Formation de coalition pour agents rationnels

Guillaume Vauvert – Amal El Fallah Seghrouchni
Laboratoire d'Informatique de Paris Nord
Université Paris XIII
Avenue Jean-Baptiste Clément
93430 Villetaneuse
{guillaume.vauvert, amal.elfallah}@lipn.univ-paris13.fr

Résumé

Les coalitions sont des organisations qui apportent dynamisme et flexibilité, ce qui permet d'appréhender des situations de coopération et de compétition dans des environnements ouverts. Dans un contexte économique distribué, les agents sont nécessairement hétérogènes, égoïstes et libre en comportement et en raisonnement, ce qui nous a amené à proposer un protocole qui privilégie la liberté des agents.
Appliqué à la répartition de tâches entre agents, le protocole est ouvert, distribué et égalitaire. Il est basé sur un échange de préférences calculées en utilisant des critères qualitatifs et quantitatifs et il permet à différentes formes de rationalités et à différentes stratégies de coexister. Nous allons montrer qu'il termine et que l'atteinte d'un consensus est garantie moyennant une hypothèse faible.

1 Introduction

Les systèmes multi-agents (S.M.A) ont comme intérêt principal la collaboration d'agents. Différentes approches ont été étudiées pour supporter cette collaboration : la résolution distribuée de problèmes, les réseaux contractuels, les approches basées sur les organisations, les protocoles de négociation, la formation de coalition, *etc.*
Les coalitions permettent aux agents de satisfaire des besoins qui requièrent la synergie de compétences de différents agent comme, par exemple, dans le cadre de la résolution de tâches complexes pour lesquelles des agents agissant seuls seraient incapables ou moins efficaces.
Une coalition peut être définie comme une organisation à court terme basée sur des engagements spécifiques et contextuels, ce qui permet aux agents de bénéficier de leurs compétences respectives. Dans un contexte économique par exemple, plusieurs compagnies s'allient "virtuellement" pour répondre à des offres nécessitant des compétences variées. La formation de coalition offre plusieurs avantages : : 1) le concept d'engagement ponctuel permet aux agents de réagir de façon opportuniste et de réviser dynamiquement leur intérêts et conséquemment leurs objectifs ; 2) la formation et la dissolution des coalition est contexte-dépendante, ce qui permet aux agents d'adapter leur production ; 3) contrairement aux organisations statiques (*i.e.* prédéfinies), la formation de coalition permet d'appréhender de façon plus flexible les environnements ouverts et dynamiques.
Du point de vue des S.M.A., les recherches sur la formation de coalition essaient de proposer des protocoles d'automatisation qui soient le plus "réaliste" possible. En effet, la recherche de solutions optimales (généralement mesurées en terme de profit total maximum) nécessite des algorithmes NP-complets and à cause de cette complexité, différentes suppositions simplificatrices sont introduites pour générer des solutions calculables en un temps raisonnable. Une des plus fortes suppositions consiste à supposer que les agents sont coopératifs voire altruistes. Dans le cadre de ce papier, contrairement aux approches existantes (cf section 2), nous ne faisons pas de supposition sur les prédispositions aux agents à collaborer. Nous nous contentons de les inciter à la coopération. Les agents sont libres de se comporter comme ils le désirent, et même d'avoir des objectifs égoïstes comme en commerce électronique. Cependant, en général, il n'existe pas d'accord qui satisfasse un ensemble d'agents purement égoïstes qui tentent de collaborer ; on aboutit alors à un blocage, ce qui ne satisfait personne. L'objectif n'est pas d'imposer une solution qui sera toujours contestable, mais de proposer un protocole pour la formation de coalitions qui respecte la liberté des agents, tout en garantissant l'atteinte d'un consensus.
Ces idées sont reprises dans [5].

Cet article est organisé comme suit : la section 2 présente un court état de l'art de la formation de coalition et introduit notre approche, en particulier comparé à celle qui traitent d'agents égoïstes. La section 3 justifie notre démarche qui privilégie la liberté des agents et en présente les grandes lignes. La section 4 formalise les concepts de la résolution du problème du consensus dans le cadre des agents libres, tandis que la section 5 présente les procédures, introduit les concepts nécessaires à la définition du protocole de consensus. La section 6 décrit l'algorithme. La section 7 discute des aspects algorithmiques du protocole comme la terminaison, la complexité et certains aspects d'implémentation. Finallement, la section 8 conclut cet article et présente les perspectives.

2 Positionnement

Plusieurs travaux en S.M.A. se sont intéressés aux organisations dynamiques et en particulier à la formation de coalition. La majorité d'entre eux suppose *a priori* que tous les agents ont la même rationalité voir la même stratégie, ce qui permet de réduire considérablement l'espace de recherche.
Les articles aves lesquels ce papier est comparése rapportent uniquement aux agents égoïstes, car considérer des agents altruistes change radicalement les problème et donc les solutions. Dans [4], le problème est simplifié en un calcul distribué de la meilleure solution pour le système comme un tout. C'est encore un problème difficile, car il est difficile pour un composant d'avoir une vision globale du système. Pour nous, il s'agit d'arriver à concilier les intérêts individuels forcément conflictuels (sinon il n'y aurait pas de problème) pour parvenir, malgré tout, à une solution qui soit acceptée par tous, mais au sens de la légitimité de sa construction.
Le problème de la formation de coalition consiste à trouver une solution qui soit le plus satisfaisant pour l'ensemble des agents. C'est typiquement le cas lorsqu'on cherche à calculer la valeur de Shapley (qui correspond grosso-modo à l'espérance de l'utilité) : on cherche à trouver la configuration la plus satisfaisante en moyenne. Certes, l'intérêt global du S.M.A. est évident mais la méthode imposée laisse peu de place à la liberté individuelle des agents. En effet, soit la configuration est calculée de façon externe, soit elle est calculée par les agents eux-mêmes, mais en leur attribuant un rôle défini qu'ils doivent suivre. C'est le cas par exemple dans [6] qui propose une méthode de calcul distribuée de la valeur de Shapley. Les auteurs imposent donc aux agents que la meilleure solution soit celle qui maximise la valeur de Shapley. Certes, pour garantir une relative satisfaction de l'agent, ils prennent soin de ne pas le forcer à former une coalition qui ne lui apporte pas plus que ce qu'il aurait eu avant. Mais ce déterminisme ne permet pas aux agents de développer leur stratégie propre : si un agent accepte de prendre le risque de créer une coalition à faible rendement dans l'espoir de former plus tard une coalition très productive, cela peut lui être très bénéfique.
Dans [2], Steven Ketchpel a redonné un peu de liberté aux agents en leur permettant d'avoir des estimations différentes du revenu d'une coalition. Le problème essentiel qui se pose alors est la répartition des bénéfices qui ne sont pas connus avec précision au moment de la formation. L'algorithme proposé consiste en une suite d'agrégations d'agents et de coalitions. Chacun négocie son prix de participation en fonction de ses croyances sur la valeur de la coalition. Dans un deuxième temps, les agents décident comment diviser les revenus futurs de la coalition. Pour chaque coalition, les agents élisent un manager qui négocie pour eux. Cette élection consiste en fait en une vente aux enchères, celui qui promet le plus est élu. Ce qui est intéressant ici, c'est que les agents peuvent avoir des conceptions différentes des revenus et parvenir tout de même à un accord. Mais, comme beaucoup d'autres, cet algorithme suppose que tout est monnayable, que tout peut être estimé par une valeur réelle, ce qui a des conséquences sur l'algorithme lui-même, le rendant ainsi très dépendant de ce présupposé. C'est pour éviter ceci que nous avons fait le choix des préférences qui sont indépendantes des critères qui les ont produites laissant aux agents toute la liberté pour élaborer leurs propres stratégies.
Un problème similaire a été étudié par Kenneth Arrow dans [1] : la combinaison des préférences individuelles en une collective. Il montre l'impossibilité de trouver une procédure de vote qui respecte ses cinq axiomes qu'une telle procédure devrait intuitivement respecter sans être dictatoriale. L'algorithme que nous présentons ne conduit pas forcément à des solutions respectant ces axiomes, mais la procédure est la moins dictatoriale possible.
La deuxième critique concerne les dépendances entre les coalitions telles que définies dans [2] : dans une formation itérative, un agent ne peut pas prendre en compte la participation future éventuelle de tel ou tel agent ; en effet, l'intérêt pour une coalition dépend de l'ensemble de ses membres et pas seulement de l'augmentation de sa valeur à l'étape en cours. Dans notre approche, toutes les solutions étant estimée globalement et en parallèle, les agents peuvent pleinement estimer leur intérêt pour chaque solution.
Les agents ont beaucoup de liberté, ce qui nous per-

met de prendre en compte naturellement ce qui apparait souvent comme des contraintes : la dépendance de tâches et celle des coalitions. Le protocole ne s'appuie que sur les préférences, il ne tient pas compte du raisonnement qui les a engendrées et par conséquent, chaque agent est libre d'avoir sa rationalité propre (cf. section 3.1). En effet, une rationalité bonne globalement (de coalition, altruisme) ne l'est pas individuellement et peut inciter les agents (tous ou certains) à ne pas l'appliquer. Ainsi, utiliser ces presupposés pour calculer la meilleure solution dans un contexte d'agents égoïstes n'est plus possible. De même, une stratégie individuelle ne peut être toujours optimale car elle dépend de celles des autres.

3 Notre démarche : privilégier la liberté des agents

3.1 Liberté et rationalité

La difficulté de gestion des coalitions augmente considérablement lorsqu'on considère des agents libres de raisonner et d'agir. La liberté de raisonnement nous empêche de faire des hypothèses sur les connaissances, les motivations, les mécanismes d'inférence ou de décision des agents. La liberté d'agir implique qu'il n'existe pas *a priori* d'action interdite.
Cependant, la liberté est légitimée voire indispensable pour une large classe d'applications. Par exemple, on ne peut pas supposer que tous les agents qui interviennent dans une négociation commerciale soient prêts à suivre à la lettre le protocole imposé, ni qu'ils seraient prêts à dévoiler leurs propres stratégies.
Avec de tels agents, on ne peut pas assurer qu'un consensus soit atteint, car si tous les agents le refusent, aucun protocole ne peut rendre possible l'atteinte d'un consensus. De la même façon, la possibilité de tricher, *i.e.* de ne pas suivre le protocole, doit aussi être prise en compte.
Le protocole que nous présentons est cependant plus efficace (pour trouver une solution) si les agents ont une rationalité économique, mais il fonctionne quand même s'ils ont une rationalité différente. Cette rationalité économique conduit les agents à faire des concessions, car s'ils ne le font pas, il y perdent.

3.2 Conséquences sur les protocoles et les organisations

La liberté des agents a des incidences directes sur la conception des protocoles et l'organisation des agents en coalitions. Pour obtenir des comportements collectifs cohérents, il est nécessaire d'établir des règles. Pour respecter les libertés individuelles, ces règles ne doivent contraindre que des données *observables*. Il n'est pas possible par exemple de contraindre les agents à avoir telle croyance ou telle règle d'inférence. Et on ne peut pas avec certitude trouver les croyance d'un agent en observant son comportement (ce qui est visible). Le protocole proposé ne peut donc reposer que sur les actes observables, car si l'on veut que les agents le respectent, il faut qu'en cas de fraude, ils soient sanctionnés. La sanction est alors le moteur d'incitation à suivre les règles données.
Dans un tel contexte, un protocole doit être : 1) universel, car on ne peut rien supposer sur les systèmes cognitifs des agents ; on respecte ainsi leur liberté ; 2) égalitaire, pour ne pas favoriser des agents sur des aspects extérieurs à l'objet de la négociation, bien qu'un agent disposant de plus de ressources (processeur, mémoire, information, processus cognitifs) sera forcément avantagé ; 3) distribué, car la centralisation amène les problèmes habituels (surcharge du réseau et de l'agent central, faiblesse aux pannes,...) en plus de la possibilité accrue de triche pour l'agent central alors en situation de pouvoir ; bien que l'on puisse faire la supposition d'un agent impartial, son intégrité pourrait toujours être mise en cause (influences externes, corruption), ce qui nuirait à la légitimité de la solution.
La liberté des agents modifie aussi la perception qu'on peut avoir des organisations. Celles-ci sont en effet parfois considérées comme des entités à part entière, dépendantes certes de leurs membres, mais possédant néanmoins des attributs propres pouvant aller jusqu'à une certaine autonomie. Dans le contexte de cet article, une organisation est un ensemble d'agents liés contractuellement et ponctuellement, mais qui disposent toujours de toute leur liberté. Chaque agent y joue un certain rôle, mais pour les mêmes raisons, on n'identifiera pas rôle et comportement. Un rôle est un ensemble de droits et de devoirs, codifiés dans un règlement, dans l'espoir d'obtenir un comportement d'ensemble cohérent. Les droits sont représentés par l'ensemble des actions possibles à chaque étape et un devoir par l'ensemble des actions attendues par les autres agents. Mais tout agent garde sa liberté et peut donc respecter ou non le règlement, et il est donc nécessaire de prendre en compte ces possibilités.

3.3 Objectif : résolution du problème de consensus pour la répartition des tâches

Nous nous plaçons dans le cadre de la résolution de tâches par des agents : un système reçoit un ensemble de tâches subdivisées en sous-tâches et propose ces sous-tâches à des agents inscrits. Ceux-ci tentent alors de se répartir l'ensemble des sous-tâches, chacun ne pouvant en exécuter qu'une partie. Une tâche est exécutée quand toutes ses sous-tâches le sont. Pour simplifier, nous avons choisi de fixer la décomposition, car celle-ci a des conséquences sur les revenus des agents et pourrait aussi constituer un enjeu de négociation. Nous supposons de plus qu'il n'y a pas de contrainte sur l'ordre des sous-tâches.
L'objectif de l'algorithme proposé est de permettre à des agents libres ayant des intérêts potentiellement incompatibles de parvenir à trouver un consensus sur la répartition des sous-tâches, mais du point de vue des agents. Il ne s'agit pas d'imposer aux agents une solution ni même une procédure de calcul de la solution qui serait justifiée par le concepteur, mais de laisser chacun agir selon sa propre stratégie cohérente avec sa rationalité.
La motivation des agents à la réalisation des sous-tâches repose sur les gains associés et les stratégies propres aux agents qui leur permettent d'établir des préférences sur l'ensemble des solutions possibles. Ainsi, le protocole que nous proposons ne tient compte que des préférences choisies.

4 Concepts

4.1 Concepts du problème

Nous allons maintenant présenter les concepts du problème et montrer leur sens à travers un exemple : des compagnies aériennes choisissent de coopérer pour proposer à leurs clients un système unifié de réservation. Jusqu'ici, un vol pouvait nécessiter plusieurs réservations dans différentes compagnies. Pour lui simplifier la tâches, des compagnies couvrant l'ensemble du globe se sont donc alliées : le choix des compagnies pour les étapes est maintenant transparent pour le passager. Le problème pour les compagnies est la répartition des étapes entre elles, car il y a concurrence sur certains trajet.
Définition 1 : Un Problème de Formation de Coalition (PFC)
Un PFC est défini par $\langle \mathcal{A}, \mathcal{T}, \mathcal{S}, \mathcal{C}, \mathcal{G} \rangle$, avec :
$\mathcal{A}$: l'ensemble des agents candidats à l'exécution des sous-tâches ;
$\mathcal{T}$: l'ensemble des tâches qui doivent être accomplies ;
$\mathcal{S}$: l'ensemble des sous-tâches à exécuter ;
$\mathcal{C}$: l'ensemble des compétences nécessaires à l'exécution des sous-tâches ;
$\mathcal{G}$: l'ensemble des revenus.
Formellement :
Définition 2 : Agent
Un agent $a \in \mathcal{A}$ possède un certain enesemble de compétences ; un agent $a \in \mathcal{A}$ est donc partiellement défini par : $a = \langle C, \ldots \rangle, C \subset \mathcal{C}$. Cette définition sera complétée ultérieurement par d'autre éléments, comme la stratégie (voir (5.3, p. 6)).

Exemple Dans notre exemple, $\mathcal{A}$ désigne des compagnies aériennes :
$\mathcal{A} = \{EUropeanAL, AMericanAL, WOrldAL, USAL, AFricaAL, FRanceAL, BUsinessAL\}$

Définition 3 : Tâche
Une tâche $t \in \mathcal{T}$ est seulement définie par l'ensemble de sous-tâches qu'elle contient : $t = \langle S \rangle, S \subset \mathcal{S}$.

Exemple Une tâche est un vol entre deux villes, avec des escales : $\mathcal{T}$ = {New York-MAdrid (via PAris and LYon), Los Angeles-MOscow (via New York and PAris) and BErlin-JOhannesburg (via PAris)}.

Définition 4 : Sous-tâche
Une sous-tâche $s \in \mathcal{S}$ est définie par $s = \langle C, g \rangle, C \subset \mathcal{C}, g \in \mathcal{G}$, où c est l'ensemble des compétences qu'un agent doit posséder pour pouvoir exécuter la sous-tâche, et g est le profit associé. Ce profit sera utilisé par les agents pour calculer leurs préférences.

Exemple $\mathcal{S}$ = {New_York→Paris, Lyon→Madrid, Paris→Moscow, ...}. Nous pouvons maintenant définir la tâche $NY - M$ par $NY - M$ = $\langle$ {NY→P, P→L, L→M}, ...$\rangle$.

Définition 5 : Competence
Une competence $c \in \mathcal{C}$ est un item unique qui représente ce qui est nécessaire à un agent pour exécuter une sous-tâche.

Exemple Chaque vol nécessite des compétences : autorisations pour effectuer un vol national, (autFrance, autUSA,...), capacité en passagers (Weak, Middle, Large), rayon d'action (Very-Short-Range, Short-Range, Medium-Range, Long-Range).
$\mathcal{C} = \{autX, WC, MC, LC, VSR, SR, MR, LR\}$ EUA = $\langle \{autFR, autEU, autRU, WC, MC, SR, MR\} \rangle$

Définition 6 : Profit
Un profit $g \in \mathcal{G}$ est remplacé ici par un revenu, mais seulement pour simplifier les calculs des agents : $\mathcal{G} = [0, ProfitMax]$.

Exemple Nous choisissons $\mathcal{G} = [0, 10000]$ et par example, on a $NY-M = \langle \{NY \rightarrow P, P \rightarrow L, L \rightarrow M\}, 8000 \rangle$.

4.2 Les concepts de la résolution

Pour résoudre le problème, les agents s'échangent leurs préférences à propos des solutions. Si aucun consensus n'est atteint, ils forment des alliances. Les agents doivent donc représenter les solutions, les préférences, les alliances et les coalitions.
Pour atteindre un consensus, les agents doivent modifier leur opinion (ici leurs préférences). Parmi toutes les possibilités imaginées (pression, corruption,...), nous avons choisi l'échange de préférences. Primo elles nous apparaissaient plus rationnelles, conduisant ainsi à une solution plus légitime (à comparer à un tirage aléatoire par exemple). Secundo, elles peuvent être utilisées par des agents hétérogènes, là où des messages utilisant des langages de haut niveau nécessitent des traitement symboliques complexes.

Définition 7 : Solution
Une solution est une assignation à chaque sous-tâche d'un agent qui soit capable de l'exécuter. Une solution $\sigma \in \Sigma$ est une application $\mathcal{S} \rightarrow \mathcal{A}$ telle que $\forall s \in \mathcal{S},\ a = \sigma(s) \Rightarrow s.C \subset a.C$.

Exemple σ_{15} = [NY→PA$_2$↪WOA, L→M↪BUA, P→MO↪EUA, BE→P↪FRA, LA→NY↪USA, P→JO↪AFA, NY→PA$_1$↪WOA, P→L↪FRA].

Définition 8 : Préférence
Une préférence est représentée par des distances (pas au sens mathématique) $\delta \in \Delta$ entre les solutions, où $\delta : \Sigma \times \Sigma \rightarrow [-1, 1]$ est une application antisymétrique. Ainsi, $\delta(\sigma_1, \sigma_2) = d$ est interprétée par "σ_2 est préférée à σ_1 avec une distance d si $d \geq 0$ et σ_1 est préférée à σ_2 avec une distance $-d$ si $d < 0$". Une distance nulle signifie que les solutions sont indifférentes.

Exemple Soit $S_1 = \{\sigma_0, \sigma_2, \sigma_4, \sigma_6, \sigma_8, \sigma_{10}, \sigma_{12}, \sigma_{14}\}$ l'ensemble des solutions qui rapportent une certaine somme et $S_2 = \{\sigma_1, \sigma_3, \sigma_5, \sigma_7, \sigma_9, \sigma_{11}, \sigma_{13}, \sigma_{15}\}$ l'ensemble de celles qui ne rapportent rien. $\delta(\sigma, \sigma') = 0$ si σ et σ' sont dans le même ensemble, $\delta(\sigma, \sigma') = 1$ si $\sigma \in S_1$ et $\sigma' \in S_2$, et $\delta(\sigma, \sigma') = -1$ si $\sigma \in S_2$ et $\sigma' \in S_1$.

Définition 9 : Histoire
Une histoire $h \in H$ est une suite de vues v_t, où chaque vue $v_t \in V$ est une application $\mathcal{A} \rightarrow \Delta$ et t le numéro du tour. Une histoire $h = (v_t)_{1 \leq t \leq T}$ représente toutes les préférences echangée du premier au dernier tour.
Une alliance est un ensemble d'agents, mais se comporte du point de vue de l'extérieur comme un seul agent : un des membres a le rôle de représentant et communique aux agents non membres de la coalition la préférence unifiée.

Définition 10 : Alliance
Une alliance $\lambda \in \Lambda$ est définie par $\lambda = \langle A, a_{rep} \rangle$, où $A \subset \mathcal{A}$ et $a_{rep} \in \mathcal{A}$ un membre de l'alliance avec un rôle spécial, avec la contrainte qu'un agent ne peut appartenir qu'à une seule alliance. L'agent a_{rep} peut être inconnu ; dans ce cas, il est noté «?».

Exemple $\lambda = \langle \{USA, EUA, BUA\}, EUA \rangle$.

Définition 11 : Coalition
Une coalition $\Xi(\sigma, t) \subset \mathcal{A}$ associée à la tâche $t \in \mathcal{T}$ dans la solution $\sigma \in \Sigma$ est définie par : $\Xi(\sigma, t) = \{a \in \mathcal{A} / \exists s \in \mathcal{S}, s \in t.S, \sigma(s) \ni a\} = \bigcup_{s \in t.S} \sigma(s)$. Une coalition contient tous les agents qui prennent part à une tâche.

5 Procédures

5.1 Procédures communes

Ces procédures sont utilisées par tous les agents.

- Fusion d'alliances
- Calcul de préférences d'une alliance

Définition 12 : Fusion d'Alliances
Une opération de fusion d'alliances $fusion : 2^{\Lambda} \rightarrow \Lambda$ est définie par $fusion(\{\lambda_1, \ldots, \lambda_n\}) = \langle A, a_r ep \rangle$, $A = \bigcup_{i \in [1,n]} \lambda_i.A$ et $a_{rep} = election(A)$, où les λ_i sont les alliances qui désirent former une nouvelle alliance et *election* une procédure d'élection d'un agent parmi les membres de l'alliance.
Cette opération peut être appliquée à des agents en convertissant un agent a en une alliance $\langle \{a\}, a \rangle$.

5.2 Procédures au niveau des membres d'une alliance

Puisqu'une alliance doit tenir le même rôle qu'un agent, il doit effectuer des calculs similaires. Elle doit intégrer les résultats des calculs des agents pour produire une seule donnée (préférence par exemple). Nous avons deux possibilités : soit on définit une fonction qui prend comme paramètres les résultats calculés par les agents ; soit on définit une fonction qui utilise les critères qui ont permis aux agents de calculer leurs résultats.
Dans cette seconde solution, tous les agents doivent avoir les mêmes critères de base, ce qui restreint les possibilités et limite les évolutions. Dans les deux cas, une fonction est imposée, mais le choix de celle-ci

a plus d'influence sur le résultat finale dans le premier cas que dans le second. En fait, il est difficile de trouver pour la première solution une fonction qui se comporte en accord avec notre intuition (comme suggéré par Kenneth Arrow dans [1]). Nous avons finalement choisi la deuxième solution.

Définition 13 **: Calcul de préférence des alliances**
Le calcul de préférence d'une alliance $CPAll$ est une application $\Lambda \rightarrow \Delta$. Celle-ci n'est connue que par les membres de l'alliance, les autres ne connaissent que le résultat du calcul. Pour simplifier, nous utilisons la même application pour tous les agents, mais l'algorithme n'en tient pas compte : une alliance est comme une boîte noire pour les non-membres.

Exemple Soit $\lambda \in \Lambda$ une alliance, $\lambda = \{A, a_rep\}$, $A \subset \mathcal{A}$. $CPAll(\lambda) = \delta$, où δ est définie par : $\forall(\sigma_1, \sigma_2) \in \Sigma^2$, $\delta(\sigma_1, \sigma_2) = \sum_{a \in A} a.\delta(\sigma_1, \sigma_2)$. Cet exemple n'utilise que les préférences des membres pour calculer celle de l'alliance.

5.3 Procédures des agents

Chaque agent a sa propre stratégie qui peut être appliquée en utilisant les procédures internes :

- Calcul des Préférences Indépendantes : calcul des premières préférences sans connaître celles des autres.
- Calcul des Préférences Dépendantes : calcul des préférences des tours suivants.
- Critère de Proposition de Déblocage : critère utilisé pour décider quand proposer de passer en mode déblocage.
- Critère d'Acceptation de Déblocage : critère qui décide d'accepter ou pas le déblocage.
- Critère de Proposition de Formation d'Alliance : renvoie une liste des agents avec lesquels former une alliance.
- Critère d'Acceptation de Formation d'Alliance : permet de répondre aux propositions de formation d'alliance.

Puisque ces fonctions dépendent des stratégies, nous ne pouvons fournir que des définitions et des exemples, mais pas de formulation générale.

Définition 14 **: Calcul des Préférences Indépendantes**
Un $CPIndep$ est un élément δ de Δ.

Exemple Soit $\delta = CPIndep, \forall(\sigma_1, \sigma_2) \in \Sigma^2), \delta(\sigma_1, \sigma_2) = profit(\sigma_2) - profit(\sigma_1)$. δ est une application antisymétrique.

Définition 15 **: Calcul des Préférences Dépendantes**
Un $CPDep$ est une fonction $H \rightarrow \Delta$, $h \mapsto \delta$.

Exemple Soit $\delta = CPDep(h)$, $h = (v_t)_t$. $\forall(\sigma_1, \sigma_2) \in \Sigma^2$, $\delta(\sigma_1, \sigma_2) = [\sum_{a \in \mathcal{A}}(v_T(a))(\sigma_1, \sigma_2)]/|\mathcal{A}|$. δ est une application antisymétrique.

Définition 16 **: Critère de Proposition de Déblocage**
Un $CPDebloc$ est une application $H \rightarrow \{False, True\}$.

Exemple On considère ici qu'il y a blocage quand une vue (ensemble des préférences) se reproduit à l'identique. Cependant, pour diminuer la complexité de calcul, seules les boucles de longueur 3 sont détectées. Soit $h = (v_t)_{1 \leq t \leq T}$. $CPDebloc(h) = False$ if $T \leq 2$ et $CPDebloc(h) = (v_T = v_{T-1}) \vee (v_{T-1} = v_{T-2}) \vee (v_T = v_{T-2})$ sinon.

Définition 17 **: Critère d'Acceptation de Déblocage**
A $CADebloc$ est an application $H \rightarrow \{False, True\}$.

Exemple $CADebloc = CPDebloc$

Définition 18 **: Critère de Proposition de Formation d'Alliance**
Un $AFPC$ est une application $h \mapsto (\lambda_1, \ldots, \lambda_n)$, où $AFPC(h) = \emptyset$ est permis et est interprété par «l'agent ne veut pas former une alliance».

Exemple Soit $d : \Delta \times \Delta \rightarrow \mathbb{R}$ une distance entre les préférences des agents, par exemple : $\forall(\delta_1, \delta_2) \in \Delta^2$, $d(\delta_1, \delta_2) = \sum_{(\sigma_1, \sigma_2) \in \Sigma^2} |\delta_1(\sigma_1, \sigma_2) - \delta_2(\sigma_1, \sigma_2)|$. Pour un agent a, $AFPC(h)$ est alors l'ensemble des agents dont les préférences sont assez proche de lui, *i.e.* dont la distance est inférieure à un certain seuil.

Définition 19 **: Critère d'Acceptation de Formation d'Alliance**
Un $CAFAll$ est une application $H, A \rightarrow \{False, True\}$.

Exemple Nous pouvons utiliser la même application que ci-dessus, mais enutilisant un seuil plus élevé, ce qui revient à être moins exigeant.

5.4 Diffusion Parallèle

Dans [6], chaque agent diffusent son information cryptées, puis lorsqu'il a reçu celles des autres, il en diffuse la clé. Mais cette technique n'empêche pas certaines situations de se produire : dans un système à trois agents a b et c, a envoie ses préférences à b et c, b à a et c, mais c uniquement à a ; comme a a reçu

toutes les préférences, il envoie sa clé à b et c, ce qui permet à c d'envoyer à b une autre préférence.
Les auteurs supposent que les agents ne trichent pas trop, ce qui est une hypothèse forte. Nous pouvons lever cette restriction en améliorant ce mécanisme et empêcher ainsi toute fraude : les agents envoient leurs préférences cryptées avec une clé privée, puis lorsque chacun à reçu toutes les préférences, il diffuse un accusé de réception. Ce n'est que lorsqu'un agent a reçu tous les accusés qu'il diffuse sa clé. Cette technique ne fait pas que déplacer le problème, elle le résout. Si on exécute le même scénario ici, l'agent qui a triché ne pourra plus changer son message crypté. Il pourrait envoyer une fausse clé qui ne fonctionne pas, mais la fraude serait visible et la suspicion pèserait alors sur lui.

Algorithm 1 Diffusion parallèle des données θ à $\mathcal{A}$

for all $a \in \mathcal{A}$ **do**
  $\theta^* \leftarrow Encrypt(\theta, key)$
  $\mathcal{B} \leftarrow \mathcal{A} \setminus \{a\}$
  a.**broadcast**(θ^*,$\mathcal{B}$)
  a.**receipt**(θ^*,$\mathcal{B}$)
  a.**broadcast**(Ack,$\mathcal{B}$)
  a.**receipt**(Ack,$\mathcal{B}$)
  a.**broadcast**(key,$\mathcal{B}$)
end for

5.5 Rôles des agents

Chaque agent peut jouer plusieurs rôles dans le système :

- Organisateur :
  s'occupe des inscriptions
  gère les tours
  envoi les données du problème aux candidats (tâches, sous-tâches, compétences nécessaires)
  informe les candidats des compétences des autres agents
- Candidat :
  cherche des sous-tâches à exécuter
  reçoit et envoie ses préférences à son alliance ou au système
  propose/accepte le passage en mode déblocage
  propose/accepte la formation de coalitions
- Surveillant : vérifie le repect du protocole par les agents (*i.e.* qu'ils envoient bien les mêmes préférences à tous)
- Représentant :
  sert d'interface de communication entre les membres de l'alliance qu'il représente et les autres membres du système
  intègre les préférences des membres de l'alliance pour calculer la préférence de l'alliance

6 Algorithme

Nous ne décrirons que l'algorithme du candidat, car c'est le rôle principal. Chaque agent a une liste d'interlocuteurs $InterList \subset \mathcal{A}$ initialisée avec la liste des candidats. L'algorithme suivant est utilisé par chaque agent a_i de façon distribuée.

Algorithm 2 Mode déblocage

broadcast("Propose de former une alliance", $AFPC(h)$)
for a tel que **receive**("Propose de former une alliance",a) **do**
  if $a \in CAFAll(h)$ **then**
    {d'après $CAFAll$, la proposition de a est acceptée}
    send("Accepte de former une alliance",a)
  end if
end for
if Pas d'alliance formée **then**
  Le système choisit les entités aux préférences les plus proches
  Le système les force à former une alliance
end if
Mise à jour de $InterList$
Return.

7 Analyse de l'algorithme

7.1 Terminaison

Sans hypothèse sur le critère de passage en mode déblocage, nous ne sommes pas capable de garantir que le processus termine. Cependant, si nous supposons que le critère consiste à détecter qu'il n'y a pas de boucle (deux fois la même situation se présente), nous pouvons prouver qu'il termine.
Définition 20 **: Une boucle dans une histoire** Nous disons qu'une histoire $h = (v_t)_{1 \leq t \leq T}$ contient une boucle si $\exists (\tau_1, \tau_2) \in [[1, T]]^2$, $\tau_1 \neq \tau_2$ tel que $v_{\tau_1} = v_{\tau_2}$. **Définition** 21 **: Un** CFP **détecte les boucles**

Algorithm 3 Principal

$IndPref \leftarrow CPIndep$ {Calcul des préférences indépendantes}
$h \leftarrow ParallelDiff(IndPref, InterList)$
while le consensus n'est pas atteint **do**
  if $CPDebloc(h)$ **then**
    {d'après ce critère}
    send("proposition de passage en mode déblocage",*)
  end if
  if receive("proposition de passage en mode déblocage",?) **then**
    if $CADebloc$ **then**
      {d'après ce critère}
      send("proposition de passage en mode déblocage",*)
    end if
  end if
  if $\forall a \in InterList$, **receive**("proposition de passage en mode déblocage",a) **then**
    {passage en mode déblocage accepté}
    appel **mode déblocage**
  end if
  $DepPref \leftarrow CPDep(h)$ {Calcul des préférences dépendantes}
  $h \leftarrow ParallelDiff(DepPref, InterList)$
end while

Un $CFP\ \langle \mathcal{A}, \mathcal{T}, \mathcal{S}, \mathcal{C}, \mathcal{G} \rangle$ détecte les boucles si (h contient une boucle $\Rightarrow$ ($\exists a_0 \in A$ tel que $a_0.CPDebloc(h) = True \wedge \forall a \in A, a.CADebloc(h) = True$)). En d'autres termes, un CFP détecte les boucles si en présence d'une boucle, au moins un agent la détecte et si tous accepte de passer en mode déblocage.

Theorem 7.1 *Si un CFP détecte les boucles, alors le processus termine.*

Preuve Comme le nombre n d'agents et le nombre k de solutions sont finis, il n'y a que $n.k!$ vues possibles. Après $n.k!+1$ tours, l'histoire contiendra nécessairement deux vues identiques, *i.e.* une boucle. Le système passe alors en mode déblocage, ce qui conduit à la formation d'au moins une alliance. Quelque soit le critère de consensus retenu, il est raisonable de supposer que si tous les agents sont d'accord, le consensus est atteint. Dans le pire des cas, on aboutit à la formation de la Grande Alliance qui contient tous les agents, ce qui nécessite $(n.k!+1)(n-1)$ tours. Il y a alors consensus. ∎

7.2 Complexité

La complexité dépend notamment du nombre de solutions possibles qui est directement lié aux données du problème. Supposons que notre système contienne n agents. Calculer le nombre moyen d'agents capables de résoudre une sous-tâche n'a pas beaucoup de sens, car cela ne tient compte ni des compétences très répandues, ni surtout des compétences qui vont par lot. Supposons donc que chaque agent puisse traiter $1/m$ des tâches, une tâche a donc en moyenne n/m agents capables de la résoudre, ce qui donne alors $k = (n/m)^s$ solutions. Lorsque le nombre d'agents croît, la quantité de solutions à traiter devient rédhibitoire ; il est alors nécessaire de limiter l'espace à considérer. On peut par exemple restreindre l'ensemble des solutions aux trois solutions préférées de chaque agent, avec éventuellement des retraits et des ajouts en cours d'avancement. Mais cela amène de nouveaux problèmes : une solution mal classée par tous peut s'avérer être une bonne candidate pour un consensus. Dans le cas le plus général, notre algorithme ne permet pas de changer de classe de complexité, mais un calcul en moyenne montre que plus les agents ont des compétences qui se recouvrent, plus il y a compétition et plus le consensus est difficile à atteindre. Cependant, il est raisonnable de supposer qu'un système bien conçu répartit les compétences de façon uniforme. Pour une bonne complexité, il faut que le rapport n/m soit le plus petit possible.

7.3 Implémentation

L'objectif est de vérifier l'implantabilité de l'algorithme et de tester les différents paramètres intervenant dans la formation de coalition : stratégies individuelles des agents (calcul des préférences, critère de déblocage, formation d'alliances) et des paramètres globaux (critère de consensus, choix des alliances en déblocage forcé).
Nous avons choisi le langage Java car il permettait *via* les threads de simuler naturellement le caractère distribué des S.M.A. Ainsi, chaque envoi de message génère un thread, ce qui permet de simuler des communications asynchrones.
Chaque agent n'a accès qu'à ses données locales et ne peut donc échanger de l'information que *via* les messages, ce qui garantit son autonomie. Chaque rôle est implanté sous la forme d'un thread chargé de réagir aux messages reçus ou d'agir de son propre chef. Lorsqu'un agent reçoit un message, il le dirige vers le thread chargé de gérer les messages destinés au rôle.

8 Conclusion et perspectives

Dans une perspective de déploiement de S.M.A. dans un contexte économique, il est nécessaire de

considérer des agents hétérogènes, égoïstes et libres dans leur raisonnement et leur comportement. Pour parvenir, dans ce cadre, à former des coalitions nous avons proposé un protocole reposant sur un échange de préférences basées sur des critères quantitatifs et qualitatifs puisque dépendantes des stratégies propres aux agents. De plus, nous avons mis en place des procédures de déblocage grâce au concept souple d'alliance pour éviter la paralysie du système.
Nous avons montré qu'un tel protocole est ouvert, distribué et égalitaire. De ce fait, il ouvre de nombreuses perspectives. Dans un premier temps, on cherchera à améliorer la complexité de calcul en introduisant des heuristiques. Nous pourrons ensuite tester logiciellement les différents paramètres et choisir ainsi les critères et stratégies les plus intéressants.
Comme dans [3], il serait également intéressant de permettre aux agents de garder une trace des alliances qui ont été bénéfiques, afin d'accélerer la convergence vers un consensus. Construire un modèle des autres permettra d'intégrer la notion de confiance et limitera la suspicion.

Références

[1] Kenneth Arrow. *The Origins of the Impossibility Theorem*, chapter 1, pages 1–4. Elsevier Science Publishers B. V., Amsterdam, 1991.

[2] Steven Ketchpel. Forming coalition in the face of uncertain rewards. In *Proceedings of the 12th National Conference on Artificial Intelligence. Volume 1*, pages 414–419, Menlo Park, CA, USA, July 31–August 4 1994. American Association for Artificial Intelligence, AAAI Press.

[3] Jorge Anacleto Louçã, Helder Coelho, and Suzanne Pinson. Forming coalition in task oriented multi-agent systems. In *Proceedings Of IBERAMIA 1996*. IBERAMIA, October 1996.

[4] Onn Shehory and Sarit Kraus. Formation of overlapping coalitions for precedence-ordered task-execution among autonmous agents. In Victor Lesser, editor, *Proceedings of the First International Conference on Multi–Agent Systems*, pages 330–337. International Conference on Multi-Agent Systems, MIT Press, 1995.

[5] Guillaume Vauvert and Amal El Fallah Seghrouchni. Coalition formation for egoistic agents. In *Proceedings of International ICSC Symposium on Multi-Agents and Mobile Agents in Virtual Organizations and E-Commerce (MAMA'2000)*. ICSC, December 2000. À paraître.

[6] Gilad Zlotkin and Jeffrey S. Rosenschein. Coalition, cryptography, and stability : Mechanisms for coalition formation in task oriented domains. In *Proceedings of the 12th National Conference on Artificial Intelligence. Volume 1*, Distributed AI, pages 432–437, Menlo Park, CA, USA, July 31–August 4 1994. American Association for Artificial Intelligence, AAAI Press.