RAIRO-Oper. Res. 43 (2009) 437–462
DOI: 10.1051/ro/2009028

RAIRO Operations Research
www.rairo-ro.org

GTES : UNE MÉTHODE DE SIMULATION PAR JEUX ET APPRENTISSAGE POUR L'ANALYSE DES SYSTÈMES D'ACTEURS

Y. Caseau[1]

Résumé. Cet article décrit une approche de la modélisation d'un système d'acteurs, particulièrement adaptée à la modélisation des entreprises, fondée sur la théorie des jeux [11] et sur l'optimisation par apprentissage du comportement de ces acteurs. Cette méthode repose sur la combinaison de trois techniques : la simulation par échantillonnage (Monte-Carlo), la théorie des jeux pour ce qui concerne la recherche d'équilibre entre les stratégies, et les méthodes heuristiques d'optimisation locale, en particulier les algorithmes génétiques. Cette combinaison n'est pas originale en soi, même si elle est rarement utilisée avec toute la puissance d'expression conjointe de ces techniques. La contribution de cet article est double : d'une part nous proposons un modèle qui permet de structurer de façon systématique cette collaboration entre différentes techniques et, d'autre part, nous utilisons la technique des algorithmes génétiques pour enrichir la recherche des équilibres de Nash sous forme de points fixes. Il s'agit d'une méthode de simulation, qui n'est pas destinée à la résolution de problèmes, mais à la validation et l'étude des propriétés d'un modèle associé à un problème particulier.

Mots Clés. Simulation, apprentissage, théorie des jeux, algorithmes génétiques, organisation d'entreprise.

Received January 30, 2008. Accepted June 17, 2009.

[1] Bouygues Telecom, 20 Quai du Point du Jour, Boulogne-Billancourt Telecom, France ; yves@caseau.com

Article published by EDP Sciences

Abstract. This paper proposes an approach towards modeling an actor system, especially suited to describe a company's organization, based on game theory [11] and learning-based (evolutionary) local optimization. This method relies on the combination of three techniques: sampling for simulation (Monte-Carlo), game theory as far as the search for equilibrium is concerned and heuristic local search methods, such as genetic algorithms. This combination is not original as such, although it is rarely used with the full combined expressive power of this array of techniques. Our contribution with this paper is twofold. On the one hand we propose a model which is a natural framework for the collaboration between these three techniques. On the other hand, we use genetic algorithms to extend the search of Nash equilibrium, obtained as fixed-points of an iterative transformation. This remains a simulation tool, not intended to solve problems but to validate a given model and to study its properties.

Keywords. Simulation, learning, game theory, genetic algorithms, enterprise organization.

Classification Mathématique. 91-08, 91A80, 91A90.

1. Introduction

Le point de départ de la démarche GTES (*Game-Theoretical Evolutionary Simulation*) est le suivant : supposons que nous ayons construit un modèle, qui nous semble valide dans sa structure, mais pour lequel il soit difficile d'obtenir les paramètres (par exemple, les coefficients des équations du modèle). Nous allons distinguer trois types de paramètres :

- les paramètres qui représentent des sensibilités, des élasticités, qui demanderaient des études poussées. Par exemple, il est raisonnable de modéliser l'appétence du public à un produit donné avec une « courbe en S », il est par contre difficile, sauf à disposer d'études marketing, de connaître précisément les valeurs qui caractérisent cette courbe en S. De plus, même si l'on dispose d'études, les incertitudes sur ces courbes qui sont sensées représenter le comportement du consommateur sont importantes. Il est donc logique de travailler avec des intervalles de valeurs qui décrivent des « familles de courbes ».
- les paramètres qui représentent la stratégie des acteurs. Nous ne connaissons pas forcément la stratégie des acteurs, mais l'intérêt de la simulation est précisément d'étudier les effets combinés des différentes stratégies, souvent sous la forme de scénarios.

- Les paramètres qui sont sous l'arbitrage de chaque acteur, mais qui sont dominés par les objectifs stratégiques. Autrement dit, il s'agit de paramètres que chaque acteur peut adapter en fonction de ses objectifs. Si nous laissions ces paramètres comme des éléments de stratégie, nous aurions un espace de combinaison très grand et dont une partie n'aurait pas de sens.

La méthode de simulation GTES repose sur deux principes. D'une part, nous allons séparer la « stratégie », qui est considérée comme fixée et sur laquelle nous ferons des « études de cas », de la tactique qui va être optimisée de façon automatique. Ce premier principe est une forme de simplification, puisqu'il consiste à réduire l'espace des paramètres en profitant des dépendances (puisque la « tactique » est dirigée par la stratégie). D'autre part, nous allons traiter l'incertitude sur les paramètres « externes » par un échantillonnage, avec une approche de type Monte-Carlo [11]. Comme nous ne disposons pas d'information sur les lois de distribution des paramètres à l'intérieur des intervalles de confiance, le résultat statistique obtenu par échantillonnage, représenté par une moyenne et un écart type, ne doit pas être interprété autrement que comme un ordre de grandeur approximatif. Mais cette approximation est cohérente avec l'objectif de cette méthode.

En effet, l'objectif de la méthode GTES est de faire émerger des propriétés d'un système complexe, incertain et partiellement inconnu. C'est en quelque sorte un outil pour explorer une modélisation d'un problème économique, pour « se faire une idée » avant de passer à une étude plus formelle ou plus intense d'un point de vue computationnel. Puisque l'objectif est une forme de « prototypage » de modélisation, nous appliquerons un principe de simplification (une forme du « rasoir d'Occam ») en cherchant à minimiser l'espace des décisions à explorer.

Une fois que l'on a séparé les deux familles de paramètres correspondants aux stratégies des acteurs et aux paramètres externes, l'optimisation des « tactiques » relève de la théorie des jeux [9,17], et plus précisément de la recherche d'un équilibre dans un jeu répété. Cette optimisation présente une double difficulté : caractériser et trouver les équilibres (des solutions d'une optimisation multicritère) d'une part, et d'autre part se convaincre que ces équilibres sont représentatifs des comportements des acteurs du monde réel que l'on cherche à représenter. Nous verrons que la méthode GTES propose une solution pragmatique qui se révèle un bon compromis sur ces deux aspects.

Cet article est organisé comme suit. La section suivante présente la méthode GTES de façon intuitive et informelle. Nous allons expliciter la méthode de décomposition que nous avons esquissée dans cette introduction. Nous définissons le problème de jeu non-coopératif et la notion d'équilibre entre acteurs. La section 3 propose un modèle formel pour décrire le problème général, sa segmentation en sous-problèmes et l'utilisation de techniques d'optimisation pour construire des équilibres de Nash par itération et recherche de point fixe. La section suivante traite de l'implémentation de cette méthode au moyen d'algorithmes. Nous montrons comment combiner différentes techniques d'optimisation locale [1], et en particulier des algorithmes « génétiques » simplifiés. Nous revenons également sur certaines considérations d'implémentation dues à la nature intensive des calculs qui

sont nécessaires pour traiter des problèmes riches. La dernière section illustre l'application de cette méthode à deux problèmes particuliers : la compétition d'entreprises sur un marché mature des télécommunications, et l'étude des stratégies de distribution dans un réseau hybride propriétaire et compétitif.

2. L'approche GTES : Jeux et Apprentissage

2.1. Motivation

Ce travail s'inscrit dans une double lignée de travaux « classiques » à l'intersection de la modélisation économique, de la théorie des jeux [10] et de l'application des algorithmes « évolutionnaires » (algorithmes génétiques, recuit-simulé, recherche stochastique et taboue [5], etc.). En particulier, cette méthode est fortement influencée par les travaux de R. Axelrod. Dans [2], R. Axelrod utilise des algorithmes génétiques pour optimiser des stratégies en simulant un jeu répété du dilemme du prisonnier. Cet article est fondamental car il contient à la fois des résultats d'un protocole expérimental (un concours d'algorithmes pour implémenter une stratégie gagnante) et des résultats de simulation numérique (dans laquelle les combinaisons entre différentes stratégies décrites par des paramètres sont optimisées par un algorithme génétique), qui coïncident (sur la supériorité globale de la stratégie TIT-for-TAT).

Plus généralement ce travail s'inscrit également dans la tradition de la modélisation et la simulation économique avec des systèmes multi-agents décrit par des algorithmes évolutionnaires [13]. Par exemple, le chapitre 7 du livre de Nelson et Winter contient un exemple du type de modèle que nous souhaitons simuler, et que les auteurs proposent d'étudier avec des chaînes de Markov. On peut également faire un parallèle avec les approches de « dynamique des systèmes », ce que nous ferons dans la dernière section 5.3.

La première motivation pour décrire cette approche (et la formaliser, ce qui sera fait dans la section suivante) est sa très large applicabilité. Il s'agit d'une méthode qui est née de l'observation des similarités entre différents programmes développés de façon *ad hoc* pour étudier une problématique de coopération (dans un cadre de simulation multi-agent [7]). Après avoir réalisé un certain nombre de ces expériences de simulation, nous avons observé que, d'une part, les résultats étaient intéressants malgré l'incertitude constatée au départ et, d'autre part, que la logique de simulation était identique pour des problèmes très différents. Une partie de ces expériences seront décrites dans la dernière section.

La seconde motivation est que nous avons été amenés à construire une collaboration forte entre l'aspect d'optimisation locale (recherche d'un optimum, par exemple au moyen d'un algorithme génétique) et l'aspect de théorie des jeux (recherche d'un équilibre de Nash). Cette collaboration, décrite formellement dans la section suivante, fonctionne très bien d'un point de vue pratique et nous semble être une contribution scientifique qui dépasse l'assemblage de « bonnes techniques ».

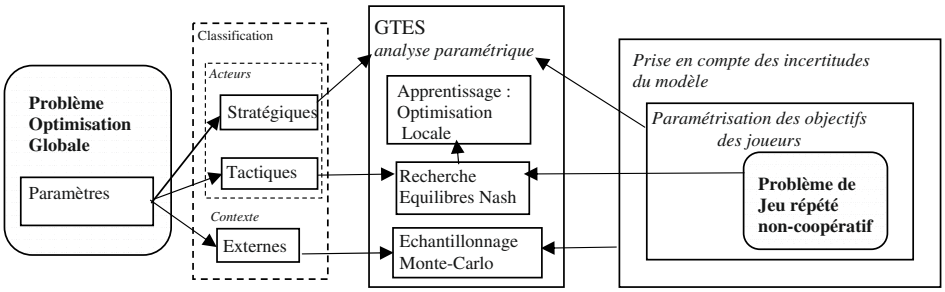

FIGURE 1. Simulation par Jeu et Apprentissage.

D'une façon générale, ce qui est illustré par la figure précédente, il y a deux manières d'arriver au concept de simulation par jeu et apprentissage :

- Soit en partant d'un problème d'optimisation globale $\max_{X \in S} F(X)$ [12] pour lequel nous effectuons une analyse sur le vecteur de paramètre x, en décomposant en trois groupes. En premier lieu nous distinguons des paramètres « externes » qui représentent le contexte, des paramètres liés aux acteurs que nous souhaitons étudier. En deuxième lieu, nous séparons les paramètres représentant les acteurs en un groupe de paramètres qui représentent la stratégie (ce que nous voulons étudier) et un deuxième groupe qui est subordonné au premier (et dont nous voulons nous affranchir de l'étude en laissant un algorithme d'apprentissage – par optimisation locale – découvrir les « bonnes valeurs »). La figure 1 exprime comment chaque famille de paramètre va être traité, ce qui est expliqué dans le reste de cette section.
- Soit nous partons d'un problème de théorie des jeux $\max_{X \in S} f_{\mathrm{p}}(x, e)$ dans lequel x désigne le vecteur de stratégies des n joueurs, et f_{p} est une fonction paramétrique qui représente le vecteur des résultats économiques des joueurs. Nous supposons que e est un paramètre d'environnement, inconnu, dans un espace E.

Pour comprendre l'intérêt de la simulation, il faut poser l'objectif principal de cette approche. Il ne s'agit pas de résoudre un problème d'optimisation, ni même d'aide à la décision. Il s'agit, comme cela a été dit dans l'introduction, d'étudier un modèle et les comportements qu'il représente. Dans les cas d'applications qui sont présentés dans la 5$^{\mathrm{e}}$ section, les logiciels qui ont été développés servent d' « aide à la compréhension ». Nous sommes plutôt dans la tradition des « wargames » que dans un contexte d'aide à la décision. Le « wargame » classique se déroule souvent sous le pilotage d'un animateur. Les décideurs essaient de voir « comment cela se passe » lorsque les acteurs adoptent telle ou telle stratégie. L'approche GTES permet de simuler une partie en automatisant une grande partie du processus : optimisation du jeu des acteurs, prise en compte des aléas.

Un des premiers intérêts d'une telle approche, et nous y reviendrons plus tard, est en fait de tester un modèle. De par sa nature stochastique et itérative, la

méthode GTES est exigeante et détecte facilement les failles de modélisation (*cf.* Sect. 4).

2.2. Notion de Jeu répété

Quelque soit le point de vue de départ, la figure 1 permet de comprendre que le composant central de l'approche GTES est la résolution du sous-problème consistant à optimiser les « tactiques » des acteurs.

Nous nous plaçons dans le cas où les acteurs sont dans un jeu non coopératif, mais peuvent s'observer et ajuster leurs tactiques. Notons que la terminologie que nous avons choisie (parler de stratégie pour les paramètres fixes et connus associés à chaque acteurs, et de tactique pour les paramètres qui sont calculés par la simulation numérique) est logique d'un point de vue « management de l'entreprise », mais elle est différente de celle de la théorie des jeux, puisque c'est justement la « stratégie » qui représente les actions à étudier dans un jeu $\langle S, f \rangle$. Dans la suite de l'article, nous utiliserons tactique pour désigner l'ensemble des paramètres que chaque acteur cherche à optimiser.

La théorie des jeux [6] distingue les jeux à somme nulle (« strictement compétitif »), ce qui n'est pas le cas ici puisque le plus souvent les acteurs partagent une « communauté de destin », par exemple sous la forme d'un marché global qui peut être globalement favorable ou défavorable à l'ensemble des joueurs. Elle distingue également la notion de jeu coopératif ou non-coopératif, selon que les acteurs peuvent ou non synchroniser leurs décisions avant de jouer.

Cette notion d' « ajustement des tactiques » traduit que nous sommes implicitement dans le contexte d'un jeu répété (fini) [15]. Cette sous-section porte sur deux points clés de notre approche :

(1) Nous nous intéressons plus à des tactiques stables qu'à la trajectoire de convergence. Autrement dit, nous cherchons à caractériser la façon dont chaque acteur doit « jouer » pour satisfaire ses objectifs le mieux possible, au lieu de chercher à caractériser la façon dont ils ajustent progressivement leurs tactiques en fonction des signaux qu'ils perçoivent.
(2) Nous allons utiliser un modèle itératif pour trouver un équilibre, car d'un point de vue opérationnel il reproduit ce qui se passe dans la réalité.

Le premier point est une conséquence logique de l'aspect « macroscopique » des modèles que nous pouvons traiter avec l'approche GTES. Étudier la trajectoire serait intéressant mais n'a de sens qu'avec un modèle précis et des échelles de temps précises. Par exemple, il faudrait disposer d'une modélisation précise de temps de réaction des acteurs. Ce n'est pas du tout le cas des modèles auxquels nous appliquerons GTES dans la dernière section. En revanche, dans l'ensemble des cas que nous avons traité, la dimension séquentielle est fondamentale. Les équations associées au modèle décrivent des séquences temporelles, qui permettent de calculer l'état des acteurs à des instants successifs. La notion de séquence permet de s'affranchir de l'échelle de temps.

La tactique d'un acteur pourrait également être représentée par un vecteur qui correspondrait à différentes décisions successives (dans le temps). Du point de vue de la modélisation, nous avions le choix entre deux approches :

- Une vision globale avec un jeu « multi-étape » avec un vecteur tactique temporel.
- Une approche « *repeated game* » ou le même jeu est joué plusieurs fois, mais pour lequel nous cherchons une tactique globale « indépendante du temps ».

La première solution est plus générale mais aboutit à une plus grande complexité de l'espace à explorer. De plus, le choix de la discrétisation des pas de temps n'est pas simple. Nous avons donc choisi de retenir la seconde approche, en partant du principe que nous cherchons à réduire la complexité de cette approche le plus possible, puisqu'il s'agit d'une méthode approchée et exploratoire.

Le second point traduit le fait que rechercher des équilibres par des méthodes itératives, qui peuvent être vues comme des itérations du jeu répété, est naturellement pertinent pour trouver des tactiques stables pour le jeu répété [14]. En effet, nous allons voir dans la section suivante que nous itérons, de façon classique, un opérateur qui optimise la tactique d'un acteur en réponse à celles des autres acteurs. L'autre avantage de cette approche est qu'elle est indépendante du cycle de temps des réactions entre joueurs (qui est toujours difficile à estimer).

2.3. Comportement des acteurs

Le point clé de la simulation est de reproduire de façon la plus réaliste possible le comportement des acteurs. S'il n'y avait qu'un seul acteur, nous serions dans un problème classique d'optimisation [1]. Il s'agirait « simplement » d'optimiser les paramètres « tactiques » pour maximiser le résultat correspondant à une stratégie donnée. Il convient également de noter que, dans ce qui suit, les acteurs sont semblables dans le sens ou ils partagent le même espace de tactiques : cela ne signifie pas que les effets sont les mêmes, mais qu'il existe une structure commune pour décrire les tactiques de l'ensemble des joueurs.

Nous sommes, en revanche, dans le cas d'un scénario à N acteurs, dans une situation de jeu non coopératif. La première idée, la plus naturelle, est de caractériser l'équilibre que cherche à atteindre les acteurs par un équilibre de Nash. Un équilibre de Nash est un choix de tactique telle qu'aucun des acteurs n'a intérêt à changer sa tactique si les autres conservent la leur. En désignant par $f_i(t)$ le résultat obtenu pour l'acteur i et une tactique t fixée, ceci se traduit par la formulation suivante :

$$\forall i, \forall x \in T_i, f_i(x, t_{-i}) \leq f_i(t_i, t_{-i}).$$

Le vecteur des tactiques globales $(t_1, \ldots, t_n)$ est un équilibre de Nash (NE) si et seulement si pour chaque acteur, t_i est optimale en supposant les tactiques des autres acteurs fixées à t_{-i}. Nous reprenons la notation classique de la théorie des jeux dans laquelle t_{-i} représente le $(n-1)$-uple $(t_1, \ldots, t_{i-1}, t_{i+1}, \ldots\ t_n)$.

Le support T_i (*profile*) d'un acteur est l'ensemble des tactiques t_i qu'il peut appliquer. On peut raisonner sur des **tactiques pures** (décrites par les supports T) ou avec des **tactiques mixtes** (combinaisons de plusieurs tactiques avec des probabilités). L'avantage de la structure mixte est de garantir l'existence d'au moins un équilibre de Nash que l'obtient par point fixe (Théorème de Kakutani) d'une séquence d'optimisation (appelée « *best response* » dans le jargon de la théorie des jeux) : $\mathrm{BR}_i(t)$ = la (ou une) tactique pour l'acteur i qui optimise le retour de i si chaque acteur j applique t_j.

Dans le cas « pur », qui est celui qui nous intéresse, il n'existe pas forcément d'équilibre de Nash (*cf.* exemples de la section suivante). Si l'on applique une séquence itérative d'optimisation et si elle converge, on trouve un équilibre de Nash, mais c'est une condition très forte. En revanche, on remarque que si l'on sait construire $\mathrm{BR}_i(t)$ par une technique itérative (BR est lui-même le point fixe d'une séquence d'optimisation), cette approche conduit à un algorithme simple de recherche de l'équilibre :

- sélectionner un acteur ;
- sélectionner une amélioration de sa tactique ;
- répéter jusqu'à stabilisation (facile à mesurer avec une distance) ou « time-out ».

Dans la pratique, la convergence est déclarée lorsque la distance entre les tactiques de deux itérations consécutive est suffisamment faible. La convergence n'est pas assurée, et nous pouvons donc distinguer trois résultats possibles pour un tel algorithme :

- Un jeu qui provoque la convergence est déclaré **stable** (et on trouve de fait un équilibre de Nash associé).
- Un jeu qui provoque une trajectoire divergente, ce qui signifie que le résultat d'au moins un des acteurs constitue une suite strictement décroissante, est qualifié de « **guerre** ».
- Les autres cas sont déclarés « **chaotiques** ».

Un cas chaotique ne signifie pas qu'il n'existe pas d'équilibre de Nash, simplement qu'il n'est pas facile à trouver. Dans la pratique, un équilibre de Nash qui serait particulièrement difficile à trouver n'est pas forcément un bon modèle pour le comportement des acteurs, donc cette distinction en 3 types, même si elle reflète un arbitraire opérationnel (plus la méthode de calcul est pertinente, moins il y a de cas « chaotique »), est pertinente.

2.4. Description informelle de la simulation par Jeux et Apprentissage

Nous pouvons résumer ce qui a été dit dans cette section et proposer une description résumée de la méthode GTES. Nous allons supposer que pour chaque combinaison de stratégie d'acteur, la fonction qui décrit le retour économique est une fonction paramétrique $f_{\mathrm{p}}(t,\ e)$, dont une partie des paramètres (vecteur e)

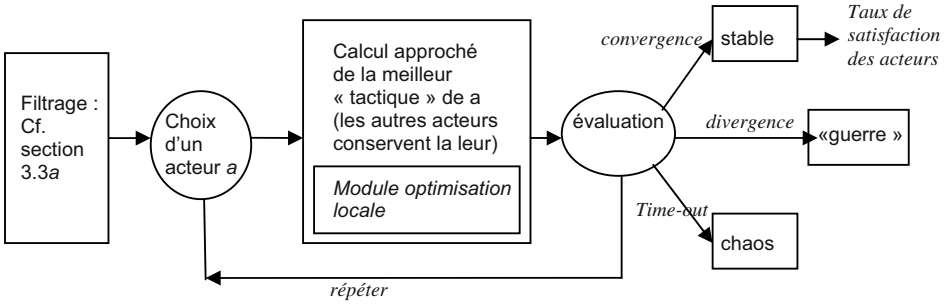

FIGURE 2. Recherche d'un équilibre.

décrit le marché et n'est connu que de façon approximative sous la forme d'intervalle, et l'autre partie (tactique t) peut être optimisée par chaque acteur.

- Les paramètres du premier groupe, dits paramètres économiques, seront gérés par une simulation de type Monte-Carlo.
- Les paramètres qui décrivent les stratégies des acteurs sont identifiés comme paramètre stratégique et vont servir à déterminer une matrice d'interaction (à n dimensions, pour n joueurs).
- Les paramètres du troisième groupe sont les « paramètres tactiques » et nous allons déterminer leurs valeurs par optimisation/apprentissage (similaire à l'apprentissage proposé dans [3]).

Le déroulement de la simulation, pour un ensemble de stratégies fixé (c'est-à-dire pour une case de la matrice) peut être décrit comme suit. Chaque tirage des valeurs des paramètres économiques correspond à une phase. La simulation d'une phase consiste à rechercher un équilibre de Nash dans l'optimisation des paramètres tactiques des acteurs. La figure suivante résume cette recherche d'un équilibre (« une phase »).

Cette recherche d'équilibre consiste à appliquer successivement à chaque acteur une transformation qui optimise sa tactique par rapport à son objectif (défini de façon paramétrique par la stratégie f_p), en supposant les tactiques des autres fixées (approximation de BR). Nous verrons dans la section 4 que plusieurs approches sont possibles, en termes de méta-heuristique d'optimisation et qu'elles sont complémentaires.

Le résultat d'une phase est de trois types :

– Si l'algorithme itératif converge en un nombre fixé et borné d'opération, le résultat est un triplet (catégorie = stable, moyennes des satisfactions des acteurs, déviations de ces satisfactions). Les deux derniers composants du triplet sont donc des vecteurs de taille N.
– Si l'algorithme diverge (voir la définition précise dans la section suivante), le résultat est simplement une valeur (catégorie = guerre).
– Sinon, le résultat est également une simple valeur (catégorie = chaos).

Le résultat global de la simulation est l'agrégation des résultats des phases sous la forme de :

- Les répartitions en pourcentage des catégories (stable, guerre, chaos). Une modélisation est « réussie » si les phases chaotiques sont « rares ».
- Les taux de satisfaction moyens et leurs déviations pour les cas stables.

Puisque nous n'avons aucune information sur la distribution des paramètres externes de E, les moyennes et les déviations n'ont pas de valeur quantitative statistique. En revanche, la moyenne a une valeur qualitative d'approximation de la réussite de la stratégie (modulo la distribution en catégories) et la déviation a une valeur qualitative pour juger de la pertinence (ou stabilité) de cette approximation.

Notons que les stratégies déterminent la fonction d'évaluation d'un résultat et conditionne donc l'optimisation des tactiques (une tactique est optimale pour une stratégie donnée). Les stratégies des acteurs peuvent être différentes (les exemples simples de la section 5, avec des stratégies uniques, ne sont pas représentatifs de l'utilisation réelle). Les stratégies des acteurs sont cachées : ils réagissent aux effets des tactiques.

3. Modèle formel et équilibre entre joueurs

3.1. Modèle du problème posé

Le problème que nous voulons étudier est un problème d'optimisation globale sous incertitude, que nous pouvons représenter par la formule suivante :

$$\max_{x \in X} f_{\mathrm{p}}(x, e), \qquad e \in E$$

dans laquelle x est un vecteur de paramètres numériques sur domaine X, e est un vecteur de paramètres « externes », f_{p} est une fonction paramétrique de $X \times E \rightarrow [0,1]^N$ qui représente la satisfaction de N acteurs. Soit P le domaine du vecteur de paramètres p.

Le premier choix de modélisation est de traiter cette formule comme une équation paramétrique

$$V(e) = \max_{x \in X} f_{\mathrm{p}}(x, e)$$

et d'exclure du champ d'optimisation les paramètres qui détermine le « payoff » des joueurs, ainsi que d'autres paramètres qui sont considérés comme « stratégiques », ce qui signifie que nous nous avons décomposé (*cf.* section précédente) $X = X_s \times T^N$. Notons que la modélisation des acteurs est identique dans le sens ou ils partagent le même espace de tactiques. On suppose qu'il existe une configuration « par défaut » ou « initiale » des acteurs, $\underline{t}$ de T^N

Le second choix de modélisation est le traitement des aléas, pour lequel nous allons procéder par échantillonnage. Nous définissons donc dans un premier temps le résultat d'une simulation pour une valeur donnée de e :

$$\mathrm{GTES}(f,\ T,\ e) =^{\perp} |_{\top}|\ [0,1]^N$$

Qui sera défini par la suite de telle sorte que :

- GTES($f,\ T,\ e$) = (y_i) signifie qu'il existe tel que (t_i) est un *équilibre de Nash local* pour le jeu $\langle T,\ f_{\mathrm{p}}\rangle$ et $f_{\mathrm{p}}((t_i),\ e) = y_{\mathrm{i}}$. De plus, (t_i) est obtenu comme point fixe d'une transformation $t \to \mathrm{GTES}(f,\ t,\ N)$ définie dans la section suivante.
- GTES($f,\ T,\ e$) $=_{\top}$ signifie que l'itération de l'opération GTES($f,\ \_,\ N$), à partir d'une configuration de départ qui est donnée, produit une « suite décroissante ».
- GTES($f,\ T,\ e$) $=^{\perp}$ signifie que cette itération ne converge pas (cas complémentaire des deux autres).

On suppose que l'ensemble des tactiques T est muni d'une structure de voisinage V($V(t)$ est un sous-ensemble de T) et on pose $C(t) = V^*(t)$ (fermeture transitive finie des voisinages). Un *équilibre de Nash local* satisfait une forme relaxée de la définition que nous avons vue précédemment :

$$\forall i, \forall x \in C(t_i), f_i(x, t_{-i}) \leq f_i(t_i, t_{-i}).$$

La méthode GTES, que nous allons maintenant expliciter, est un algorithme heuristique qui renvoie une valeur dans $R \times (R^n \times R^n) \times R$, définie de la façon suivante :

> GTES(f, T, E) = (α, (x, σ), β) signifie que lors d'un tirage aléatoire uniforme de K valeurs de E, la simulation GTES($f,\ T,\ e$) a retourné une valeur réelle (cas stable) dans α (%) des cas, la valeur $\top$ qui indique une divergence dans β % des cas (et la valeur $^{\perp}$ dans $(1 - \alpha - \beta)$ des cas). De plus la moyenne des valeurs réelles est x tandis que l'écart-type est σ.

3.2. Recherche des équilibres entre joueurs

La méthode GTES est une méthode paramétrique qui repose sur une transformation d'optimisation locale τ, qui optimise la tactique d'un acteur en supposant que les autres tactiques sont fixées. Nous noterons $\tau_i(t)$ la transformation de t_i, la $i^{\text{ème}}$ composante de t, en supposant que t_{-i} est fixé. Par construction, nous avons :

$$\forall i, \forall t \in T^N, f^i_{\mathrm{p}}(\tau_i(t), t_{-i}) \geq f^i_{\mathrm{p}}(t_i, t_{-i}).$$

Par la suite, nous allons omettre le paramètre p. On suppose également que la transformation d'optimisation locale est complète pour la structure de voisinage V :

$$\forall i, \forall x \in V(t_i), \exists q, f^i(x, t_{-i}) \leq f^i((\tau_i)^q(t), t_{-i}).$$

On suppose pour finir que la suite $(\tau_i)^q(t)$ converge vers une tactique que nous noterons $\tau_i^*(t)$. Par construction, $\tau_i^*(t)$ est une version « locale » de la « meilleure réponse » de l'acteur i.

La méthode GTES consiste à appliquer de façon itérative cette opération de « meilleure réponse ». Plus formellement, on définit :

$$\mathrm{GTES}(f, t, 1) = \tau_1^*(t) \text{ et } \forall i \in [2, N], \mathrm{GTES}(f, t, i) = \tau_i^*(\mathrm{GTES}(f, t, i-1)).$$

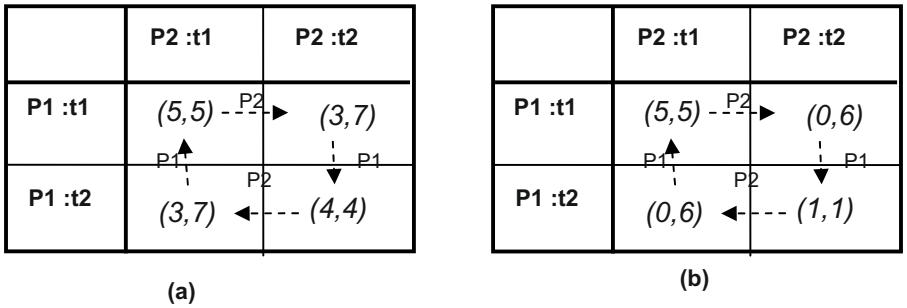

FIGURE 3. Deux cas de non-équilibre de Nash.

On calcule ensuite le point fixe de cette transformation GTES(f, _, N), s'il existe. S'il existe, on a obtenu un équilibre de Nash local par construction. On le démontre très simplement par l'absurde en supposant qu'il existe une tactique w qui viole la condition de « Nash local » pour un des acteurs i tout en appartenant à $C(t^*)$ (en utilisant une récurrence sur la longueur de la chaîne de voisinages qui va de t^* à w).

On remarque qu'il n'est pas nécessaire de calculer le point fixe de τ_i puisque cette première itération est elle-même placée au sein d'une seconde itération. On peut donc proposer, pour un entier q quelconque fixé, une version « entrelacée » de cet algorithme pour lequel une étape d'itération est définie par :

$$\mathrm{GTES}^q(f,t,1) = \tau_1^q(t) \text{ et } \forall i \in [2,N], \mathrm{GTES}^q(f,t) = \tau_i^q(\mathrm{GTES}_i^q(f,t,i-1)).$$

Cet aspect d'entrelacement sera évoqué dans la section 4.2, avec quelques résultats numériques.

3.3. Recherche des équilibres et filtrage

Comme cela a été dit dans la section 2, tous les jeux ne sont pas réductibles à des équilibres de Nash lorsqu'on se limite à des tactiques pures. La figure 3 illustre ce fait avec deux exemples qui provoquent une non-convergence de l'algorithme précédent (illustré par les deux circuits qui représentent, sur la figure, les « *best response* » de chaque acteur par rapport à une case donnée).

Lorsqu'on étudie les jeux « chaotiques » issus de problèmes traité avec l'approche GTES, on s'aperçoit qu'il existe souvent des situations conflictuelles que des acteurs « réels » éviteraient (après un apprentissage pour découvrir que ces « tactiques » sont sans issue). C'est par exemple le cas du second exemple (b) de la figure. Alors que le premier est réellement instable, le second décrit un jeu dans lequel la solution (t_1, t_1) émerge naturellement d'un jeu répété avec de vrais acteurs (en appliquant une évaluation « maxmin », P2 a intérêt à choisir t1, et P1 a donc intérêt ensuite à choisir t1).

Ceci nous a amené à proposer une extension qui peut faire penser à la notion de tactique mixte et que nous utiliserons (*cf.* Sect. 4) comme étape préliminaire dans notre recherche d'équilibres. Au lieu d'appliquer une recherche itérative sur une tactique, nous associons un k-vecteur de tactiques à chaque acteur (par exemple 3). Pour chaque acteur i, S_i est un ensemble de tactiques. La valuation proposée pour cette extension est une minimisation de toutes les combinaisons de tactiques :

$$f^i(S) = \min_{(t_1 \ldots, t_N) \in S} f^i(t).$$

Il s'agit donc de minimiser la prise de risque, c'est-à-dire trouver la tactique qui assure le meilleur retour dans le pire cas. La notion d'équilibre peut s'étendre comme suit.

S est un « k-équilibre de Nash » si et seulement si :

$$|S| = k, \forall i, \forall S' \in T^k, f^i(S', S_{-i}) \leq f^i(S, S_{-i}).$$

La recherche d'un tel équilibre se fait également avec un algorithme itératif. On associe un « *pool* » de k tactiques à chaque acteur. Une étape d'optimisation locale (qui produit une nouvelle tactique pour un acteur i) n'est conservée que si cette tactique apporte une amélioration par rapport à un membre du pool pour la métrique « min » précédemment définie.

L'algorithme itératif est intéressant parce qu'il ressemble à un algorithme génétique, et on peut donc de la même façon construire en parallèle la recherche de la « *best response* » pour l'ensemble des acteurs. Par construction, il est d'autant plus facile de converger que le pool est grand (même si la convergence est exponentiellement longue en fonction de la longueur k de ce pool). En particulier, si la recherche GTES converge vers un équilibre de Nash t^*, la recherche k-étendue converge vers un k-équilibre qui contient k copies du NE, soit {NE} en tant qu'ensemble.

4. Implémentation

4.1. Choix des méthodes d'optimisation locale

En termes d'optimisation locale nous sommes partis des techniques développées pour « découvrir des algorithmes » par apprentissage [3]. Dans cet article, les algorithmes étaient représentés par des termes algébriques et nous avions développé une approche pour explorer l'algèbre. Ici nous avons un problème d'apprentissage plus simple puisque une tactique est un objet qui représente un n-uplet de paramètres numériques.

La méthode la plus simple est une recherche de petits déplacements monodimensionnels. Les voisinages sont obtenus simplement en prenant des variations d'un des paramètres. La méta-heuristique de contrôle est simplement un « *hill climbing* » : une recherche des mouvements qui produisent une suite monotone

d'améliorations. Pour accélérer la convergence, la précision de la recherche (liée au module des variations qui sont considérées) décroît exponentiellement, en suivant le principe d'une recherche dichotomique.

L'approche monodimensionnelle n'est pas toujours suffisante. Il est facile d'adapter et d'ouvrir la structure de voisinage à des déplacements multidimensionnels (variations sur plusieurs paramètres). Nous avons implémenté deux méthodes :

- Mouvements quelconques et exploration randomisée (on tire un déplacement aléatoire). La stratégie de contrôle (méta-heuristique) est soit un *hill climbing* ou une recherche taboue.
- Mouvements sur 2 ou 3 paramètres et exploration systématique selon des algorithmes k-opt (avec $k = 2$ ou 3). Dans un exemple de modélisation de la compétition entre différents canaux de distribution (différents de ceux détaillés dans Sect. 5), nous avions une situation de départ ou les tactiques des acteurs sont connues, et où il n'était pas possible de construire des améliorations sans utiliser de 3-opt, ce qui montre que les acteurs réels avaient déjà fait un travail d'optimisation.

La dernière technique que nous avons implémentée conformément à la section précédente est un algorithme « pseudo-génétique ». Le principe est d'associer à chaque acteur, non plus une tactique mais un vecteur de k tactiques. Comme l'algorithme de calcul de performance (min sur le produit cartésien des ensembles) est de complexité exponentielle k^N, nous nous limitons à $k = 3$ dans la pratique. L'implémentation des étapes d'évolution du vecteur de tactique est classique et reprend ce qui est décrit dans [3] (inspiré, entre autres, par Chap. 6 de [1]) :

- À chaque étape, une nouvelle tactique est introduite dans chaque vecteur.
- Cette nouvelle tactique est obtenue soit par croisement (une « moyenne » des deux n-uplets), soit par mutation (une variation aléatoire multidimensionnelle). Comme cela a été très justement noté par un relecteur, la moyenne devrait être remplacée par un barycentre avec des coefficients aléatoires pour éviter un resserrement trop rapide de l'espace des valeurs. Dans ce qui suit, les résultats sont obtenus en ajoutant une perturbation aléatoire pour éviter ce « resserrement ».
- Comme pour l'algorithme de recherche locale présenté ci-dessous, le taux de mutation est contrôlé par un paramètre qui décroît à chaque itération, de telle sorte que la perturbation aléatoire a une amplitude qui décroit au cours du temps.

L'exécution d'une étape consiste simplement, successivement pour chaque acteur i, à évaluer la valeur minmax de la nouvelle tactique et à éliminer celle dont la valuation « min » est la plus faible (autrement dit, conserver les k meilleures valuations de type maxmin).

Comme cela a été dit précédemment, nous utilisons cet algorithme génétique dans une première phase de l'algorithme, qui agit comme un filtrage (*cf.* Fig. 2). Le « filtrage » consiste à faire démarrer la recherche de point fixe à partir d'un point « remarquable » plutôt qu'un point aléatoire ou arbitraire. L'application de l'algorithme génétique construit un k-vecteur de « bonnes tactiques » pour chaque

	E1b	E2b	E3b	E4b	E5b	E6b	E1	E2	E3	E4	E5	E6
Stable	99 %	77 %	98 %	72 %	71 %	53 %	98 %	83 %	96 %	75 %	72 %	51 %
War	0 %	2 %	1 %	6 %	2 %	8 %	1 %	4 %	2 %	9 %	5 %	11 %
Chaos	1 %	21 %	1 %	22 %	27 %	40 %	1 %	13 %	2 %	16 %	24 %	38 %

FIGURE 4. Intérêt de la réduction de k-vecteurs de tactiques.

joueur. Nous choisissons ensuite la meilleure (selon un minmax) parmi les k. Il n'y a aucune garantie que cette sélection (1 parmi k) produise un équilibre de Nash, ce qui explique pourquoi nous lançons ensuite la recherche itérative par optimisation locale successive.

Voici un exemple tiré de la première application de la section 5. Nous avons reproduit 6 séries d'expériences, une première fois (à gauche, E1b à E6b) sans filtrage, et une seconde fois avec filtrage – E1 à E6). Chaque colonne est associée à une expérience (12 colonnes) et fournit la répartition des phases (produites par échantillonnage) selon les trois catégories expliquées précédemment : stable, guerre et chaos. Un des objectifs du filtrage est de réduire les occurrences de type « chaos » qui sont les situations pour lesquelles on ne peut rien conclure.

Ces résultats confirment l'intuition de la section précédente : la recherche de k-ensembles de tactiques permet d'éviter certaines zones de non-convergence. Cependant, le gain apporté par le filtrage reste mesuré. Malgré différentes tentatives, nous n'avons pas trouvé d'autres façons d'exploiter cette structure de « k-équilibre » mais ce sujet reste ouvert. De la même façon, l'extension à des tactiques mixtes (une combinaison de tactiques avec des probabilités) semble une voie naturelle, mais nous nous heurtons pour l'instant à des problèmes de temps de calculs (*cf.* Sect. 4.3)

4.2. Implémentation globale

L'implémentation de la méthode GTES commence par l'implémentation du modèle que l'on souhaite étudier et qui décrit la fonction f_{p}. L'intérêt de cette méthode est précisément de ne faire aucune hypothèse sur cette fonction qui est un sous-programme indépendant. Il n'y a aucune hypothèse restrictive et le modèle peut être arbitrairement complexe. Dans la pratique, il s'agit d'étudier un « objet que l'on comprend » et les modèles des exemples de la section 5 (ou des autres exemples évoqués précédemment) sont assez simples. En revanche, il s'agit le plus souvent d'une série temporelle qui nécessite plusieurs étapes de calcul (du type $g(t+1) = G(g(t))$, où $g(t)$ représente l'état à l'instant t et G la formule de transformation de l'état courant vers l'état successif). Le résultat de la simulation du modèle est donc une « trajectoire ».

Pour compléter la description de l'implémentation globale, nous pouvons résumer la structure de l'algorithme global par le pseudo-code suivant :

Répéter K fois (nombre d'instanciations Monte-Carlo) :

{ // une phase
Tirer une valeur aléatoire des paramètres externes du scénario (e)
Initialiser les acteurs avec la tactique par défaut ($\underline{t}$)
Filtrage :
{Associer un k-vecteur de tactiques à chaque acteur
Répéter K_1 fois une boucle d'optimisation du vecteur :
Pour chaque acteur i, introduire une nouvelle tactique et l'évaluer
Choisir comme tactique de chaque acteur la meilleure du k_vecteur selon l'évaluation maxmin}
// itération de GTES(f, _, N)
Executer K_2 fois une boucle d'optimisation des paramètres tactiques :
Pour chaque acteur i
Appliquer l'algorithme d'optimisation locale pour produire la nouvelle version de la tactique
RésultatPhase : en fonction de l'analyse des résultats des itérations}

Produire les résultats statistiques et les enregistrer sur un fichier.

La mesure de la convergence, qui permet de déterminer le résultat d'une phase (*RésultatPhase*), s'appuie sur deux mesures :

(1) la mesure de la distance d entre les situations successives après chaque application d'une étape d'optimisation locale. Nous utilisons un agrégateur $A_n = 0{,}8 \times A_{n-1} + 0{,}2 \times d$ qui représente une moyenne pondérée des « dernières distances » constatées lors des cycles d'optimisation ;
(2) la stabilité des résultats obtenus à chaque évaluation, qui est mesurée par l'écart type des satisfactions moyennes ($f^i_{\mathrm{p}}(t_i)$), durant les 10 derniers cycles d'itérations. Autrement dit, nous mesurons la satisfaction moyenne et son écart-type pendant les cycles de $K_2 - 9$ à K_2.

Les critères pour classer une phase selon la définition de la section 3.1 sont les suivants :

– La phase est considérée comme stable si la valeur de An ainsi que l'écart-type des satisfactions moyennes sont suffisamment faibles (le résultat est alors le vecteur des satisfactions moyennes).
– La phase est considérée comme divergente si pour un des acteurs, soit sa satisfaction devient presque nulle (inférieure à 0.1), soit elle décroît à chaque itération d'un incrément significatif (inférieure à –0.04). La valeur renvoyée est $\top$.
– Dans tous les autres cas, la résultat est $\bot$ (*chaos*).

La figure 5 illustre l'architecture logicielle de l'implémentation de la méthode.

La table (Fig. 6) permet de mesurer l'effet de l'entrelacement que nous avons décrit précédemment. Nous comparons trois approches : GR (*great response*) consiste à appliquer un petit nombre de pas d'optimisation local à chaque itération

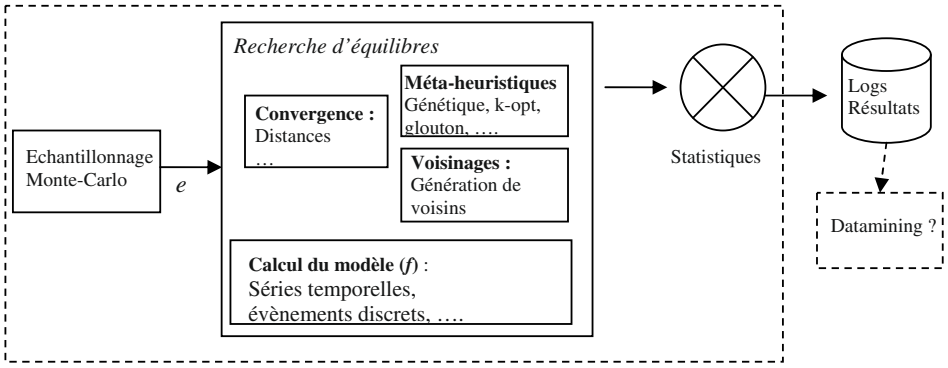

FIGURE 5. Architecture GTES.

	GR	ER	BR
% stable	71	69	62
% chaos	25	27	30
Satisfaction moyenne %	86,7	86,7	81,9
Déviation moyenne %	25,7	10,7	26,3

FIGURE 6. Effet de l'entrelacement de l'optimisation locale.

par acteur (la valeur q de Sect. 3.2), ER (*excellent response*) utilise une valeur plus importante (de l'ordre de 15), tandis que BR est une approximation de la recherche du point fixe τ^*.

Le paramètre K2 est ajusté pour chaque algorithme de telle sorte que le temps de calcul global est constant (plus l'optimisation est poussé à chaque étape, moins il y a d'étape). On voit d'une part que le principe d'entrelacement fonctionne (ER > BR) et d'autre part qu'il y a un réglage à faire pour trouver un bon compromis entre la vitesse de convergence et la stabilité. Dans la pratique, nous utilisons le réglage correspondant à la colonne ER.

4.3. CONSIDÉRATIONS D'IMPLÉMENTATION

L'implémentation de la méthode ne présente aucune difficulté en elle-même. En revanche, des problèmes de performance apparaissent rapidement, qui nécessitent une optimisation des structures de données. Pour illustrer ces contraintes d'exécution, voici des chiffres tirés d'un exemple de l'application de GTES à la simulation des processus d'une entreprise (un autre exemple différent de ceux de la section 5, qui n'est pas repris parce que l'explication du modèle mériterait un article dédié).

– Le calcul du modèle f consiste à simuler une année d'activité et à ordonnancer 25 000 tâches. Après optimisation, cela prend de 1 à 2 s.

	$K = 30$	$K = 60$	$K = 100$	$K = 200$
% de cas stables	92,66	95,50	96,70	98
Satisfaction A %	95	98,38	97,30	99,20
Résultat A %	1215	1234	1235	1228
Ecart-type %	7,87	5,90	7,43	4

FIGURE 7. Stabilité des résultats en fonction du nombre d'instanciations.

– Une itération d'apprentissage nécessite, avec une forme très simple d'optimisation locale, 1000 simulations, soit 25 millions de tâches à ordonnancer. Comme il s'agit d'un exemple à 5 acteurs, cela signifie que l'optimisation de chaque acteur se fait en 200 étapes (entrelacées). La bonne surprise de cet exemple est que ce nombre suffit à obtenir des résultats stables, mais il est clair qu'il serait intéressant d'appliquer une optimisation plus poussée.
– Une expérience, même sous une forme très sommaire limitée à 30 instanciations Monte-Carlo prend 1 journée. Une expérience un peu plus représentative (100 itérations) prend plusieurs jours.

Le nombre de phases nécessaires (les K instanciations Monte-Carlo) dépend bien sûr de chaque problème. Cependant, dans la pratique, la structure du problème est telle que peu d'instanciations suffisent à obtenir des premiers résultats significatifs.

Le tableau (Fig. 7) décrit des résultats obtenus pour le premier problème de la section 5 en fonction du nombre d'itérations. On voit que les premiers résultats sont intéressants (rappelons que seuls les ordres de grandeurs sont significatifs étant donné le côté approximatif du modèle initial), même s'il faut faire beaucoup plus d'instanciations pour obtenir une valeur significative de l'écart-type.

Un autre point ouvert pour améliorer l'efficacité de la méthode est d'ajouter une phase de traitement des résultats, au delà de statistiques très simples (ce qui est indiqué en pointillé sur Fig. 8).

Un des aspects les plus intéressants de la méthode GTES est la capacité à servir de « banc d'essai » des modèles. C'est un point que nous avions déjà constaté dans les précédentes expériences d'apprentissage [3] et qui s'explique de deux façons :

- L'instanciation aléatoire des paramètres (Monte-Carlo) d'une part, et l'exploration de l'espace des tactiques par apprentissage d'autre part, jouent un rôle de « test automatisé » en explorant de façon statistique les différents composants du modèle. Le taux de couverture du programme associé au modèle est généralement excellent.
- La nature de la phase d'apprentissage fait que les « failles » de modélisation, en particulier les conditions aux limites, sont souvent exposées. Par exemple, nous utilisons souvent des modèles linéarisés qui décrivent le comportement d'un système complexe autour d'une position connue. Le principe des « petits déplacements » autorise des équations simplifiées pour décrire, par exemple, des élasticités de marché (quelle réaction si un prix

augmente de quelques euros ?). Ce type de modèle est intrinsèquement fragile si l'on ne prend pas le soin de borner les déplacements autorisés.

Dans les deux exemples de la section suivante (qui sont des modèles qui décrivent un comportement « autour » d'une position initiale), nous avons remplacé des équations linéaires par l'utilisation de « courbes en S ». Cela s'est traduit par une forte augmentation du pourcentage de phases « stables » (*cf.* Sect. 5).

5. Applications

Cette section décrit deux des applications qui ont été réalisées avec l'approche GTES. Il s'agit d'illustrer l'application de la méthode, et non pas de prouver son intérêt. En effet, puisque l'objectif est d'évaluer la pertinence d'un modèle, le succès peut sembler subjectif et est difficile à apprécier en dehors d'un « débat d'expert du métier considéré » qui se heurte rapidement à des considérations de confidentialité.

5.1. Compétition entre des entreprises sur un marché mature

Le premier exemple représente une simulation de trois acteurs dans un marché mature, tel que celui de la téléphonie mobile. Le modèle économique est simple, il décrit le marché selon deux processus, celui de « *churn* » (départ des clients) et celui d'acquisition (obtention de nouveaux clients). Les équations sont des équations « différentielles », en ce sens que les valeurs sont obtenues comme la somme d'une valeur par défaut et d'une différence. Elles sont appliquées pour simuler l'évolution du marché sur 3 ans, mois par mois. Les valeurs par défaut (*churn* et part de marché) sont celles qui ont été publiées en 2005. Les différences sont des fonctions linéaires des changements de prix, de subvention et de fidélisation, encapsulées dans des « courbes en S » pour tenir compte de conditions limites (*cf.* Sect. 4.3). Les paramètres économiques sont précisément ces coefficients d'élasticité du comportement des consommateurs par rapport aux prix.

Les stratégies représentent simplement les objectifs, sous forme d'une trajectoire espérée, en termes de part de marché, d'EBITDA ou de chiffre d'affaire. Par exemple, la stratégie d'un acteur peut être de garantir une croissance de 5 % de son EBITDA par an. Certaines stratégies sont plus agressives que d'autres (par exemple, si chaque acteur souhaite augmenter sa part de marché), mais il est possible de représenter des stratégies plus « financières » (orientées résultat). Le marché est supposé mature : la plupart des « nouveaux » clients quittent un des joueurs pour en rejoindre un autre, même si il reste une légère croissance organique.

La tactique dans ce modèle représente les règles qui vont déterminer le prix de vente du « produit moyen », le montant de la subvention moyenne et le montant de ce qui est dépensé par client dans le programme de fidélisation. La même approche différentielle/linéaire est appliquée : nous partons des valeurs moyennes 2005 avec des coefficients pour introduire des variations linéaires en fonction des variations constatés sur les trois paramètres stratégiques (part de marché réelle *vs.* objectif,

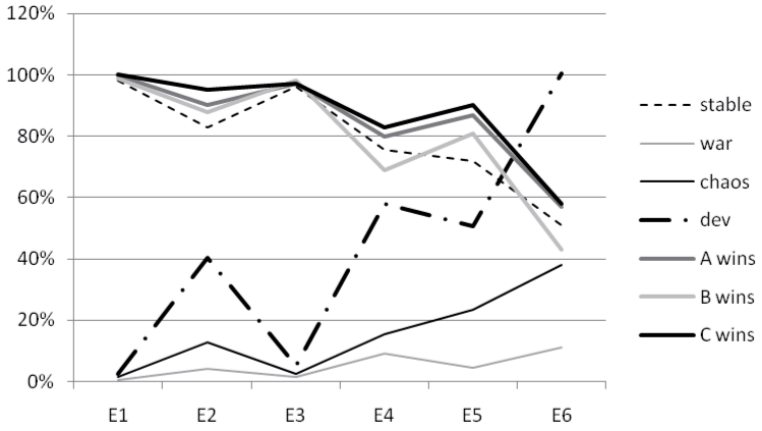

FIGURE 8. Résultats sur 6 combinaisons de stratégies.

EBITDA constaté vs. objectif et chiffre d'affaire vs. objectif). La tactique est donc décrite avec une matrice 3×3. L'apprentissage se fait par optimisation locale (*hill-climbing*) de chacun des 9 coefficients et semble donner de bons résultats : pour une phase et une situation donnée (les jeux des autres acteurs), l'algorithme d'apprentissage converge très rapidement. Une expérimentation est définie par :

- Un scénario qui donne l'espace des paramètres qui doit être exploré par instanciation Monte-Carlo. Une partie importante du travail consiste à valider les valeurs extrêmes des intervalles de variation.
- Des stratégies pour chaque joueur. Nous avons défini 6 stratégies possibles pour les joueurs, dans un ordre plus ou moins croissant d'agressivité (de S1 à S6, comme il y a plusieurs objectifs, il n'y a pas d'ordre strict). Comme expliqué précédemment, la stratégie est définie par des objectifs en termes d'EBITDA, de chiffre d'affaire et de part de marché d'acquisition (l'EBITDA = *Earning Before Interest, Taxes, Depreciation and Amortization* est un indicateur de la marge que dégage l'entreprise). Par exemple, la stratégie S1 correspond à la préservation de l' EBITDA, sous la contrainte que le chiffre d'affaire et la part de marché ne peuvent pas décroître de plus de 1 % par an. A l'inverse, la stratégie S6 correspond à un objectif de croissance annuel de l'EBITDA de 6 % par an, avec un gain de 1 % en part de marché.

La figure 8 représente les résultats de 6 expériences (E_i consiste à choisir la stratégie S_i pour chaque joueur). La figure reproduit les pourcentages de phases stables, de type « guerre » et de type « chaotique ». Ces pourcentages correspondent aux lignes en trait fin sur la figure. Sont également tracés la valeur moyenne de l'écart type des EBITDA (dev) et les pourcentages de succès (taux de satisfaction supérieur à 70 %). L'indication « *A wins* » signifie précisément que le taux de satisfaction de A est supérieur à 70 %.

L'utilisation réelle de cet outil n'est pas de tester l'ensemble des combinaisons, mais plutôt de faire des études de scénarios. Que se passe-t-il si ? ... on introduit un quatrième acteur, si un acteur change de stratégie, etc. Une combinaison telle que E6, dans laquelle l'ensemble des acteurs est en échec par rapport à sa stratégie, n'est pas forcément réaliste.

Au contraire, lorsqu'on s'en tient à des stratégies « réalistes », le taux de stabilité obtenu est de l'ordre de plus de 80 % (le filtrage permettant de gagner 2 à 4 %). Cela signifie que notre modèle correspond à un jeu assez stable, pour lequel il existe des « bonnes » stratégies stables et un processus naturel d'ajustement entre les acteurs, qui conduit à évoluer vers ces stratégies.

Il reste approximativement 5 % de phases qui correspondent à des situations de marchés très difficiles ou l'atteinte des objectifs conduit à une guerre qui produit la disparition d'un des joueurs. Il reste à valider que ces situations sont réellement instables et qu'il ne s'agit pas d'une faiblesse de notre implémentation (par exemple, une meilleure stratégie d'apprentissage collectif pourrait améliorer le taux de convergence).

Un autre intérêt de ces simulations vient des observations que l'on peut faire lorsqu'on dispose d'un grand nombre de résultats. Comme cela a été dit précédemment, il serait intéressant de coupler cette approche avec des techniques de « data mining » pour automatiser ces observation. Par exemple, dans le cas de cette simulation d'un marché à trois acteurs, on voit émerger des *propriétés connues ou suspectées* de cet équilibre (au bout de plusieurs centaines d'expériences, qui mélangent des stratégies variées et qui dépasse le cadre étroit de Fig. 7) :

- La meilleure stratégie du plus petit joueur est d'être un peu plus agressif que les deux autres mais pas trop.
- Le pilotage financier (la définition d'objectifs fondés sur l'EBITDA) conduit à un jeu stable. La recherche de la part de marché est une autre affaire ... qui produit une plus forte mortalité.
- L'augmentation du profit collectif conduit à réduire les coûts d'acquisition, mais à augmenter l'effort de fidélisation.
- Lorsque la compétition augmente (des stratégies plus agressives pour chaque joueur), on observe une pression sur les prix, mais qui est stabilisée par l'impact sur l'EBITDA, puisque le marché est fluide (*cf.* le point sur la stabilité du jeu lorsque la stratégie est dominée par l'EBITDA).
- Le point précédent adoucit l'effet d'une vérité évidente : la guerre économique est favorable aux gros et aux joueurs dont la structure fixe/variable des coûts diminue le « poids mort ».
- Dans ce contexte (les phases stables), la tactique optimale des joueurs conduit à un certain mimétisme : les évolutions des prix, à des niveaux différents, se ressemblent. La « meilleure » tactique produit à la fois de la « prudence » (peu d'agressivité sur la baisse des prix) et un « couplage » des comportements (et non pas une entente).

Bien sûr, ce modèle souffre de nombreux défauts. Par exemple, la tactique est une matrice indépendante de la situation. Cela signifie que ce modèle n'est « valide »

que lorsque les phases correspondent à des évolutions proches de la situation de 2005. Les premiers résultats sont néanmoins intéressants. La prochaine étape, en cours de réalisation, sera de tester une autre forme du modèle qui réponde aux inconvénients de cette première expérience (avec une tactique qui varie « dans le temps » – c'est-à-dire variable selon les périodes de la séquence –, et une meilleure prise en compte des réseaux de distribution).

5.2. Stratégie de distribution

Ce deuxième exemple de l'approche GTES porte sur un problème de stratégie de distribution. La simplicité (relative) du modèle (et du problème posé) permet d'apprécier le fonctionnement et l'intérêt d'une telle approche. Nous avons implémenté ce modèle sur un problème précis, avec de vrais chiffres, mais nous nous contenterons ici de le décrire de façon générique.

Supposons qu'il existe deux compagnies de téléphone A et B qui utilisent plusieurs réseaux de distribution (R1 à R6). Ces réseaux peuvent soit vendre des téléphones (avec un contrat associé), soit procéder à des renouvellements : remplacer un vieux téléphone par un neuf en conservant le contrat existant). La stratégie de distribution de chaque compagnie (A ou B) est définie par quatre montants, pour chacun des réseaux de distribution : le montant de la subvention pour l'achat ou le renouvellement, qui est la somme d'argent implicitement transférée au consommateur, sous la forme de réduction du prix d'achat, et le montant de la rémunération du réseau de vente, également différentiée selon l'achat ou le renouvellement.

On pourrait penser qu'il s'agit donc d'un problème « simple » d'allocation des ressources, chaque compagnie essayant de générer le maximum de flux pour un montant d'investissement de distribution donné. En fait, il existe deux sources de complication : (1) les réseaux sont en compétition pour les même clients (2) chaque réseau peut influencer le client qui entre en contact en fonction de son intérêt propre.

Pour étudier ce couplage et pouvoir comparer les différentes stratégies des deux joueurs A et B, nous utilisons un modèle simple du comportement d'un client pendant deux ans. Ces clients achètent un téléphone au « temps 0 », puis effectuent une deuxième action au bout de 12 mois : ne rien faire, renouveler son téléphone ou décider d'acheter un nouveau téléphone avec un nouveau contrat (ce que l'on appelle le « *churn* »).

Deux « courbes en S » sont utilisées pour modéliser ce marché de 100 clients :

- Une courbe de marché qui donne l'appétence en fonction du prix facial du mobile.
- Une courbe de *churn* qui donne la répartition renouvellement/changement en fonction du différentiel de prix entre un achat et un renouvellement (plus la différence est importante, plus le client décidera de quitter son contrat plutôt que d'utiliser l'offre de renouvellement).

Le modèle proprement dit se décompose en cinq étapes élémentaires :

1. Déterminer les parts de marché globale (A+B) de chacun des réseaux en fonction des prix faciaux pratiqués (c'est-à-dire en tenant compte des subventions). Les déviations, par rapport aux parts de marché initiales, sont obtenues à partir de la courbe en S « de marché ».
2. Déterminer les parts de marché respectives de A et B dans chaque réseau avec la même méthode.
3. Déterminer le taux de *churn* au bout d'un an, en partant d'un taux de souhait de renouvellement (un paramètre du modèle). Le pourcentage des clients qui veulent changer est réparti en deux catégories, *churn* et renouvellement, à partir du taux de churn original, et un ajustement différentiel obtenu à partir de la seconde courbe en S.
4. Les renouvellements sont distribués selon les réseaux de distribution avec une méthode qui est similaire à celle de l'étape 1–2 : on repart des parts de marché constatées pour les renouvellements, avec un ajustement déduit de la courbe en S et des prix faciaux proposés aux clients.
5. Les clients qui « churnent » sont transformés en ventes pour l'année 2 en utilisant le modèle des étapes 1–2 (la même chose une année plus tard).

Si l'on cherche à décomposer ce modèle suivant les trois catégories de l'approche GTES, on obtient :

1. Les paramètres « externes » du modèle sont les deux courbes en S précitées, le taux de changement au bout d'un an et le taux d'indécision, c'est-à-dire la fraction du public qui peut se faire influencer par le réseau pour choisir entre A et B. Les deux derniers paramètres sont assez bien connus, et les sensibilités des courbes en S peuvent être obtenues par différentiation, en supposant que les subventions à l'acquisition sont optimisées par rapport au compromis dépense/acquisition, et que les subventions à l'acquisition sont optimisées par rapport à l'équilibre *churn*/renouvellement.
2. Les stratégies des joueurs (opérateurs) sont les 6×4 paramètres qui ont déjà été présentés (6 canaux $\times$ 2 processus $\times$ subvention/rémunération).
3. Les « tactiques », qui sont calculées par optimisation (locale ou algorithme génétique), représentent les stratégies des réseaux de distribution pour optimiser leur revenus (implicitement, tous les réseaux ont la même stratégie). Ces tactiques sont représentées par 3 chiffres : la sur-subvention que le réseau consent pour être plus compétitif (une partie de la rémunération qui est reversée au client), en acquisition et en fidélisation, et un paramètre qui décrit la façon dont le réseau influence les clients indécis.

Le déroulement d'une simulation est fort simple : étant données les stratégies des deux acteurs-opérateurs A et B, le programme d'optimisation calcule les tactiques optimales de chaque acteur-réseau, pour maximiser ses revenus. On relève alors les résultats financiers des joueurs A et B. Cette méthode permet de prendre en compte la façon dont les réseaux de distribution s'adaptent aux changements de stratégie des joueurs. La sophistication de cette simulation traduit les couplages subtils qui existent entre l'ensemble des parties prenantes. En fait, la réaction des

distributeurs par rapport à un changement de A dépend clairement de la stratégie de B. C'est du simple bon sens : si A essaye d'optimiser ses résultats aux dépens de ses distributeurs, ceux-ci changeront d'allégeance si les conditions de B sont plus favorables.

Cette approche permet de voir que cette adaptation « amortit » les effets positifs du changement que A souhaite mettre en place, mais ne l'annule pas. C'est donc une façon constructive d'instruire le débat entre les partisans du changement qui vont dire que cette nouvelle répartition des investissements commerciaux est plus efficace (ce qu'on « voit » avec un tableur) et les « prudents » qui vont expliquer (à juste titre) qu'une partie d'échecs se juge sur plusieurs coups et que la distribution réagira à ces changements de façon contradictoire.

Cet exemple illustre bien les apports réciproques de la théorie des jeux et des méthodes d'apprentissage pour simuler des processus complexes de coopération.

5.3. Lien avec la dynamique des systèmes

Les deux exemples présentés dans cette section, ainsi que les autres applications qui ont été développées avec cette méthode, sont semblables aux simulations qui sont développées dans le cadre de la « dynamique des systèmes » (« *system dynamics* » [8,16]), dans le domaine de la modélisation et simulation des entreprises. La dynamique des systèmes possède un historique riche de plusieurs dizaines d'années, et s'appuie sur des modèles qui décrivent les relations dynamiques entre les différents paramètres d'un système. Un des points forts de cette école est l'analyse des boucles, selon les différentes « polarités » des interactions (pour une introduction en français enrichie de plusieurs exemples, lire [4]).

La proximité intellectuelle est claire, il s'agit même d'une filiation puisque je me suis inspiré des travaux de Forrester [8] ou de Sterman [16] pour certaines de mes applications. Plus précisément, les modèles qui sont simulés avec l'approche GTES peuvent facilement être vus comme des modèles de dynamique des systèmes, puisque j'utilise le même formalisme (réseau d'indicateurs métiers et de liens de dépendances) pour les construire. Le fait de vouloir caractériser ou comprendre un système par un ensemble de simulations est précisément le cœur de l'approche « dynamique des systèmes ». En revanche, un des principes de la dynamique des systèmes est que la création du réseau est la partie centrale de la simulation, ce qui n'est pas corroboré par mon expérience. Le réseau est facile à obtenir, pour les applications présentées ici par exemple, tandis que le réglage du modèle est plus complexe, parce que la méthode GTES est plus exigeante. En effet, l'utilisation conjointe de la randomisation et de l'optimisation locale ont tendance à « secouer » le modèle pour en faire apparaitre les défauts (*cf.* Sect. 4.3). Au travers des différentes applications développées avec GTES, mon expérience est que :

(1) Le « détail » du modèle est crucial et change radicalement le résultat. La stabilité et le comportement du système, lorsqu'on cherche un équilibre par recherche locale, dépend fortement des équations qui représentent les liens entre indicateurs. Par exemple, si un modèle représente un système à l'équilibre (ce qui est souvent le cas), la stabilité autour de cette équilibre

dépend de la convexité/concavité des fonctions (signe de la dérivée seconde), ce qui est plus subtil qu'une simple indication de polarité (« + » pour des fonctions croissantes et « - » pour des fonctions décroissantes).

(2) En conséquence, il est nécessaire d'avoir une méthode pour investiguer un modèle (on pourrait parler de « *model workbench* »), ce qu'apporte précisément l'approche GTES.

6. Conclusion

La méthode de simulation par jeux et apprentissage est une combinaison de techniques classiques, abondamment utilisées. Cette approche nous semble néanmoins une contribution au domaine de la simulation d'entreprises pour plusieurs raisons. D'une part, il s'agit d'une méthode très générale qui s'applique à un très grand nombre de problèmes, caractérisés par un fort degré d'incertitude et une structure complexe. D'autre part, les résultats obtenus, même s'ils sont préliminaires, sont intéressants. Dans les deux cas, la simulation est capable d'exhiber des propriétés non-triviales et de participer de la sorte à l'analyse du problème posé.

D'un point de vue plus technique, la contribution de cette méthode est de proposer un « patron de conception » pour l'utilisation de méta-heuristiques fondées sur l'optimisation locale pour la recherche d'équilibres de Nash. En particulier, nous avons montré l'intérêt d'utiliser des algorithmes génétiques à la fois pour accélérer la convergence et pour pouvoir guider la recherche initiale vers des parties plus « stables » du paysage.

Il reste encore plusieurs aspects à approfondir. D'un point de vue calcul, cette approche est fortement consommatrice de ressources et il serait utile d'une part de pouvoir paralléliser (ce qui devrait être simple) et d'autre part de continuer à rechercher des structures de données adaptées pour accélérer les traitements unitaires. D'un point de vue théorique, il serait intéressant de voir si la notion de k-équilibre de Nash mérite d'être développée en tant que telle, et non pas comme une phase intermédiaire.

Références

[1] E. Aarts and J.K. Lenstra, *Local search in combinatorial optimisation*. Wiley (1993).

[2] R. Axelrod, *The complexity of cooperation-agent-based models of competitions and cooperation*. Princeton University Press (1997).

[3] Y. Caseau, G. Silverstein and F. Laburthe, Learning hybrid algorithms for vehicle routing problems. *Theory Pract. Log. Program.* **1** (2001) 779–806.

[4] G. Donnadieu and M. Karsky, *La systémique, penser et agir dans la complexité*. Éditions Liaisons (2002).

[5] J. Dréo, A. Petrowski, P. Siarry and E. Taillard, *Métaheuristiques pour l'optimisation difficile*. Eyrolles, Paris (2003).

[6] R. Duncan Luce and H. Raiffa, *Games and decisions – Introduction and critical survey*. Dover Publications, New York (1957).

[7] J. Ferber, Les Systèmes multi-agents : vers une intelligence collective. Dunod, Paris (2007).

[8] J. Forrester, *Principles of systems*. System Dynamics Series, Pegasus Communications, Waltham (1971).
[9] R. Gibbons, *Game theory for applied economists*. Princeton University Press (1992).
[10] S. Jørgensen, M. Quincampoix and T. Vincent (Eds.) *Advances in dynamic game theory: numerical methods, algorithms, and applications to ecology and economics (Annals of the International Society of Dynamic Games)*. Birkhauser, Boston (2007).
[11] G.A. Korn, *Advanced dynamic-system simulation: model-replication techniques and Monte Carlo simulation*. Wiley Interscience (2007).
[12] M. Mongeau, *Introduction à l'optimisation globale*. Bulletin ROADEF N° 19 (2007).
[13] R. Nelson and S. Winter, *An evolutionary theory of economic change*. Belknap, Harvard (1982).
[14] N. Nissan, T. Roughgarden, E. Tardos and V.V. Vazirani, *Algorithmic game theory*. Cambridge University Press (2007).
[15] B. Slantchev, *Game theory: repeated Games*. University of California – San Diego. http://polisci.ucsd.edu/~bslantch/courses/gt/07-repeated-games.pdf (2004).
[16] J. Sterman, *Business dynamics – System thinking and modeling for a complex world*. McGraw Hill (2001).
[17] J. Watson, *Strategy – An introduction to game theory*. Norton (2002).