d'entreprise

Université Pierre et Marie Curie - MOSIMA
24 octobre - 27 novembre 2016

Antunes Costa Gonçalves Daniel
Wolfrom Matthieu

1 Etude du modèle

1.1 Question 1

Les auteurs de l'article emploie le terme *corporate culture* pour définir un équilibre atteint dans une structure complexe, ici un équilibre où les agents exercent le même effort.
Ils ont décidé d'aborder la question par une simulation multi-agents, en modélisant l'effort des différents agents et en cherchant à étudier les cas où leurs interactions les mènent vers un niveau commun d'effort, un tel équilibre étant une *corporate culture*.

1.2 Question 2

Le modèle proposé correspond à une organisation, composée simplement d'un ensemble d'agents. Un agent doit interagir avec un autre agent pour réaliser leurs tâches, ainsi lorsque deux agents se rencontrent, ils forment une équipe et complètent leurs tâches. Suite à une telle coopération, chaque agent de l'équipe observe l'effort fourni par son coéquipier pour adapter son comportement futur. Les deux agents calculent également à ce point le profit qu'ils ont dégagés de cette interaction.
Pour représenter les agents ainsi que les interactions, l'environnement est simulé par une grille où chaque agent occupe une case, et peut interagir avec un autre agent s'il lui fait face.
Les agents de ce modèle sont ainsi définis par leur type, leur position et orientation sur la grille, leur effort actuel et précédent, profit actuel et cumulé, l'effort et le profit de leur dernier coéquipier, l'effort moyen et le profit moyen du voisinage.
Les actions que les agents peuvent réaliser sont tourner dans une direction aléatoire, se déplacer d'une case vide, et travailler quand la situation l'autorise.
Les types d'agents sont les suivants :

0. *null effort* : L'agent fournit toujours un effort quasiment nul
1. *shrinking effort* : L'agent réalise un effort égal à la moitié de l'effort de son précédent coéquipier
2. *replicator* : L'agent réalise un effort égal à l'effort réalisé par son précédent coéquipier
3. *rational* : L'agent offre la meilleure réponse à l'effort de son précédent coéquipier
4. *profit comparator* : L'agent compare son profit à celui de son précédent coéquipier et augmente son effort si son profit était plus important
5. *high effort* : L'agent fournit toujours un effort maximal
6. *average rational* : L'agent offre la meilleure réponse à l'effort moyen de l'ensemble de ses coéquipiers
7. *winner imitator* : L'agent débute avec un effort maximal mais copie l'effort de son coéquipier si cela résulte en un plus grand profit
8. *effort comparator* : L'agent compare son effort avec celui de son précédent coéquipier, et cherche à rapprocher son effort vers celui observé
9. *averager* : L'agent choisit comme effort la moyenne entre son effort et celui de son précédent coéquipier

1.3 Question 3

La fonction de profit est la suivante : $\pi_i = 5\sqrt{e_i + e_j} - e_i^2$ où e_i représente l'effort réalisé par l'agent i.
Les agents sont distribués de manière aléatoire sur la grille. Ensuite la simulation réalise un cycle de 2 étapes : mouvement, interaction (l'étape d'interaction peut être vu comme une composition de 2 étapes : jeu et adaptation). Le mouvement est réalisé de manière aléatoire en respectant la règle de n'avoir qu'un seul agent par case de la grille. Le jeu se déclenche ensuite si deux agents se font face. Après cela, les agents adaptent leurs comportements en fonction de leur type. Leur effort initial est dans la plupart des cas aléatoire. Les agents *null effort* et *high effort* exerce directement leurs efforts respectifs. L'agent *winner imitator* commence systématiquement avec un effort maximum.

Algorithm 1 Movement

Ensure: déplace les agents aléatoirement
  for all agent a **do**
    $a.direction = random(0, 90, 180, 270)$
    if $a.place\_ahead$ is free **then**
      $a.move\_To(a.place\_ahaed)$
    end if
  end for

Algorithm 2 Game

Ensure: algorithme permettant aux agents de jouer le jeu
  for all agent a **do**
    if $a.is\_agent\_ahead\_facing()$ **then**
      $b = a.place\_ahead.get\_Agent()$
      $a.profit\_function$(a.effort,b.effort)
    end if
  end for

Le modèle peut utiliser un bruit sur les observations de l'effort de son coéquipier, l'agent observant aura alors une valeur plus haute ou plus basse que la réalité. La bruit va de 1%, résultant en une multiplication par un facteur allant de 0.99 à 1.01, jusqu'à 50% avec un facteur entre 0.5 et 1.5.

Algorithm 3 Adaptation

Ensure: adapte l'effort des agents pour les interactions futures

for all agent a **do**

if $a.type ==$ shrinking effort **then**

$a.effort = a.lastPartnerEffort/2$

end if

if $a.type ==$ replicator **then**

$a.effort = a.lastPartnerEffort$

end if

if $a.type ==$ rational **then**

$a.effort = argmax_e(5\sqrt{e + lastPartnerEffort} - e^2)$

end if

if $a.type ==$ profit comparator **then**

if $a.lastProfit > a.lastPartnerProfit$ **then**

$a.effort+ = 10\%$

end if

end if

if $a.type ==$ average rational **then**

$a.effort = argmax_e(5\sqrt{e + meanPartnerEffort} - e^2)$

end if

if $a.type ==$ winner imitator **then**

if $a.lastProfit < a.lastPartnerProfit$ **then**

$a.effort = lastPartnerEffort$

end if

end if

if $a.type ==$ effort comparator **then**

if $a.lastEffort < a.lastPartnerEffort$ **then**

$a.effort+ = 10\%$

else

$a.effort- = 10\%$

end if

end if

if $a.type ==$ averager **then**

$a.effort = (lastEffort + lastPartnerEffort)/2$

end if

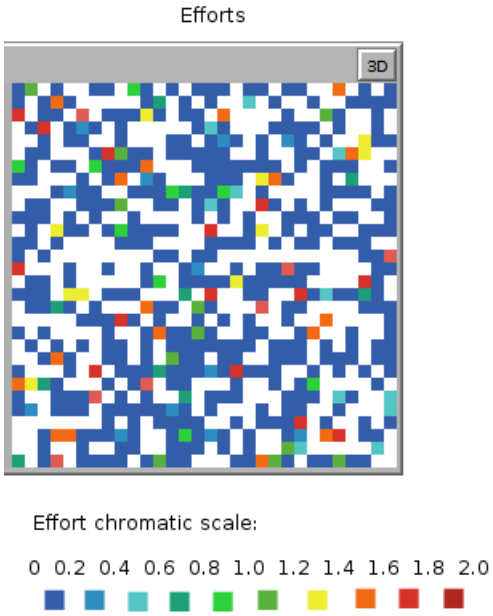
end for

2 Implémentation

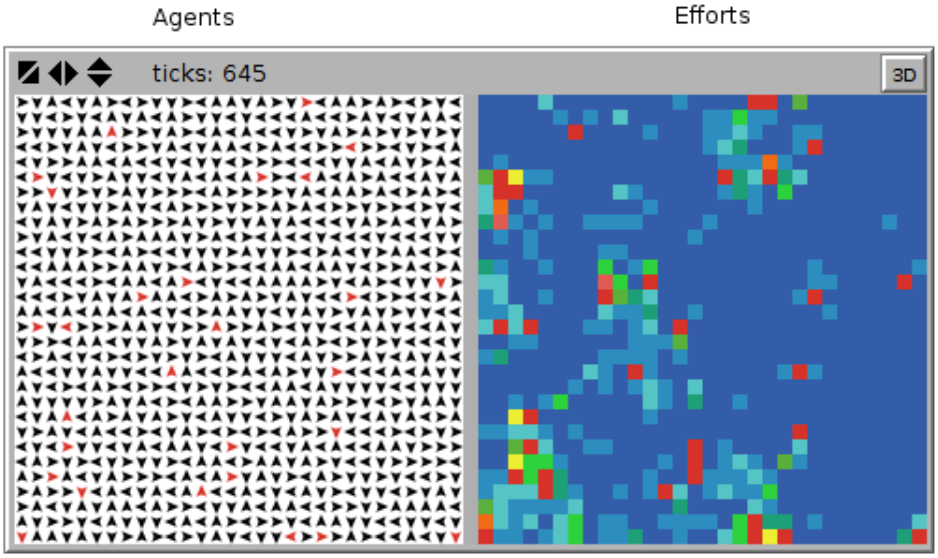
Pour les paramètres de la simulation, nous avons choisi :
— Dimensions X et Y de la grille
— La possibilité d'afficher ou non la fenêtre d'effort
— La possibilité d'activer et de calibrer le bruit
— Sélection du nombre de catégories d'agents actifs
— Pour chaque catégorie d'agent, le nombre et le type d'agents voulus

Nous affichons à l'écran plusieurs graphes permettant de suivre l'évolution au cours du temps de certaines variables comme l'effort moyen des agents, l'écart-type de l'effort et le profit moyen.

Pour reproduire la figure 3 nous affichons les efforts des agents dans une fenêtre grâce à une échelle chromatique, par exemple les efforts $e \in [0; 0.2]$ sont représentés en bleu foncé, et les efforts $e \in]1.8; 2]$ sont représentés en rouge foncé.



Nous affichons une représentation spatiale des agents ainsi que leur direction et leurs efforts dans une fenêtre scindée en deux. On peut observer dans la figure suivante une simulation de la situation de la fig. 5 de l'article où l'on obverse qu'aucun équilibre n'est formé dans une population d'agent de type "effort comparator" (en noir) avec quelques agents "high effort", (en rouge) et que l'effort des agents "effort comparator" est plus élevé dans les zones avec une concentration d'agents "high effort".



3 Expérimentations et reproduction des résultats

3.1 Question 1

— Une population qui ne comprend que des agents rationnels :
L'effort des agents rationnels est initialisé aléatoirement, il adapte ensuite son effort après chaque interaction avec un autre agent en maximisant la fonction de profit étant donné l'effort de son dernier partenaire, soit $argmax_{e_i}\ 5\sqrt{e_i + e_j} - e_i^2$ où e_j est l'effort exercé par le dernier partenaire. Dans une population homogène de ces agents nous obtenons un équilibre des efforts en 0.92101 en moins de 35 ticks. Les efforts des agents convergent ainsi vers une corporate culture qui représente l'équilibre de Nash de la simulation.

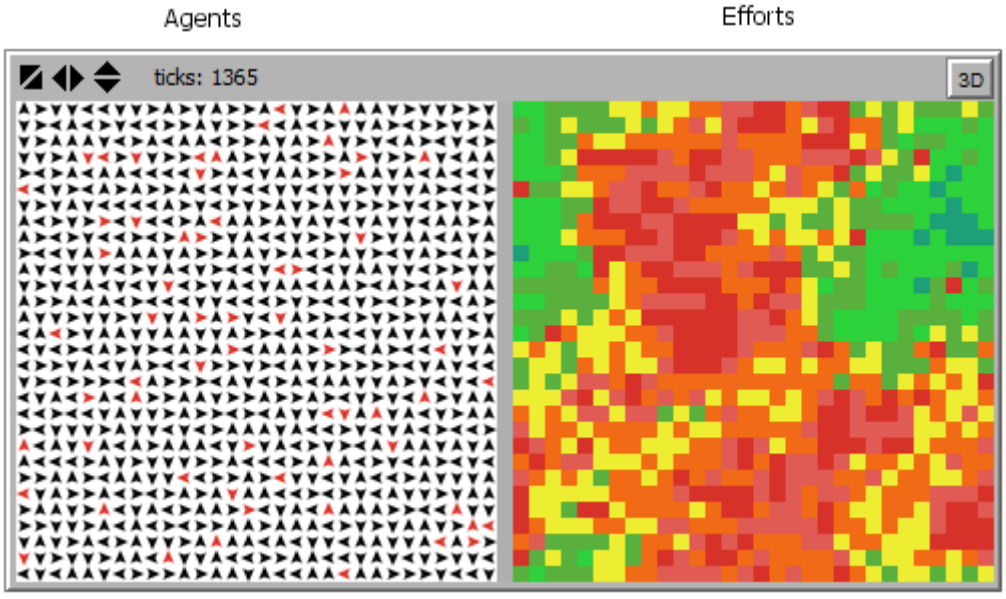
— Une population constituée seulement d'agents « shrinking effort »
Les agents "shrinking effort" adaptent leurs effort en réduisant à moitié l'effort de leur dernier partenaire. Après une interaction entre les agents i et j, ils mettent leur effort à jour à : $e_i(t) = e_j(t-1)/2$ et $e_j(t) = e_i(t-1)/2$. Cela entraîne une décroissance constante de l'effort moyen et mène à un équilibre des efforts en "null effort".

— Une population constituée de réplicateurs avec un seul agent à l'effort fixe
Les agents "replicator" reproduisent l'effort de leur dernier partenaire, donc lorsque deux agents "replicator" jouent le jeu entre eux, ils s'échangent leur valeur, en revanche lorsqu'un agent "replicator" interagit avec un agent dont l'effort est fixe à chaque tick, il reproduit cette valeur mais la valeur de son ancien effort est perdu. En simulant une telle configuration les agents "replicator" propagent la valeur d'effort fixe et ainsi après N-1 (N étant le nombre total d'agents) interaction de l'agent à effort fixe avec un agent "replicator" ayant un effort différent, on a un équilibre des efforts en la valeur de l'effort fixe.

— Une population constituée de réplicateurs avec un seul agent rationnel
L'agent rationnel exerce un effort qui tend vers l'équilibre de Nash du jeu, cela correspond à un effort d'environ 0.921007 avec la fonction de profit utilisée. Les agents réplicateurs vont progressivement adopter cette valeur et la propager en interagissant entre eux.
Pour obtenir ce résultat, avec un monde de dimension 30x30, en utilisant 899 agents "replicator" et 1 agent "rational", 81356 ticks ont été nécessaires pour atteindre un effort moyen de 0.921006.
— Une population de réplicateurs avec un seul agent à effort maximal
Cette configuration est inclue dans celles du point 3, la simulation converge donc vers une "corporate culture" où les agents exercent l'effort maximal.

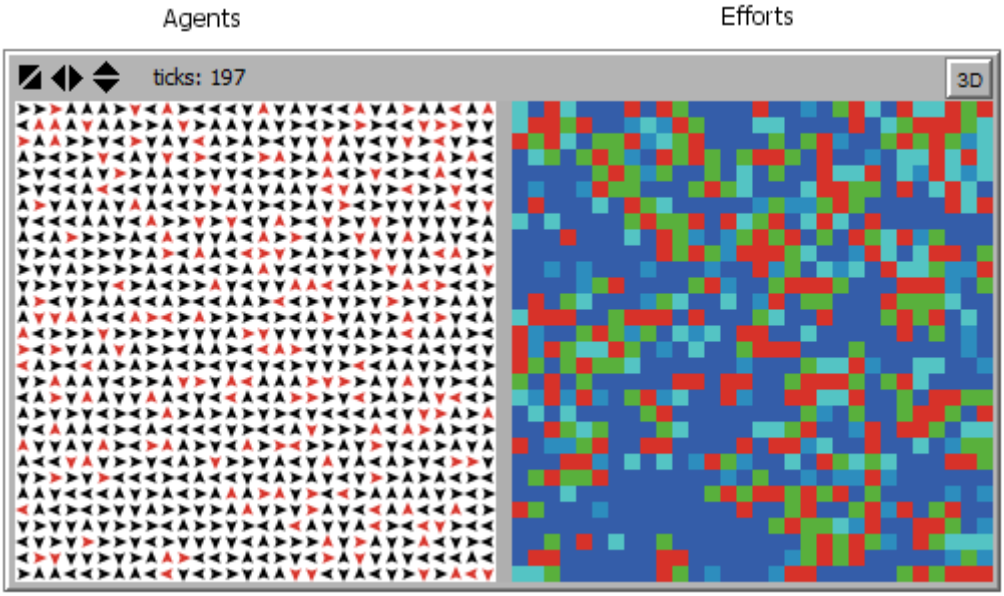
3.2 Question 2

Voici certains cas où aucun équilibre est atteint :
— Une population composée de 93% d'agents "effort comparator" et de 7% d'agents "high effort" ne permet pas d'obtenir un équilibre, en effet les agents comparateurs produiront un effort plus élevé là où la densité d'agent "high effort" est plus élevé. Exemple avec en rouge les agents "high effort" et une population totale de 900 :



Il est possible d'observer un résultat similaire en remplaçant les agents "effort comparator" avec des agents "profit comparator"
— Une population composée uniquement d'agents "replicator" ne converge pas non plus. Les valeurs initiales sont générées aléatoirement et ensuite à chaque interaction les agents s'échangent simplement leurs efforts.
— Une population constituée de différents types d'agents à effort fixe ne converge pas vers un équilibre, car ces agents n'adaptent pas leurs efforts, une "corporate culture" ne peut pas être atteinte.
— Une population constituée d'un mélange d'agents "shrinking effort" et

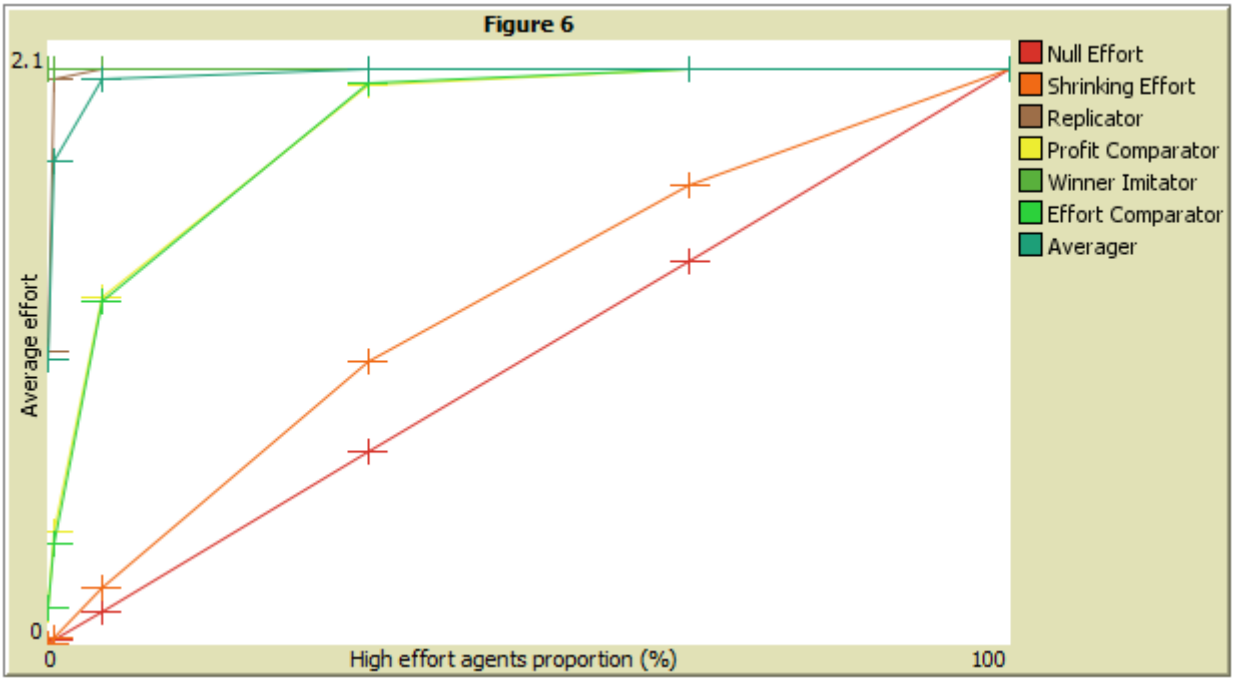
"high effort" ne convergera pas non plus. en effet, les agents "shrinking effort" copient la moitié de l'effort de leur partenaire après une interaction alors que les autres sont des agents à effort fixe. On obtient ainsi un résultat similaire au premier cas évoqué. Exemple avec 80% d'agents "shrinking effort" :



3.3 Question 3

Nous avons choisi de créer un système de simulation paramétrable pour tester plusieurs configurations dans le but d'obtenir les résultats présentés dans l'article, en faisant varier le nombre d'agents par exemple.
Nous utilisons un critère d'arrêt basé sur l'écart-type des efforts des agents, lorsque la valeur absolue de la différence entre l'écart-type actuel et celui du tick précédent est inférieure au paramètre *stdToleranceSimulation* pendant *nbTestsSimulation* ticks consécutifs. On récupère alors la moyenne des moyennes d'effort relevées sur les *nbTestsSimulation* ticks.
Nous calculons ainsi l'effort moyen des agents dans les configurations demandées puis nous utilisons les plots de NetLogo pour afficher les résultats.
Le critère *stdToleranceSimulation* augmente au cours de la simulation si le nombre de ticks devient trop important.

Pour les figures 6 et 7 nous avons choisi de simuler avec 900 agents, une tolérance initiale de 0.0001 et 500 tests consécutifs.
Tout d'abord la figure 6 :



L'objectif était d'étudier l'effet provoqué sur l'effort moyen par l'ajout d'agents "high effort" dans une population homogène d'agents à rationalité limitée. Il y a ainsi 7 types d'agents évalués :

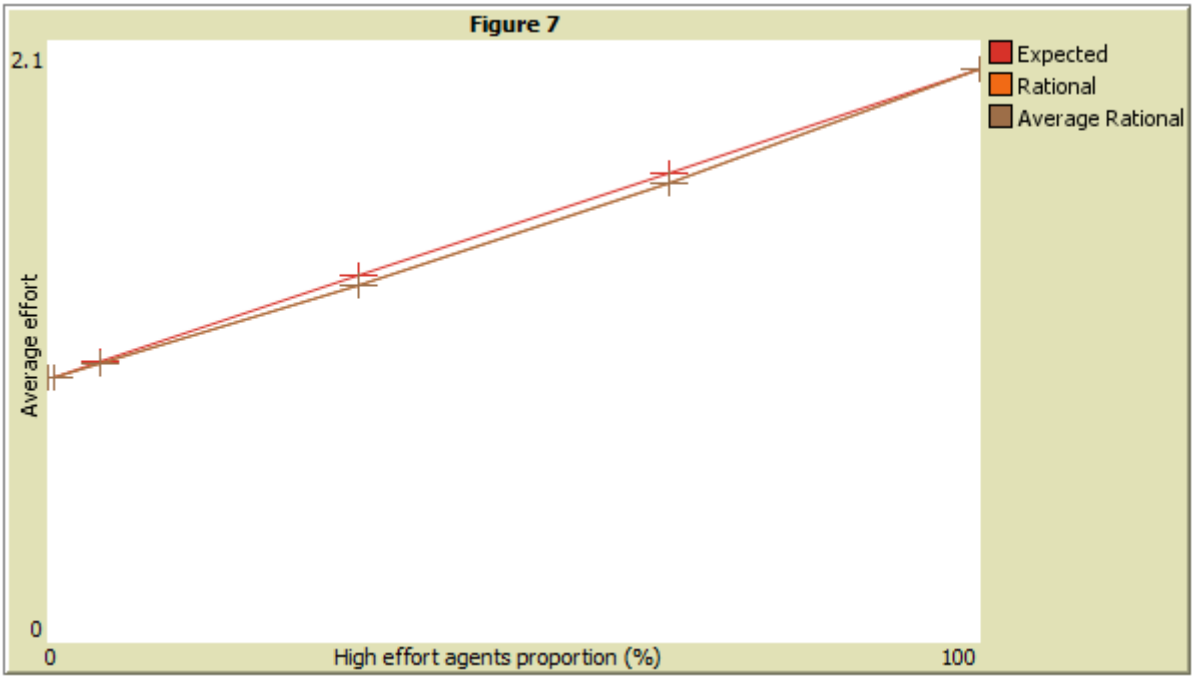
- Agents "null effort" : Ils n'évoluent pas en fonction de leurs intéractions donc l'effort est en réalité une fonction linéaire de la proportion P d'agents "high effort" : $AverageEffort = P * 2.001 + (1 - P) * 0.0001$. On obtient bien ce résultat attendu.
- Agents "shrinking effort" : Une population homogène d'agents "shrinking effort" converge vers un effort minimum. En ajoutant des agents "High effort", les agents "shrinking effort" à proximité vont fournir un effort moyen plus élevé car chaque interaction avec un "high effort" passe leur effort à $2.001/2 = 1.0005$. On constate bien l'amélioration sur l'effort moyen, ce qui permet aux agents "shrinking effort" de dépasser les agents "null effort".
- Agents "replicator" : L'ajout d'un seul agent "high effort" est suffisant pour converger vers une "corporate culture" où tous les agents produisent un effort maximal. Cela s'explique par le fait qu'une interaction entre un "replicator" et un "high effort" fait copier la valeur par l'agent "replicator", ce qui augmente l'effort moyen, là où une population homogène ne peut pas faire varier l'effort moyen car les valeurs d'effort sont simplement échangées à chaque interaction. On obtient bien un résultat cohérent.
- Agents "profit comparator" : L'ajout d'agents "high effort" créé un effet local où les agents "profit comparator" vont augmenter leur effort car ils obtiendront toujours un profit plus élevé que les agents "high effort". Nous n'avons pas trouvé de moyen de prédire le résultat en fonction de la proportion d'agents "high effort" car cela dépend, entre autre, de leur disposition sur la grille. Il est néanmoins logique que l'effort moyen aug-

mente, grâce aux agents "comparator" situés proches d'une forte densité d'agents "high effort".

— Agents "winner imitator" : Cas trivial, à l'initialisation, les "winner imitator" produisent un effort maximal, ce qui implique que tous les agents produisent un effort maximal et dispose tous du même profit, donc ils n'évoluent pas. Les résultats correspondent bien, avec un moyenne fixe à 2.001 peu importe la proportion d'agents "high effort".
— Agents "effort comparator" : De même que pour les agents "profit comparator" étant donné que les comparaisons de profit et d'effort sont identiques avec cette fonction de profit.
— Agents "averager" : Comme pour les agents "replicator", un seul agent "high effort" est suffisant pour converger vers un effort maximal pour tous les agents "averager". En effet, quand deux agents "averager" interagissent, ils obtiennent le même effort en conséquence et la moyenne d'effort globale est conservée. Lors d'une interaction avec un agent "high effort", la moyenne augmente car ce dernier n'ajuste jamais son effort. On obtient des résultats cohérents, même si l'effort moyen n'atteint pas exactement la valeur de l'effort maximal dans les configurations avec peu d'agents "high effort" car cela demanderait un temps très important.

La figure 7 représente une expérience similaire, en étudiant cette fois les agents rationnels :

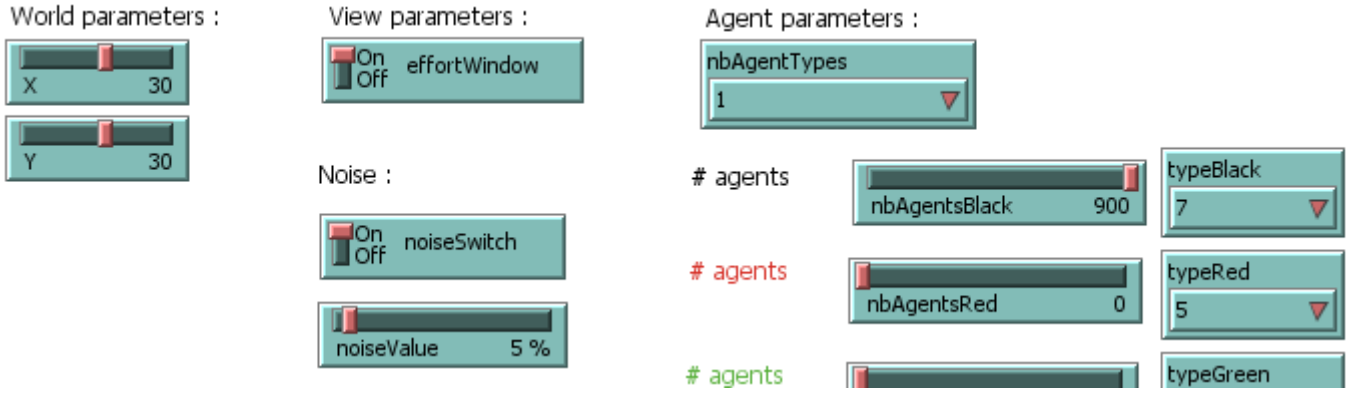
% High Effort	0,0%	0,6%	5,6%	33,3%	66,7%	100,0%
Expected	0,92101	0,92823	0,97348	1,25872	1,59225	2,00100
Observed	0,92101	0,92612	0,97317	1,24356	1,60308	2,00100



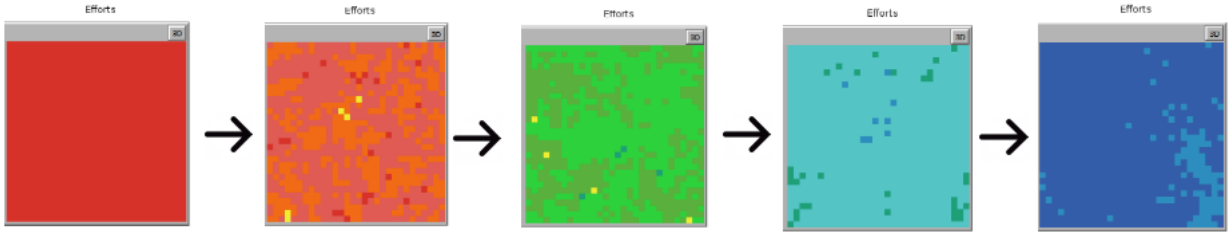
Nos résultats correspondent bien à ceux présentés dans l'article.Pour ces simulations nous avons paramétré une tolérance de 0,0005, en réduisant cette tolérance nos valeurs observées se rapprochent des valeurs espérées.

3.4 Question 4

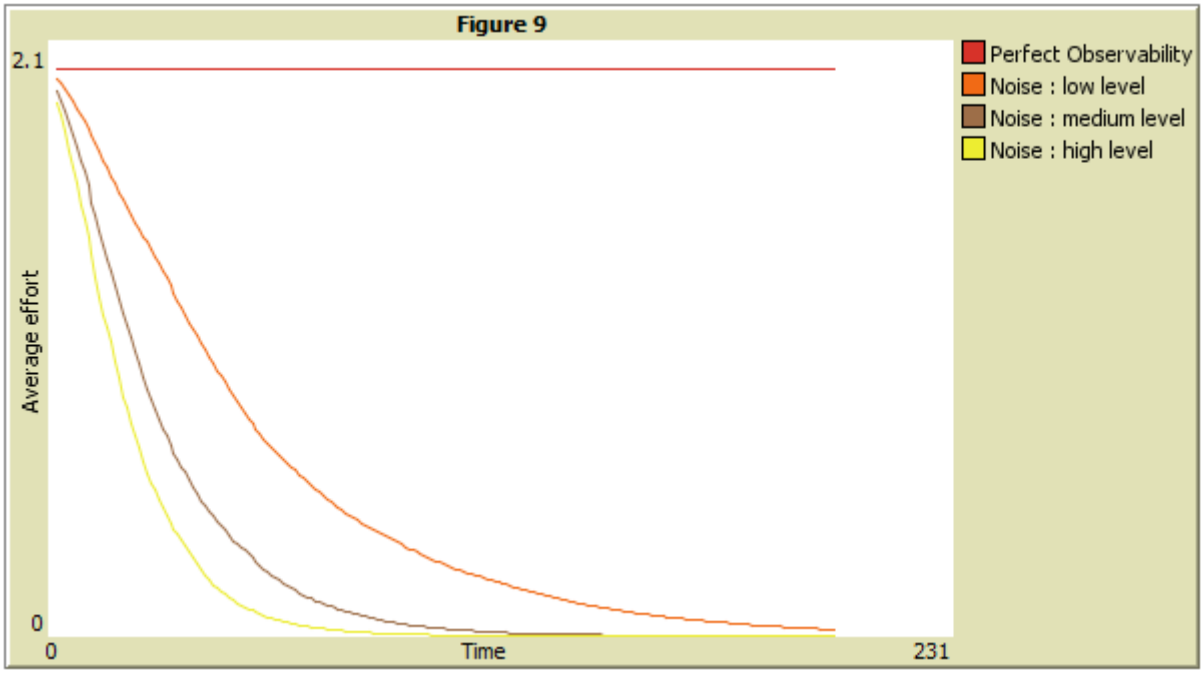
Nous avons implémenté le système de bruit présenté dans l'article en ajoutant comme contrainte que l'effort observé respecte les limites d'effort minimum et maximum.
Pour reproduire la figure 8, il suffit de choisir les paramètres suivants :



Il est possible de changer noiseValue pour voir évoluer plus ou moins rapidement les efforts.



Pour reproduire la figure 9, nous avons simulé 4 situations avec 500 agents "winner imitator" en faisant varier la valeur de bruit, noiseSwitch off pour obtenir l'observabilité parfaite, puis noiseSwitch on et noiseValue de 15% pour niveau faible, 30% pour niveau moyen et 45% pour niveau élevé.
On relève ensuite l'effort moyen des agents sur 200 ticks pour l'afficher dans le plot.



Les résultats obtenus sont cohérents, en effet les agents "winner imitator" débutent tous avec un effort maximal puis évoluent en se basant sur un effort faussé de leur partenaire. La fonction de profit est telle que si l'effort que l'on fournit est supérieur à celui de notre partenaire, on obtient un profit plus faible que ce dernier, et inversement. Ainsi, les agents "winner imitator" conservent leur effort si celui-ci est inférieur ou égal à celui de leur partenaire.
En revanche, si l'effort de leur partenaire est inférieur au leur, ils vont copier cette valeur. Le bruit fait parfois observer des valeurs plus faibles que la réalité, cela ayant pour conséquence une surestimation du profit du partenaire, ce qui

amène les agents à réduire leur effort.
Comme ces agents ne peuvent pas augmenter la valeur de leur effort avec la fonction de profit utilisée, on ne peut que converger vers un effort nul. Une valeur de bruit plus élevée permet d'observer des valeurs plus faibles, ce qui réduit plus rapidement l'effort des agents.

3.5 Question 5

Il serait intéressant de rajouter un système de hiérarchie au modèle, avec possiblement une gestion de licenciement pour les agents ne produisant aucun effort.
L'ajout d'un concept d'effort total minimum pour assurer le fonctionnement de l'entreprise, en modifiant la fonction de profit.

3.6 Question 6