Répartition des tâches : coopération et apprentissage par renforcement[1]

SAMUEL DELEPOULLE[2], PHILIPPE PREUX[3] ET JEAN-CLAUDE DARCHEVILLE[4]

(10 mars 1998)

Mots clé :
apprentissage par renforcement, coopération, division du travail, spécialisation, système non supervisé.

Résumé :
De nombreux systèmes nécessitent l'action collective de plusieurs intervenants. Si l'action peut être découpée en tâches, comment peuvent-ils se répartir la charge de travail pour optimiser leur intervention ? Nous proposons d'utiliser le modèle de l'apprentissage par renforcement pour apporter une solution à ce problème. Nous décrivons une architecture d'agents « essai/erreur » très simple qui s'adapte à leur environnement en fonction des conséquences de leurs actions. Dans le cas ou le problème peut être résolu analytiquement, nous montrons qu'un partage optimal ou proche de l'optimum est généralement obtenu. Dans le cas ou l'environnement ne peut plus être optimisé analytiquement, nous montrons que le même système d'agent parvient a obtenir de bonnes performances.

1. Quels modèles pour des agents coopératifs?

Les systèmes industriels nécessitent fréquemment l'interaction et la coordination de plusieurs intervenants. Le « pilotage » d'avions, de navires commerciaux (Hoc, 1996) ou d'unités industrielles de production ne peut pas être réalisé complètement par une seule personne (Driskell et Salas, 1992). Dès lors, chaque intervenant se voit assigner un rôle qui consiste généralement à effectuer un certain nombre de tâches. L'ensemble de ces tâches doit être réalisé pour que le système fonctionne. La question se pose alors de savoir comment ces tâches sont réparties entre les différents acteurs.

Une solution consiste à confier à un acteur particulier le rôle de répartir les tâches entre les intervenants. Ce rôle de superviseur consiste à identifier chacune des taches afin de la confier à l'un ou l'autre des intervenants. Ce type de solution pose des contraintes sur le système :

- le superviseur doit pouvoir connaître l'ensemble des taches exigées par l'environnement. En effet, si une tâche n'est pas détectée par le superviseur, elle n'a aucune raison d'être effectuée. Ceci peut

[1] Cette recherche est soutenue par le Conseil Régional Nord - Pas de Calais. (Contrat n° 97 53 0283)

[2] delepoulle@univ-lille3.fr, Université Lille 3, Equipe Dynamique, Evolution des Comportements et des Apprentissages, B.P. 149, 59653 Villeneuve d'Ascq Cedex

[3] preux@lil.univ-littoral.fr - Laboratoire d'Informatique du Littoral, B.P. 719, 62228 Calais Cedex

[4] darcheville@univ-lille3.fr, Université Lille 3, Equipe Dynamique, Evolution des Comportements et des Apprentissages, B.P. 149, 59653 Villeneuve d'Ascq Cedex

mettre en péril le système dans son ensemble ;
- le superviseur doit pouvoir connaître l'ensemble des intervenants du système ainsi que les tâches qu'ils effectuent. C'est cette connaissance qui guide le superviseur dans la répartition des taches.

L'utilisation de ce modèle suppose qu'un acteur au moins (le superviseur) possède une connaissance exhaustive de la situation. Or, le partage des tâches s'effectue le plus souvent dans des environnements complexes, qui ne peuvent être connus par un seul intervenant (Amalberti, 1996). La coordination de l'ensemble des acteurs par l'un d'entre-eux devient alors un problème important (Sundström, 1993).

Une autre solution est d'imaginer des systèmes dans lesquels les agents ajustent leur comportement en temps réel et en fonction de leurs conséquences. C'est - notamment - pour satisfaire à cette exigence qu'ont été élaborés les algorithmes d'apprentissage par renforcements. (Sutton, 1991 et Sutton et Barto, 1990). La question que nous voulons poser ici est de savoir si de tels agents peuvent parvenir à un partage optimal des tâches si leur comportement évolue par renforcement.

1.1. L'apprentissage par renforcement

L'apprentissage par renforcement est une forme d'apprentissage dans laquelle le comportement d'un agent est modifié en fonction de ses interactions avec son environnement. L'agent émet des comportements qui sont suivis de conséquences. Ces conséquences (les renforcements, entres autres) vont elle même modifier la probabilité d'apparition du comportement. Le terme d'apprentissage par renforcement provient initialement de la psychologie comportementale, en particulier des études sur l'apprentissage animal (Skinner 1968). On s'intéresse, par exemple au comportement d'un animal devant appuyer sur une pédale pour obtenir de la nourriture. Au départ, l'animal appuie sur la pédale par hasard et reçoit une petite quantité de nourriture. Le comportement d'appuyer sur la pédale sera donc renforcé par la nourriture, c'est à dire que l'animal va répéter ce comportement.

Les algorithmes d'apprentissage par renforcement ont été imaginés pour simuler ces comportements. Pour cela, un algorithme d'apprentissage par renforcement est caractérisé par trois fonctions. Une fonction de réponse, une fonction de récompense et une fonction de valeur.

1.1.1. La fonction de réponse

La fonction de réponse détermine le comportement de l'agent à un moment donné. L'agent perçoit son environnement via un certain nombre d'entrées perceptives. La fonction de réponse associe à chaque pattern d'entrée perceptive un pattern de sortie. L'ensemble de ces patterns de sortie constituent le répertoire comportemental de l'agent.

1.1.2. La fonction de récompense

La fonction de récompense définit la conséquence du comportement. C'est un nombre réel, positif ou négatif. D'un point de vue biologique, la fonction de récompense formalise les notions de douleur (une conséquence négative) ou de plaisir (conséquence positive). L'agent cherche à maximiser les récompenses. Pour cela, la fonction de récompense modifie la fonction de réponse.

1.1.3. La fonction de valeur

La fonction de valeur permet d'évaluer l'état le plus profitable à long terme pour optimiser les récompenses. La fonction de valeur correspond en quelques sorte à une estimation (une prédiction) des récompenses à long terme en fonction de la situation.

L'utilisation conjointe d'une fonction de récompense et d'une fonction de valeur permet à l'algorithme d'adapter son comportement dans des situations où il y a discordance entre les conséquences à court terme et à long terme. Il est important, par exemple, de ne pas éliminer du répertoire comportemental certaines actions qui se traduiraient par une conséquence négative à court terme (douleur, fatigue) mais qui auraient un intérêt sur le long terme.

1.2. Nécessité d'optimiser exploration et exploitation

Comme nous venons de le voir, l'agent tend donc à maximiser la quantité de renforcement. Dès que des conséquences positives suivent un comportement l'agent est placé face à un dilemme : soit émettre à nouveau ce comportement (exploiter la source de renforcement) soit émettre un autre comportement (explorer d'autres sources de renforcement possibles). Comme nous l'avons précisé, l'agent ne dispose pas de modèle de l'environnement. Celui-ci doit donc toujours chercher à optimiser l'équilibre entre exploitation et exploration. L'optimisation de cet équilibre est vitale pour un organisme vivant. De nombreuses méthodes mathématiques ont tenté de résoudre cette question. Sutton et Barto (1998) discutent des avantages des méthodes non supervisées sur les méthodes supervisées lorsqu'il faut optimiser cet équilibrage.

D'un point de vue pratique, les algorithmes d'apprentissage par renforcement ont également montré leur intérêt. Des algorithmes de ce type ont été utilisés pour simuler des activités telles que l'orientation du regard (Foner et Maes, 1994), le jeu de Backgamon (Tesauro, 1992) ou l'optimisation de cultures céréalières (Attonaty, Chatelin, Garcia et Seydina, 1997). Les algorithmes génétiques (Mitchell, 1997) ou les algorithmes imitant des colonnies d'insectes (Theraulaz, 1994) en sont d'autres exemples.

Nous pensons que la situation de partage dynamique des tâches au sein d'une communauté d'agents se prête particulièrement bien à l'utilisation d'algorithmes d'apprentissage par renforcement car le comportement optimal d'un agent est loin d'être prédictible à tout instant. Des agents autonomes, en l'absence de coordination globale peuvent-ils, par auto-adaptation acquérir un comportement collectif performant? C'est la question que nous nous proposons d'étudier, c'est pourquoi il nous faut présenter quelques caractéristiques formelles de cette situation.

2. Quelle division optimale du travail dans les situations coopératives ?

Certaines tâches coopératives se prêtent à la spécialisation des individus ; dans ce cas il existe souvent une division optimale du travail (Leplat, 1994). En effet, si tous les agents effectuent la même tâche, l'efficacité du groupe risque d'être faible. En revanche s'il n'existe aucune intersection dans les tâches qu'effectuent les agents, l'échec de l'un des agents peut mettre en péril l'ensemble du groupe. Il parait donc raisonnable de penser que la performance optimale du groupe est atteinte si les tâches sont réalisées par plusieurs agents tout en diversifiant les tâches de chaque agent. C'est le cas, par exemple dans une situation imaginée par Zajonc (1966) ou plusieurs personnes doivent retenir un ensemble d'items. Zajonc imagine une situation ou un groupe de sujets doit retenir un ensemble de syllabes sans signification. Zajonc pose la question de savoir s'il vaut mieux que chaque sujet retienne toutes les syllabes ou s'il vaut mieux, au contraire, répartir les items. Il montre que pour maximiser la performance, chaque syllabe doit être retenue par plusieurs sujets mais qu'il ne faut pas non plus « surcharger » les sujets sous peine de voir à nouveau la performance décroître.

D'un point de vue formel, soient N sujets qui ont à réaliser H tâches d'égale difficulté, appelons p sa probabilité de réussite dans l'une des H parties de la tâche. Nous supposerons que, d'une part, chaque sujet est également « adroit » dans la réalisation de cette tâche et que d'autre part, cette probabilité ne varie pas dans le temps. Admettons que chaque sujet effectue y tâches ; réciproquement une tâche est réalisée par x individus.

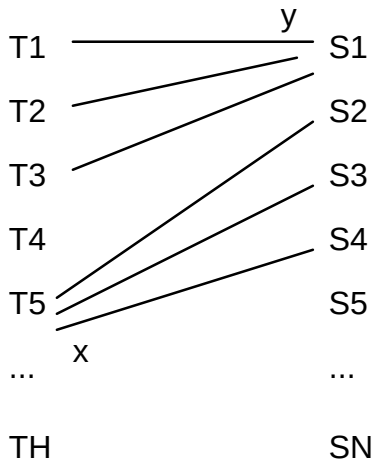

Figure 4 : H tâches sont réparties entre N sujets. (chaque sujet réalise y tâches et chaque tâche est effectuée par x sujets)

Nous pouvons d'ores et déjà écrire :

$$x \cdot H = y \cdot N \Leftrightarrow \frac{y}{x} = \frac{H}{N} \qquad (1)$$

Nous nous intéressons à la probabilité P qu'une tâche soit exécutée avec succès. Nous pouvons dire que la probabilité d'échec est de 1-p. Comme n sujets ont tenté cette tâche, globalement la probabilité d'échec est de $(1-p)^x$. La probabilité de réussite peut donc s'écrire :

$$P = 1 - (1-p)^x$$

Nous cherchons a connaître le maximum de cette fonction. Pour cela cherchons les valeurs de x qui annulent sa dérivée.

$$\frac{dP}{dx} = \frac{d\left[1-(1-p)^x\right]}{dx} = -\frac{d(1-p)^x}{dx}$$

On peut donc écrire :

$$\frac{dP}{dx} = -\left\{x(1-p)^{x-1}\left[\frac{d(1-p)}{dx}\right] + (1-p)^x \ln(1-p)\right\}$$

d'ou

$$\frac{dP}{dx} = (1-p)^x\left[\frac{dp}{dx}\left(\frac{x}{1-p}\right) - \ln(1-p)\right]$$

Rappelons que notre but était de résoudre l'équation :

$$\frac{dP}{dx} = 0$$

Eliminons dès à présent le cas trivial ou (1-p) = 0, ce qui revient à dire que p vaut un. C'est dire que la division est optimale, dans tous les cas, si la probabilité de réussite est totale. Pour notre propos, ce cas ne présente pas grand intérêt. En revanche étudions la (les) solution(s) du deuxième membre :

$$\left[\frac{dp}{dx}\left(\frac{x}{1-p}\right) - \ln(1-p)\right] = 0 \quad (2)$$

Pour cela, il nous faut faire des hypothèses quant à la relation qui existe entre p et y, respectivement la probabilité de succès et le nombre de tâches effectuées par un sujet. Globalement, nous pouvons penser que p est une fonction décroissante de h. Il est possible, par une passation individuelle de quantifier cette relation. Prenons l'exemple d'une modélisation selon une loi exponentielle négative. On pourra par exemple écrire :

$$p = e^{-(ky)^2} \quad (3)$$

où k est un paramètre spécifiant la difficulté de la tâche.

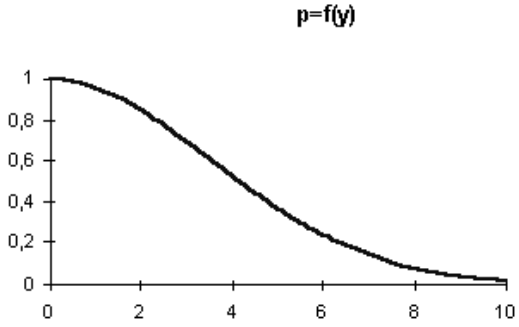

Figure 5 : probabilité de réussite en fonction du nombre de tâches par sujet.

la relation (3) peut aussi s'écrire :

$$\ln(p) = -(ky)^2 \quad (4)$$

donc

$$\frac{dp}{dx} = e^{-(ky)^2}\left(-k^2\right)\frac{dy^2}{dx} = e^{-(ky)^2}\left(-k^2\right)2y\frac{dy}{dx} \quad (5)$$

donc,

$$\frac{dy}{dx} = \frac{H}{N} \quad (6)$$

et finalement

$$\frac{dp}{dx} = e^{-(ky)^2}\left(-k^2\right)2y\frac{H}{N}$$

Ce qui permet de résoudre (2)

$$\left[e^{-(ky)^2}\left(-k^2\right)2y\frac{H}{N}\left(\frac{x}{1-p}\right) - \ln(1-p)\right] = 0$$

Qui devient, avec les relations (3) et (1) :

$$\left[p\left(-k^2\right)2y^2\left(\frac{1}{1-p}\right) - \ln(1-p)\right] = 0$$

Et finalement, avec (4)

$$\frac{2p\ln(p)}{1-p} - \ln(1-p) = 0$$

On peut facilement démontrer que cette fonction admet une unique solution sur l'intervalle]0 ;1[. Par recherche dichotomique, on trouve p=.8396

De (4), on peut déduire que

$$y^2 = \frac{\ln(p)}{-k^2}$$

rappelons que comme p est compris entre 0 et 1,

$$\ln(p) \le 0$$

donc

$$y = \sqrt{\frac{\ln(p)}{-k^2}}$$

Le nombre h de tâches par sujet qui maximise P est donc de 2,1 (si k=.20) comme le montre la figure 6.

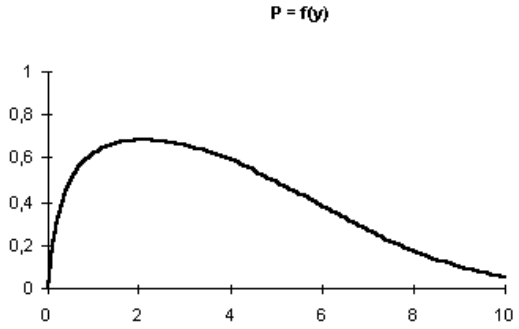

Figure 6 : évolution de la performance du groupe en fonction du nombre h de tâches par sujet.

En conclusion, h ne dépend ni de H ni de N : si la taille du groupe augmente, chaque tâche sera effectuée par plus de sujets mais la valeur optimale de tâches par sujet reste identique. La figure 7 présente l'évolution de la réussite (en probabilité) du groupe à une tâche en fonction du rapport H/N. Si la performance augmente effectivement quand H/N diminue (c'est-à-dire qu'il y a moins de tâches par sujet), l'optimum de cette performance reste identique (proche de 2 dans notre exemple).

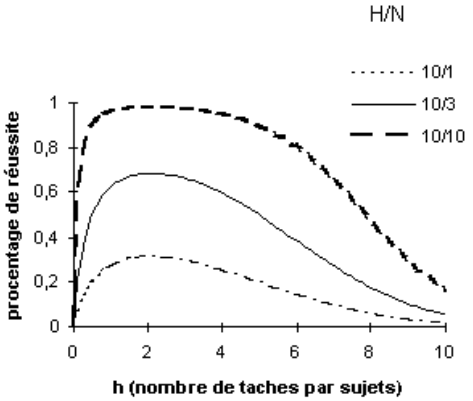

Figure 7 : évolution de la probabilité P de réussite à l'une des tâches en fonction du nombre h de tâches par sujet et du rapport H/N

Le seul facteur qui influence réellement l'optimum est la fonction p qui détermine la difficulté de la tâche en fonction du nombre d'items. Dans notre exemple, l'évolution du paramètre k modifie les choses. Si k augmente (la difficulté de la tâche augmente) il faudra moins de sujets par tâche, inversement si k diminue (la tâche est plus facile) l'optimum sera atteint quand les sujets font plus de tâches.

Cette démonstration illustre que pour un ensemble de tâches données, il existe une division optimale du travail. Les comportements doivent posséder un certain degré de recouvrement afin d'éviter les erreurs individuelles. D'un autre côté, si les individus s'engagent dans trop de tâches, la probabilité d'un échec est élevée, ce qui détériore la performance globale. La question que nous devons encore nous poser est de savoir si les comportements individuels peuvent s'adapter pour converger vers l'optimum que nous avons décrit. Le modèle que nous avons proposé suppose que chaque tâche ait la même probabilité de succès. Dans la réalité, c'est rarement le cas. Que se passe-t-il lorsque certaines tâches sont plus difficiles ? Enfin nous avons initialement supposé qu'il n'y avait aucun apprentissage des tâches, c'est-à-dire que la probabilité de succès est indépendante de la répétition de la tâche. Encore une fois, dans la réalité, les tâches complexes engendrent des apprentissages. Il nous paraît donc important soit de maîtriser ce facteur, soit d'évaluer son impact. Nous pensons que l'apprentissage de certaines tâches tend à favoriser ce phénomène de spécialisation.

Afin de mieux comprendre la dynamique de cette situation, nous allons la mettre en œuvre avec des automates. Nous commencerons, ici, en utilisant un modèle extrêmement simple d'apprentissage par renforcement.

3. Division du travail : une simulation

3.1. Modèle d'agent utilisé

N agents peuvent choisir y tâches parmi H. La probabilité qu'une tâche soit effectuée correctement dépend du nombre de tâches tentées par l'agent. La probabilité de succès est une fonction décroissante du nombre de tâches tentées. Cette relation est représentée par l'équation 3. Pour la suite, nous fixerons le paramètre k, qui détermine la difficulté des tâches (k=0,2). A chaque itération, les agents sont renforcés si et seulement si toutes les tâches ont été menées à bien.

Chaque agent, à chaque itération, choisit un nombre de tâches à effectuer. Si à l'essai considéré, il a reçu le renforcement, il choisira le même nombre de tâches pour l'essai suivant ; dans le cas contraire, il choisira aléatoirement un nombre de tâches uniformément distribué entre 0 et H.

Cette architecture d'agent constitue un algorithme d'apprentissage par renforcement extrêmement élémentaire (par « essai-erreur »). La fonction de réponse consiste à choisir aléatoirement une ou plusieurs tâches à tenter. La fonction de récompense maintient cette réponse dans le temps s'il y a eu un renforcement et la fonction de valeur est ici absente.

3.2. Résultats

3.2.1. Comparaison entre la simulation et le modèle

Les agents artificiels ont été testés dans 49 situations en faisant varier N (le nombre d'agents) et H (le nombre de tâches). Le tableau 1 présente le nombre de tâches moyen après 100 itérations de l'algorithme observé au sein du groupe.

H\N	1	2	3	4	5	6	10
1	**1** (0)	**1** (0)	**0,4** (0,24)	**0,2** (0,16)	**0** (0)	**0** (0)	**0** (0)
2	**0,6** (0,64)	**1,6** (0,24)	**1,4** (0,24)	**0,8** (0,16)	**1** (0)	**1** (0)	**1** (0)
3	**1,6** (1,84)	**1,6** (1,44)	**1,4** (0,64)	**1,4** (0,24)	**1,4** (0,24)	**1,8** (0,16)	**1,4** (0,24)
4	**1,8** (3,36)	**2,4** (1,04)	**2** (1,2)	**2** (0,4)	**2** (0,8)	**2,2** (0,16)	**1,8** (0,16)
5	**3** (2,8)	**3,6** (0,24)	**2,2** (1,36)	**2** (0,4)	**2,4** (0,64)	**2,4** (0,64)	**2,2** (0,16)
6	**4** (4,4)	**3** (0,4)	**2,6** (0,24)	**2,8** (0,16)	**3,4** (0,64)	**2,4** (0,64)	**2,8** (0,16)
10	**5,6** (15,04)	**5,2** (1,36)	**3,6** (2,64)	**4** (1,2)	**4,6** (3,04)	**4** (1,6)	**4,2** (0,96)

Tableau 1 : Nombre moyen de tâches au sein d'un groupe de N agents ayant à choisir parmi H tâches. Le nombre entre parenthèses est la variance. Moyenne et variance sont calculées sur 5 exécutions de l'algorithme.

La convergence théoriquement prédite à 2 tâches par agent s'observe effectivement lorsque N est relativement proche de H. Si le nombre de tâches diminue (H<N), le nombre moyen de tâches par agent au sein du groupe est inférieur à l'optimum théorique. En effet, il suffit qu'une partie des agents réalise les tâches pour qu'elles soient menées à bien et donc que l'ensemble du groupe reçoive l'agent renforçateur. D'un autre côté, s'il y a trop de tâches relativement au nombre d'agents (H>N), l'algorithme ne converge pas (la moyenne vaut approximativement H/2 et la variance est élevée). En fait, la probabilité de recevoir le renforcement est faible car les agents sont obligés de choisir beaucoup de tâches, ce qui diminue d'autant leur performance. Nous pouvons nous apercevoir de ce phénomène en étudiant le nombre moyen de renforcements pour chacune de ces situations (tableau 2).

Plus le nombre d'agents augmente, plus la probabilité que le groupe soit renforcé est élevée. Ceci explique que dans les situations où N est nettement supérieur à H, les agents peuvent être renforcés même si la majorité n'effectue aucune tâche. On constate à l'extrême ce phénomène pour une tâche et dix agents : il suffit que quelques uns la réalisent pour que le groupe soit toujours renforcé. Ce qui conduit à un comportement d'exploitation massif. À l'inverse quant N est grand, il devient quasiment impossible d'obtenir le renforcement et le comportement des agents devient aléatoire.

H\N	1	2	3	4	5	6	10
1	**89,8** (10,16)	**98,2** (5,76)	**98,8** (1,36)	**99,6** (0,64)	**99** (0,4)	**100** (0)	**100** (0)
2	**43** (3,2)	**82,8** (119,4)	**93** (38)	**93,8** (4,56)	**97,4** (1,84)	**97,8** (1,36)	**99,8** (0,16)
3	**8,4** (1,84)	**43** (117,6)	**61,6** (90,64)	**80,2** (45,36)	**92,6** (11,84)	**95** (10,8)	**99,6** (0,24)
4	**1,2** (0,96)	**15,2** (16,56)	**36,4** (10,24)	**57,2** (30,56)	**69,8** (24,56)	**82,4** (24,64)	**97** (5,2)
5	**0** (0)	**3,4** (3,44)	**11,2** (2,96)	**24,6** (9,44)	**46,8** (11,36)	**53,4** (15,44)	**88,8** (24,6)
6	**0** (0)	**0** (0)	**4** (2)	**9,6** (9,84)	**18,2** (10,16)	**25** (18,4)	**70** (9,6)
10	**0** (0)	**0** (0)	**0** (0)	**0** (0)	**0** (0)	**0** (0)	**3,6** (1,04)

Tableau 2 : nombre moyen de renforcement du groupe. Les données présentent les moyennes et les variances, entre parenthèses, pour 5 exécutions de 100 itérations.

Dans les cas « médians », même si le nombre moyen de tâches se rapproche de l'optimum théorique, il reste une grande variabilité à l'intérieur même des groupes. En fait, il n'est pas rare que la moyenne du groupe converge vers l'optimum théorique alors qu'aucun agent ne présente ce comportement moyen. De même, nous constatons qu'un agent seul ne peut trouver l'optimum dès que le nombre de tâches excède quatre. Un groupe de six agents converge parfaitement pour le même nombre de tâches.

3.2.2. Cas non prévus par le modèle analytique

Les hypothèses nécessaires à l'analyse mathématique sont extrêmement contraignantes pour un système réel (nombre de tâche fixes, nombre d'agents fixes, tâches de difficultés identiques et ne variant pas dans le temps). L'intérêt de la simulation par des agents réactifs est de ne faire aucune de ces hypothèses. Notre simulation, en dépit de son extrême simplicité, doit pouvoir s'adapter à des situations impossibles à décrire formellement. Pour nous rendre compte de ce fait, étudions le comportement des agents dans le cas ou le paramètre k, spécifiant la difficulté de la tâche n'est pas identique pour toutes les tâches.

Nous prenons ici le cas de 2 agents ayant à effectuer 4 tâches. Le paramètre k vaut 0,01 pour deux tâches et 0,15 pour les deux autres. Afin de pouvoir mesurer la performance des agents dans cette situation, nous prendrons pour critère le nombre de fois (en pourcentage) où l'ensemble des tâches ont été réalisées en même temps. Cependant, en l'absence de modèle simple permettant de connaître le « bon » comportement, nous comparerons cette performance à celle d'agents dont les comportements sont aléatoires. Les résultats de cette simulation sont consignés dans le tableau 3.

	k = 0,1	k1=0,01 k2=0,15
Agents « renforcés »	52,3	50,3
Agents aléatoires	31,1	29.6

Tableau 3 : Nombre moyen fois ou toutes les tâches ont été réussies (sur 100 itérations). La moyenne est donnée pour 10 exécutions.

Dans ce cas, les tâches de difficulté différente occasionnent une performance moindre que si k est identique pour toutes les tâches[5]. Ce qui est plus important à noter est que le nombre de fois ou toutes les tâches ont été réalisées est supérieur si les agents sont sensibles au renforcement. Une analyse de variance (ANOVA) paramétrique en facteurs indépendants permet de conclure que cette différence est statistiquement significative. La valeur du test F(1,9) est de 245,0068. Nous pouvons donc conclure avec un risque d'erreur inférieur à 10^{-5}.

4. Discussion et perspectives

Dans le cadre d'un ensemble de tâches coopératives, il arrive que les comportements des agents se spécialisent. Il est possible de rendre compte de ce phénomène par le biais d'un modèle mathématique qui permet de déterminer le nombre optimal de tâches qui doit être réalisé par chaque agent. La simulation montre que, sans faire d'hypothèses sur la situation, des agents peuvent tendre vers un résultat proche.

Dans le cadre très strict du modèle d'environnement que nous avons proposé, une architecture supervisée pourrait suffire : l'analyse mathématique du système permet de connaître pour chaque agent le nombre optimal de tâches qu'il doit prendre en charge. Ce n'est plus le cas dès que la difficulté des tâches est variable. D'autre part, le comportement de l'agent utilisé ici est déterminé par une loi très simple (Cf. 3.1). Ceci explique que dans les situations extrêmes, on observe une sous ou une sur-convergence. Dans le cas ou trop d'agents sont présents, une minorité suffit à exécuter les tâches qui rendront le renforcement disponible pour tous. En revanche si les agents sont trop peu nombreux, la probabilité que toutes les tâches soient effectuées devient très faible. Donc les agents reçoivent peu de renforcement ; leur comportement n'est alors jamais sélectionné. Ce problème serait vraissemblablement mieux résolu par des modèles plus sophistiqués d'apprentissage par renforcement (Q-learning, TD models).

D'autre part, cette simulation exhibe une situation de coopération et de répartition des tâches sans communication explicite entre les agents. Communication que de nombreux auteurs considèrent néanmoins comme préalable à toute coopération. C'est le cas, par exemple de Akira Ito (1997) qui montre comment des agents peuvent coopérer dans le cadre du dilemme des prisonniers si il y a échange des informations relatives aux actions précédentes. C'est le cas aussi pour Abric (1987) pour qui la communication et la connaissance d'un but commun joue un rôle fondamental en situation sociale. Dans notre cas, nous constatons que les comportements des agents co-évoluent, s'adaptent en temps réels les uns aux autres et à l'environnement. Le résultat est un comportement collectif cohérent et coopératif sans concertation ni communication explicite.

[5] Même si la moyenne des coefficients k est identique dans les deux situations, il est normal que la difficulté moyenne ne soit pas équivalente. Rappelons en effet que la difficulté n'est pas une fonction linéaire de k (cf. équation 3)

Références bibliographiques :

Abric J.-C. (1987), Coopération, compétition et représentation sociale. Deleval, Fribourg.

Amalberti, R., (1996) La conduite de systèmes à risques. PUF. Collection le travail humain.

Attonaty J.M. , Chatelin M.-H., Garcia F. et Seydina M. N. (1997) Using extended machine learning and simulation technics to design crop management strategies. In Proc. of the 1st European Conference for Information Technology in Agriculture, Copenhagen.

Driskell, J. E. et Salas, E. (1992) Collective Behavior and Team Performance. *Human Factor, 34* (3)

Foner, L., et Maes, P., (1994) Payint Attention to What's Important: Using Focus of Attention to Improve Unsupervised Learning. in From Animals to Animats III. Proceedings of the third international conference on simuation of adaptive behavior.

Hoc, J.M. (1996) Supervision e contrôle de processus, La cognition en situation

dynamique. Presse Universitaire de Grenoble, Grenoble.

Ito, A. (1997) How Do Selfish Agent Learn to Cooperate. Proceeding of the Fifth International Workshop on the synthesis and simulation of living systems. C. Langthon et K. Shimohara (Eds), pp 185-192

Leplat, J. (1994) Ergonomie et activités collectives. In : Six, F. et Vaxevanoglou, X. (Eds). Les aspects collectifs du travail. Octarès Editions.

Mitchell,M. (1997) colonies d'insectes appliquees a la resolution de problemes : M. Dorigo et L. Gambardella, Ant Colony System: A Cooperative Learning Approcah to the Traveling Salesman Problem, IEEE Trans. on Evolutionary Computation, 1(1)

Skinner, B. F. (1968) The behavior of organisms : an experimental analysis. Englewood Cliff, Prentice Hall.

Sundsröm, G.A. (1993) Towards models of tasks and task complexity in supervisory control applications. *Ergonomics. Vol 36, n° 11*.

Sutton R. (1991) *Reinforcement Learning Architectures for Animats*. In From Animals to Animats. Proceedings of the first international conference on simuation of adaptive behavior.

Sutton, R., et Barto, A., (1990) Integrated architectures for learning, planning, and reacting bases on approximating dynamic programming. In Proceeding of the Senventh International Conference on Machine Learning. Morgan Kaufmann.

Sutton, R.,et Barto, A., (1998) Reinforcement Learning: An introduction. Cambridge, MA: MIT Press.

Tesauro G. (1992) *Practical issues in temporal difference learning*. In Sutton, edior, *Renforcement learning*. Kluwer Academic Publishers.

Theraulaz, G.(1994) Du super-organisme à l'intelligence en essaim : modèles et représentations du fonctionnement des sociétés d'insectes, dans E. Bonabeau, G. Theraulaz (coord.), Intelligence Collective, Hermes.

Zajonc, R.B. (1966), Social Psychology : an experimental approch. Wadsworth Publishing Company, Inc. Belmont, California, USA